Title
Geographic Question Answering with Spatially-Explicit Machine Learning Models

Author
Mai, Gengchen

Publication Date
2021

University of California
Santa Barbara

Geographic Question Answering with Spatially-Explicit Machine Learning Models

A dissertation submitted in partial satisfaction
of the requirements for the degree

Doctor of Philosophy
in
Geography

by

Gengchen Mai

Committee in charge:

Professor Krzysztof Janowicz, Chair
Professor Werner Kuhn
Professor Konstadinos Goulias

September 2021

The Dissertation of Gengchen Mai is approved.

Professor Werner Kuhn

Professor Konstadinos Goulias

Professor Krzysztof Janowicz, Committee Chair

June 2021

Geographic Question Answering with Spatially-Explicit Machine Learning Models

by

Gengchen Mai

To my parents and other family members for their love and support

Acknowledgements

Studying at UC Santa Barbara is an incredible experience for me. I still can remember the moment when I first received the offer from UC Santa Barbara. It was right before the 2015 Chinese New Year. I was so excited. I withdrew all the other Ph.D. applications and decided to come to Santa Barbara without any hesitation. I still think this is the wisest decision I have ever made. The Ph.D. study was not an easy journey. I have gone through a lot of setbacks and difficulties in the past six years both in my academic life and daily life. But it is quite a rewarding experience. And there are many people I would like to thank for their love and support during my Ph.D. study.

First, I would like to thank my Ph.D. advisor, Prof. Krzysztof Janowicz, who has spent a great amount of time guiding me through the academic journey for the past six years and made me a mature scientific researcher now. When I first came to Santa Barbara, I was just a young man who graduated with a B.S. degree from Wuhan University who did not have any experience in Geospatial Semantic, Knowledge Graphs, Artificial Intelligence, or even Java & Python programming. Jano was very patient and taught me everything step by step. When I met some difficulties in my research or my daily life, he has always been the first to support me and help me through the difficulties. He is also very supportive and allows me to explore any new research opportunities and topics. I have done five AI/ML-related internships at different high-tech companies such as Esri, SayMosaic, Apple Map, and Google X. Even during my internship, he was always there to help if I met any difficulties or need some life advice. He also supported me to organize workshops, give presentations, and apply for grants and awards whenever I want. He also encourages strong collaborations among our STKO lab members and collaborations among different research groups within or outside of UCSB. From him, I have learned what makes a great researcher, a great collaborator, and a great leader. I feel so lucky to have him as my Ph.D. advisor.

I would like to thank two of my Ph.D. committee members - Prof. Werner Kuhn and Prof. Kostas Goulias - for their support and guidance through my Ph.D. study. Werner is always very interesting in my Ph.D. topic and even provided a GEOG 288WK class about geographic question answering (GeoQA) in Winter 2019. We discussed many challenges and opportunities about GeoQA during that class. Many fruitful thoughts from these discussions become part of my dissertation. Kostas is a very outgoing and humorous person. I enjoyed every discussion with him. He challenged my proposal and pushed me to think about my research from a higher conceptual level. Both Werner and Kostas were also very supportive during my job-seeking process and gave me a lot of great suggestions. I really appreciate their guidance and suggestions.

Next, I would like to thank my “industrial advisor” and “machine learning mentor”, Dr. Ni Lao, for his great mentorship and guidance through my Ph.D. study. Ni was my internship mentor when I interned at SayMosaic Inc. We shared so many common interests and became close friends right after we met. We used to run together every afternoon along the Adobe Creek Loop Trail at Mountain View. During the running, we discussed freely from the grid cells in mammals’ brains to the drawbacks of the current deep learning models. These discussionswent way beyond the internship project and turned out to be very beneficial to my own career. My first paper at the top ML/AI conference - ICLR - was based on one discussion between us during hiking and running. Our friendship continues after my internship and Ni becomes my important ML mentor for my Ph.D. research. Until now we have published six peer-reviewed papers together and we also have four important research items under review. I am so fortunate to meet Ni and he has transformed my thoughts about deep learning and GeoAI research. I also would like to thank Sumang Liu for his supports and many useful life suggestions which helped me a lot especially during the job-seeking process. Moreover, Sumang introduced me to Ni in the first place and encouraged me to dig deeper in the ML/AI field during my internship at Saymosaic. I am so grateful for his suggestion and guidance.

I would like to thank Sathya Prasad from Esri Inc., Nick Martinelli from Apple Maps, and Dr. Hongxu Ma from Google X for their tremendous support during my internships at different companies. They showed me how to make a difference with the knowledge I have learned. Hongxu showed me a lot of very exciting projects at Google X which really blew up my mind. Through these internship experiences, I saw the industry need for geospatial technology which really encouraged me to dive into my current Ph.D. research.

I am so fortunate to have many very supportive lab members at UC Santa Barbara. Dr. Bo Yan, my previous lab member, is my torchbearer on deep learning and machine learning research. We have collaborated on 17 peer-review papers and he is my top one co-author in Google Scholar aside from Jano. I am always amazed by his insightful thoughts and suggestions. Dr. Rui Zhu is like my old brother and always looks after me through my Ph.D. life in Santa Barbara. He is very patient and helpful and a great leader of the whole lab. Ling Cai is my undergraduate classmate back at Wuhan University. It is so wonderful that she came to the STKO lab after three years of master's study at the Chinese Academic of Science. It is great to have my old friend here. Ling is a very smart researcher and helps me a lot through my Ph.D. study. I also would like to thank Dr. Yingjie Hu, Dr. Song Gao, Dr. Blake Regalia, Yiting Ju, Meilin Shi, Dr. Kang Liu, Dr. Ben Adam, Dr. Grant McKenzie, Dr. Jiue-An (Jay) Yang, Anagha Uppal, Zhangyu Wang, Zilong Liu, Dr. Dara Seidl, Su Yeon Han for their help and companion. I feel so attached to the whole STKO family and reluctant to leave the lab.

I would like to thank my roommates here for their companionship. After living with Ning Zhang, Haoxin Zhou, Yao Xuan for 6 years, I feel like we are brothers and we will surely become lifetime friends. In addition, I am grateful to have many close friends here in Santa Barbara who made my Ph.D. life colorful: Jingyi Xiao, Jing Xu, Jianfeng Wang, Tianjiao Yang, Xiaofang Feng, Zhe Fu, Tong Gao, Jiaxiang Wang, Weiming Huang, Mike Johnson, Dami Eyelade, Sara Lafia, Behzad Vahedi, Thomas Hervey, Rafael Ramos, and so on.

Last but foremost, I would like to thank my mother and father as well as my other family members for their unconditional love and support during my Ph.D. study. I still remember it was on August 25th, 2015, my father and mother went to Shanghai with me. With mixed feelings, they saw me off at Shanghai Pudong International Airport. Because of a hurricane, there were thousands of people lining up in front of the security checkpoint. Since it was almost impossible to get on the plane before the gate close, everyone got anxious and I had to run away from my parents to make sure I can get on the plane in time. I got on the plane successfully but sadly failed to say farewell to my parents and gave them a hug. And that was the last time in my life I could talk with my father in person because five months later, my father had a serious brain stem hemorrhage. After staying in the ICU for 3 months and a sanatorium for 7 months, he passed away forever. If I could go back in time, I would not get on that plane. This becomes my lifetime pain point and will accompany me until the end of my life. My mother has such a strong mind (or she has to). She encouraged me to continue my Ph.D. study and took all the pains by herself. Without her unconditional love and support, I will for sure quit my Ph.D. study 5 years ago. Without the support from my grandparents and my aunt, my life will fail down from the hill. So I am very much grateful for all the love and support I have got from my family.

Let me conclude this acknowledgment with the famous sentences in Chu Shi Biao written by Zhuge Liang, “I depart now on a long expedition. Blinded by my own tears, I know not what I write.”

Curriculum Vitæ
Gengchen Mai

Education

2021	Ph.D. in Geography (Expected), Emphasis in Cognitive Science, University of California, Santa Barbara.
2017	M.A. in Geography, University of California, Santa Barbara.
2015	B.S. in Geographic Information System, Wuhan University, China.

Professional Experiences

10/05/2020 - 12/18/2020	**AI/ML Advisor at Google X Inc. Core AI Team**
06/17/2019 - 09/06/2019	**AI Resident at Google X Inc. Core AI Team**
06/17/2019 - 09/06/2019	**Cartographic Engineer Internship at Apple Inc. Map Team**
06/18/2018 - 09/22/2018	**Machine Learning & NLP Research Internship at SayMosaic Inc.**
06/23/2017 - 09/10/2017	**Software Development Internship at Esri Inc. AppStudio Team**
09/2015 - 06/2021	**Graduate Student Researcher at STKO Lab, UC Santa Barbara**

Grants

1. **UCSB Schmidt Fellowship - Research Accelerator Award** (2020 Summer) ($8000): **Gengchen Mai (PI)**: A Hybrid Spatially-Explicit Machine Learning Model for Species Spatio-temporal Distribution Modeling and Biodiversity Hotspot Prediction
2. **Microsoft AI for Earth Grant** (2019-2020) ($15000): **Gengchen Mai (PI)**, Krzysztof Janowicz, Ni Lao, Wenyun Zuo, Ling Cai. Deep species spatio-temporal distribution modeling for biodiversity hotspot prediction.

Fellowships & Awards

06/11/2021	AGILE 2021 Esri Grant
05/27/2021	The Jack and Laura Dangermond Graduate Fellowship ($5000)
05/02/2021	Chinese-American Engineers and Scientists Association of Southern California (CESASC) Scholarship ($1000)
02/2021	AAG Dissertation Research Grants ($1000)

06/15/2020 - 09/27/2020	UCSB Schmidt Fellowship - Research Accelerator Award ($8000)
06/15/2020 - 09/27/2020	UCSB Geography Research Stipend Award ($705)
04/26/2020	ICLR 2020 Travel Grant
2019 - 2020	UCSB Graduate Division Doctoral Student Travel Grant
2019 - 2020	UCSB Graduate Student Association Conference Travel Grant
11/19/2019 - 11/21/2019	Jack & Laura Dangermond Travel Scholarship for K-CAP 2019 ($1000)
11/21/2019	The 1st Best Full Paper Award at K-CAP 2019 (co-author)
07/08/2019 - 07/13/2019	Jack & Laura Dangermond Travel Scholarship for 2019 Esri UC ($501)
06/20/2019	The 1st Best Full Paper Award at AGILE 2019 (lead author)
04/02/2019 - 04/07/2019	Jack & Laura Dangermond Travel Scholarship for AAG 2019 ($600)
11/12/2018 - 11/16/2018	Jack & Laura Dangermond Travel Scholarship for EKAW 2018 ($1500)
11/06/2018 - 11/09/2018	Jack & Laura Dangermond Travel Scholarship for ACM SIGSPATIAL 2018
10/08/2018 - 10/12/2018	Jack & Laura Dangermond Travel Scholarship for ISWC 2018 ($500)
10/08/2018 - 10/12/2018	NSF Student Travel Awards for ISWC 2018 ($800)
08/27/2018 - 08/31/2018	ESRI GIScience 2018 Student Travel Awards ($375)
08/27/2018 - 08/31/2018	Jack & Laura Dangermond Travel Scholarship for GIScience 2018 ($1600)
06/12/2018 - 06/15/2018	Jack & Laura Dangermond Travel Scholarship for AGILE 2018 ($1000)
04/10/2018 - 04/14/2018	Jack & Laura Dangermond Travel Scholarship for AAG 2018 ($600)
03/01/2018 - 03/02/2018	NSF Student Fellowship for US2TS 2018 ($950)
11/07/2017 - 11/10/2017	Jack & Laura Dangermond Travel Scholarship for ACM SIGSPATIAL 2017
04/06/2017	The 1st Best Paper Award at AAG 2017 GIS Student Paper Competition (co-author)
04/01/2017 - 04/09/2017	Jack & Laura Dangermond Travel Scholarship for AAG 2017 ($618)
10/31/2016 - 11/03/2016	Jack & Laura Dangermond Travel Scholarship for ACM SIGSPATIAL 2016

09/27/2016 - 09/30/2016	Jack & Laura Dangermond Travel Scholarship for GIScience 2016 ($1005)
09/05/2016 - 09/09/2016	NSF Student Fellowship for RW2016 ($2000)
09/05/2016 - 09/09/2016	Jack & Laura Dangermond Travel Scholarship for RW2016 ($500)
03/29/2016 - 04/02/2016	Jack & Laura Dangermond Travel Scholarship for AAG 2016 ($500)
09/2015	UCSB Geography Doctoral Scholars Fellowship
06/2015	Outstanding Undergraduate of Wuhan University (20000 Yuan)
2012&2013&2014	China National Fellowship (8000 Yuan Each)
2012&2013&2014	Merit Student of School of RES & ENI SCI, Wuhan University
2012&2013&2014	First-class scholarship of School of RES & ENI SCI, Wuhan University

Publications

Google Scholar Citation[1]: 489; h-index: 11; h10-index: 17 (Date: 06/2021)

Conferences for different domains: ⋆ GIScience; ‡ Machine Learning; † Semantic Web.

Peer-Review Journal Articles

13. **Gengchen Mai**, Krzysztof Janowicz, Yingjie Hu, Song Gao, Bo Yan, Rui Zhu, Ling Cai, Ni Lao. Location encoding for GeoAI. *International Journal of Geographical Information Science.* (Under Review) ⋆

12. **Gengchen Mai***, Yao Xuan*, Wenyun Zuo, Krzysztof Janowicz, Ni Lao. Multi-Scale Representation Learning for Spatial Feature Distributions using Grid Cells and Double Fourier Spheres, *Journal of Machine Learning Research (JMLR)* . (* co-first author) (Under Review) ‡

11. Rui Zhu, Krzysztof Janowicz, Ling Cai, **Gengchen Mai**. Representing and Reasoning Qualitative Higher-order Spatial Relations in Geospatial Knowledge Graphs. *International Journal of Geographical Information Science.* (Under Review) ⋆

10. **Gengchen Mai**, Krzysztof Janowicz, Ling Cai, Rui Zhu, Blake Regalia, Bo Yan, Meilin Shi, Ni Lao. SE-KGE: A Location-Aware Knowledge Graph Embedding Model for Geographic Question Answering and Spatial Semantic Lifting. *Transactions in GIS*, 2020, 24(3) 623-655. `DOI:10.1111/tgis.12629` ⋆

[1] `https://scholar.google.com/citations?user=X2Wfl1UAAAAJ&hl=en`

9. Ling Cai, Krzysztof Janowicz, **Gengchen Mai**, Bo Yan, Rui Zhu. Traffic Transformer: Capturing the Continuity and Periodicity of Time Series for Traffic Forecasting. *Transactions in GIS*, 2020, 24(3) 736-755. `DOI:10.1111/tgis.12644` ⋆

8. **Gengchen Mai**, Krzysztof Janowicz, Bo Yan, Simon Scheider. Deeply Integrating Linked Data with Geographic Information Systems. *Transactions in GIS*, 2019, 23(3) 579-600. `DOI:10.1111/tgis.12538` ⋆

7. Bo Yan, Krzysztof Janowicz, **Gengchen Mai**, Rui Zhu. A Spatially-Explicit Reinforcement Learning Model for Geographic Knowledge Graph Summarization. *Transactions in GIS*, 2019, 23(3) 620-640. `DOI:10.1111/tgis.12547` ⋆

6. Rui Zhu, Krzysztof Janowicz, **Gengchen Mai**. Making Direction a First-Class Citizen of Tobler's First Law of Geography. *Transactions in GIS*, 2019, 23(3) 398-416. `DOI:10.1111/tgis.12550` ⋆

5. Krzysztof Janowicz, Pascal Hitzler, Blake Regalia, **Gengchen Mai**, Stephanie Delbecque, Maarten Frohlich, Patrick Martinent, Trevor Lazarus. On the Prospects of Blockchain and Distributed Ledger Technologies for Open Science and Academic Publishing [Editorial]. *Semantic Web Journal*, 2018, 9(5) 545-555. †

4. **Gengchen Mai**, Krzysztof Janowicz, Yingjie Hu, Song Gao. ADCN: An Anisotropic Density-Based Clustering Algorithm for Discovering Spatial Point Patterns with Noise. *Transactions in GIS*, 2018, 22(1) 348-369. `DOI:10.1111/tgis.12313` *** Top 10% Most Downloaded Papers in TGIS (01/2018-12/2019)** ⋆

3. Rui Xiao, Shiliang Su, **Gengchen Mai**, Zhonghao Zhang, Chenxue Yang. Quantifying determinants of cash crop expansion and their relative effects using logistic regression modeling and variance partitioning. *International Journal of Applied Earth Observation and Geoinformation*, 2015, 34 258-263. ⋆

2. Shiliang Su, Yaping Wang, Fanghan Luo, **Gengchen Mai**, Jian Pu. Peri-urban vegetated landscape pattern changes in relation to socioeconomic development. *Ecological Indicators*, 2014, 46 477-486. ⋆

1. Shiliang Su, Yina Hu, Fanghan Luo, **Gengchen Mai**, Yaping Wang. Farmland fragmentation due to anthropogenic activity in rapidly developing region. *Agricultural Systems*, 2014, 131 87-93. ⋆

Peer-Review CS/GIScience Main Conference Papers

17. Ling Cai, Krzysztof Janowicz, Bo Yan, Rui Zhu, **Gengchen Mai**. Time in a Box: Advancing Knowledge Graph Completion with Temporal Scopes, In: *KDD 2021*. (Under Review)

16. **Gengchen Mai**, Krzysztof Janowicz, Rui Zhu, Ling Cai and Ni Lao. Geographic Question Answering: Challenges, Uniqueness,Classification, and Future Directions, In: *AGILE 2021*. (Accepted)

15. Meilin Shi, Krzysztof Janowicz, Ling Cai, **Gengchen Mai**, Rui Zhu. A Socially Aware Huff Model for Destination Choice in Nature-based Tourism, In: *AGILE 2021.* (Accepted)

14. **Gengchen Mai**, Krzysztof Janowicz, Sathya Prasad, Meilin Shi, Ling Cai, Rui Zhu, Blake Regalia, Ni Lao. Semantically-Enriched Search Engine for Geoportals: A Case Study with ArcGIS Online, In: *Proceedings of AGILE 2020*, Jun. 16 - 19, 2020, Chania, Crete, Greece. `https://doi.org/10.5194/agile-giss-1-13-2020` **(Acceptance Rate ≈35%)** ⋆

13. **Gengchen Mai**, Krzysztof Janowicz, Bo Yan, Rui Zhu, Ling Cai, Ni Lao. Multi-Scale Representation Learning for Spatial Feature Distributions using Grid Cells, In: *Proceedings of the 8th International Conference on Learning Representations (ICLR 2020)*, Apr. 26 - 30, 2020, Addis Ababa, ETHIOPIA. *** Spotlight Paper (Acceptance Rate 6%, 156 out of 2594 submissions)** ‡

12. **Gengchen Mai**, Krzysztof Janowicz, Bo Yan, Rui Zhu, Ling Cai, Ni Lao. Contextual Graph Attention for Answering Logical Queries over Incomplete Knowledge Graphs, In: *Proceedings of ACM K-CAP 2019*, Nov. 19 - 21, 2019, Marina del Rey, CA, USA. **(Acceptance Rate 25%)** †

11. Ling Cai, Krzysztof Janowicz, Bo Yan, **Gengchen Mai**, Rui Zhu. TransGCN: A Translation-Based Graph Convolutional Network Model for Link Prediction, In: *Proceedings of ACM K-CAP 2019*, Nov. 19 - 21, 2019, Marina del Rey, CA, USA. *** 1st Best Full Paper Award (1 out of 28 accepted full papers out of 112 submissions)** †

10. **Gengchen Mai**, Bo Yan, Krzysztof Janowicz, Rui Zhu. Relaxing Unanswerable Geographic Questions Using A Spatially Explicit Knowledge Graph Embedding Model, In: *Proceedings of AGILE 2019*, June 17 - 20, 2019, Limassol, Cyprus. *** 1st Best Full Paper Award (1 out of 19 accepted full papers)** ⋆

9. **Gengchen Mai**, Krzysztof Janowicz, Bo Yan. Support and Centrality: Learning Weights for Translation-based Knowledge Graph Embedding Models, In: *Proceedings of EKAW 2018*, Nov. 12 - 16, 2018, Nancy, France. **(Acceptance Rate 26%)** †

8. Krzysztof Janowicz, Bo Yan, Blake Regalia, Rui Zhu, **Gengchen Mai**. Debiasing Knowledge Graphs: Why Female Presidents are not like Female Popes [Vision Paper], In: *Proceedings of ISWC 2018*, Oct. 8 - 12, 2018, Monterey, CA, USA. †

7. Bo Yan, Krzysztof Janowicz, **Gengchen Mai**, Rui Zhu. xNet+SC: Classifying Places Based on Images by Incorporating Spatial Contexts, In: *Proceedings of GIScience 2018*, Aug. 28 - 31, 2018, Melbourne, Australia. **(Acceptance Rate 37%)** ⋆

6. **Gengchen Mai**, Krzysztof Janowicz, Sathya Prasad, Bo Yan. Visualizing The Semantic Similarity of Geographic Features [Short Paper], In: *Proceedings of AGILE 2018*, June 12 - 15, 2018, Lund, Sweden. ⋆

5. Blake Regalia, Krzysztof Janowicz, **Gengchen Mai**, Dalia Varanka, E Lynn Usery. GNIS-LD: Serving and Visualizing the Geographic Names Information System Gazetteer As Linked Data, In: *Proceedings of ESWC 2018*, June 3 - 7, 2018, Heraklion, Crete, Greece. **(Acceptance Rate 40%)** †

4. Bo Yan, Krzysztof Janowicz, **Gengchen Mai**, Song Gao. Reasoning About Place Type Similarity and Relatedness by Learning Embeddings From Augmented Spatial Contexts, In: *Proceedings of ACM SIGSPATIAL 2017*, Nov. 7 - 10, 2017, Redondo Beach, California, USA. **(Acceptance Rate 18%)** ⋆

3. **Gengchen Mai**, Krzysztof Janowicz, Yingjie Hu, Song Gao. ADCN: An Anisotropic Density-Based Clustering Algorithm, In: *Proceedings of ACM SIGSPATIAL 2016*, Oct. 31 - Nov. 3, 2016, San Francisco Bay Area, California, USA. **(Acceptance Rate 37%)** ⋆

2. Song Gao, Rui Zhu, **Gengchen Mai**. Identifying Local Spatiotemporal Autocorrelation Patterns of Taxi Pick-ups and Drop-offs, In: *Proceedings of GIScience 2016*, Sep. 27 - 30, 2016, Montreal, Canada. ⋆

1. Krzysztof Janowicz, Yingjie Hu, Grant McKenzie, Song Gao, Blake Regalia, **Gengchen Mai**, Rui Zhu, Benjamin Adams, and Kerry Taylor. The Moon Landing or Safari? A Study of Systematic Errors and their Causes in Geographic Linked Data, In: *Proceedings of GIScience 2016*, Sep. 27 - 30, 2016, Montreal, Canada. **(Acceptance Rate 33%)** ⋆

Peer-Review CS/GIScience Conference Workshop Papers

5. **Gengchen Mai**, Krzysztof Janowicz, Cheng He, Sumang Liu and Ni Lao. POIReviewQA: A Semantically Enriched POI Retrieval and Question Answering Dataset [Short Paper], In: *Proceedings of GIR'18 Workshop co-located with ACM SIGSPATIAL 2018*, Nov. 6 - 9, 2018, Seattle, Washington, USA. ⋆

4. **Gengchen Mai**, Krzysztof Janowicz, Bo Yan. Combining Text Embedding and Knowledge Graph Embedding Techniques for Academic Search Engines, In: *Proceedings of SemDeep-4 Workshop co-located with ISWC 2018*, Oct. 8 - 12, 2018, Monterey, CA, USA. †

3. **Gengchen Mai**, Krzysztof Janowicz, Yingjie Hu, Song Gao, Rui Zhu, Bo Yan, Grant McKenzie, Anagha Uppal, and Blake Regalia. Collections of Points of Interest: How to Name Them and Why it Matters [Short Paper], In: *Proceedings of Spatial big data and machine learning in GIScience Workshop at GIScience 2018*, Aug. 28 - 31, 2018, Melbourne, Australia. ⋆

2. Blake Regalia, Krzysztof Janowicz, **Gengchen Mai**. Phuzzy.link: A SPARQL-Powered Client-Sided Extensible Semantic Web Browser, In: *Proceedings of VOILA2017 co-located with ISWC 2017*, Oct. 22, 2017, Vienna, Austria. †

1. **Gengchen Mai**, Krzysztof Janowicz, Yingjie Hu, Grant McKenzie. A Linked Data Driven Visual Interface for the Multi-Perspective Exploration of Data Across Repositories, In: *Proceedings of VOILA2016 co-located with ISWC 2016*, Oct. 17, 2016, Kobe, Japan. †

Book Chapters

3. Song Gao, Mingxiao Li, Jinmeng Rao, **Gengchen Mai**, Timothy Prestby, Joseph Marks. (2019) Automatic Urban Road Map Generation from Massive GPS Trajectories of Taxis. In Martin Werner, Yao-Yi Chiang et al.(Eds): *Handbook of Big Geospatial Data*, Springer.
2. Bo Yan, **Gengchen Mai**, Yingjie Hu, Krzysztof Janowicz. (2019) Harnessing Heterogeneous Big Geospatial Data. In Martin Werner, Yao-Yi Chiang et al.(Eds): *Handbook of Big Geospatial Data*, Springer.
1. Song Gao, **Gengchen Mai**. (2017) Mobile GIS and Location-Based Services. In Bo Huang, Thomas J. Cova, and Ming-Hsiang Tsou et al.(Eds): *Comprehensive Geographic Information Systems*, Elsevier. Oxford, UK.

Patents

3. **United States Patent (2020)**: Hongxu Ma, **Gengchen Mai**, Bin Ni. Neural Location Encoders for Large Scale Geospatial Prediction Problems. (Under Review)
2. **United States Patent (2020)**: Hongxu Ma, **Gengchen Mai**, Bin Ni, Charlotte Leroy, Grigory Bronevetsky. Deep neural network based Spatiotemporal Casual Inference Model. (Under Review)
1. **United States Patent (2020)**: **Gengchen Mai**, Cheng He, Sumang Liu, Ni Lao. System and Method for Natural Language Processing (NLP) based Searching and Question Asnwering.

Conference Presentations

25. **Online Video presentation (2021)**: Location Encoding and Spatially-Explicit Machine Learning, In: *China GIScience Symposium Series No. 5: GeoAI*, Feb 16, 2021, Beijing, China.
24. **Online Video presentation (2020)**: Space2Vec: Multi-Scale Representation Learning for Spatial Feature Distributions using Grid Cells, In: *Thinkspatial Tech Talk at Spatial@UCSB*, Dec 1, 2020, Santa Barbara, CA, USA. (Online Due to COVID-19)

23. **Online Video presentation (2020)**: Space2Vec: Multi-Scale Representation Learning for Spatial Feature Distributions using Grid Cells, In: *Google AI Tech Talk 2020*, August 12, 2020, Mountain View, CA, USA. (Online Due to COVID-19)
22. **Online Video presentation (2020)**: SE-KGE : A Location - Aware Knowledge Graph Embedding model for Geographic Question Answering and Spatial Semantic Lifting, In: *Esri UC 2020*, July 13 - 16, 2020, San Diego, CA, USA. (Online Due to COVID-19)
21. **Spotlight presentation (2020)**: Multi-Scale Representation Learning for Spatial Feature Distributions using Grid Cells, In: *ICLR 2020*, Apr. 26 - 30, 2020, Addis Ababa, ETHIOPIA. (Online Due to COVID-19)
20. **Oral presentation (2020)**: Knowledge Graphs and Spatiotemporal Data, In *NSF OKN VoCamp 2020*, Jan. 27 - 28, 2019, Santa Barbara, CA, USA.
19. **Oral presentation (2019)**: Knowledge Graphs and Spatiotemporal Data, In *Spatial Data Science Symposium 2019*, Dec. 9 - 11, 2019, Santa Barbara, CA, USA.
18. **Oral presentation (2019)**: Contextual Graph Attention for Answering Logical Queries over Incomplete Knowledge Graphs, In *K-CAP 2019*, Nov. 19 - 21, 2019, Marina del Rey, CA, USA.
17. **Oral presentation (2019)**: A Spatially Explicit Machine Learning Model for Map Generalization, In *Apple Map Leadership Presentation*, August 15, 2019, Sunnyvale, CA, USA.
16. **Poster presentation (2019)**: A Spatially Explicit Machine Learning Model for Map Generalization, In *Apple Intern Machine Learning Symposium 2019*, August 1, 2019, Cupertino, CA, USA.
15. **Oral presentation (2019)**: Deeply Integrating Linked Data with Geographic Information Systems, In *Esri UC 2019*, July 8 - 13, 2019, San Diego, CA, USA.
14. **Oral presentation (2019)**:Relaxing Unanswerable Geographic Questions Using A Spatially Explicit Knowledge Graph Embedding Model, In *Spatial Discovery III*, May 1 - 3, 2019, Santa Barbara, CA, USA.
13. **Oral presentation (2019)**:Relaxing Unanswerable Geographic Questions Using A Spatially Explicit Knowledge Graph Embedding Model, In *GeoAI and Deep Learning Symposium: Geo-Text Data and Location-based Social Media*, April 2 - April 7, 2019, Washingtin D.C., USA.
12. **Oral presentation (2018)**: Support and Centrality: Learning Weights for Translation-based Knowledge Graph Embedding Models, In *EKAW 2018*, Nov. 12 - 16, 2018, Nancy, France.
11. **Oral presentation (2018)**: POIReviewQA: A Semantically Enriched POI Retrieval and Question Answering Dataset, In *GIR'18 Workshop co-located with ACM SIGSPATIAL 2018*, Nov. 6 - 9, 2018, Seattle, Washington, USA.

10. **Oral presentation (2018)**: Combining Text Embedding and Knowledge Graph Embedding Techniques for Academic Search Engines, In *SemDeep-4 Workshop co-located with ISWC 2018*, Oct. 8 - 12, 2018, Monterey, CA, USA.

9. **Oral presentation (2018)**: xNet+SC: Classifying Places Based on Images by Incorporating Spatial Contexts, In *GIScience 2018*, Aug 27 - 31, 2018, Melbourne, Australia.

8. **Oral presentation (2018)**: Collections of Points of Interest: How to Name Them and Why it Matters, In *Spatial big data and machine learning in GIScience Workshop at GIScience 2018*, Aug 27 - 31, 2018, Melbourne, Australia.

7. **Oral presentation (2018)**: Visualizing The Semantic Similarity of Geographic Features, In *AGILE 2018*, June 12 - 15, 2018, Lund, Sweden.

6. **Oral presentation (2018)**: Visualizing The Semantic Similarity of Geographic Features, In *Artificial Intelligence and Deep Learning Symposium: Geospatial Semantics and Geo-Text Data Analytics II*, April 10 - April 14, 2018, New Orleans, Louisiana, USA.

5. **Oral presentation (2017)**: A Semantically Enabled Geographic Information Retrieval Framework by using Representation Learning: A Simple Case Study of DBpedia, In *GIS Day@UCSB Geography*, November 17, 2017, Santa Barbara, California, USA.

4. **Oral presentation (2017)**: ADCN: An Anisotropic Density-Based Clustering Algorithm for Discovering Spatial Point Patterns with Noise, In *2017 Annual Meeting of AAG: Spatiotemporal Symposium – Big Spatiotemporal Data Discovery and Mining Session*, April 5- April 9, 2017, Boston, Massachusetts, USA.

3. **Poster presentation (2016)**: ADCN: An Anisotropic Density-Based Clustering Algorithm, In *24th International Conference on Advances in Geographic Information Systems (ACM SIGSPATIAL 2016)*, October 31 - November 3, 2016, San Francisco Bay Area, California, USA.

2. **Poster presentation (2016)**: A Linked Data Driven Visual Interface for the Multi-Perspective Exploration of Data Across Repositories, In *12th Reasoning Web Summer School (RW 2016) co-located with 10th International Conference on Web Reasoning and Rule Systems (RR 2016)*, September 5 - 9, 2016, Aberdeen, Scotland, UK.

1. **Oral presentation (2016)**: Tea Plantation Expansion in Hangzhou, China: Process, Related factors & Ecological Effect, In *2016 Annual Meeting of AAG: The Quest to Map Plant Species Session*, March 28- April 1, 2016, San Francisco, California, USA.

Teaching Experiences

2020 Winter	Guest Lecturer of GEOG 176B: Lecture 9: Analysis, Buffers and Map Algebra
2019 Spring	Guest Lecturer of GEOG 176C: Geographic Question Answering
2018 Spring	Guest Lecturer of GEOG 176C: Geographic Knowledge Graph
2017 Spring	Guest Lecturer of GEOG 176C: GIS Applications - Project Proposal
2017 Winter	Guest Lecturer of GEOG 176B: Lecture 7 - Conceptual Modeling and Semantics
2016 Summer	Teaching Assistant of GEOG 3B: Land, Water & Life
2016 Spring	Teaching Assistant of GEOG 176C: GIS Application

Peer Reviewer for Academic Journals

International Journal of Geographic Information Science
Transactions in GIS
Journal of Spatial Information Science
IEEE Geoscience and Remote Sensing Letters
PLOS ONE
Journal of Transport Geography
Semantic Web Journal
Annuals of GIS
Habitat International
Social Indicators Research
Land Use Policy

Conferences or Workshop Committees

2021	**Workshop Organizer:** GIScience 2021 - Geospatial Knowledge Graphs and GeoAI Workshop **Metadata Chair:** GIScience 2021; **PC Member:** GIScience 2021, ESWC 2021, ISWC 2021
2020	**PC Member:** ESWC 2020, ISWC 2020; **Subreviewer:** ACM SIGSPATIAL 2020, ISWC 2020
2019	**PC Member:** LDOW-LDDL 2019, ESWC 2019, ISWC 2019, K-CAP 2019 **Student Organizer:** SDSS 2019 **Session Organizer:** AAG 2019 - Geo-Text Data and Location-based Social Media session
2018	**Local Student Organizer:** IOT 2018
2017	**PC Member:** K-CAP 2017

Services

2017/09 - 2018/09	Campus Ambassador of Esri Inc. at UCSB
2016/09 - 2017/08	Program Manager of UCSB Cognitive Science
2011/09 - 2013/06	Undersecretary of Department of Life and Welfare in the Student Union of School of Resource and Environmental Sciences, Wuhan University

Links:

Website:	`https://gengchenmai.github.io/`
Linkedin:	`https://www.linkedin.com/in/gengchen-mai-144439121/`
Google Scholar:	`https://scholar.google.com/citations?user=X2Wfl1UAAAAJ&hl=en`
GitHub:	`https://github.com/gengchenmai`
ResearchGate:	`https://www.researchgate.net/profile/Gengchen-Mai-2`

Abstract

Geographic Question Answering with Spatially-Explicit Machine Learning Models

by

Gengchen Mai

As an important part of Artificial Intelligence (AI), Question Answering (QA) aims at generating answers to questions in natural language. With the advancement of deep learning technology, we have witnessed substantial progress in open-domain question answering. However, QA systems are still struggling to answer questions that involve geographic entities or concepts and that require spatial operations. In order to tackle these challenges, this dissertation specifically focuses on the problem of Geographic Question Answering (GeoQA) and develops a series of spatially-explicit machine learning models to handle different GeoQA tasks. First, in Chapter 1, we discuss the challenges of answering geographic questions and the uniqueness of GeoQA. A classification of geographic questions has been presented to facilitate the development of GeoQA. Next, in Chapter 2 a spatially-explicit query relaxation model is presented to demonstrate the usefulness of geographic information and spatial thinking in the geographic question answering and query relaxation process. To develop a more generalizable approach for GeoQA and other geospatial tasks, in Chapter 3, we present a general-purpose multi-scale representation learning model for geographic locations which can be utilized in multiple downstream tasks. It has been later on utilized to build a location-aware knowledge graph embedding model for a knowledge graph-based GeoQA model in Chapter 4. Only relying on points as the spatial representations for geographic entities is not sufficient to answer many geographic questions that involve spatial relations such as topological relations and cardinal direction relations. So in Chapter 5, we present a polygon encoder that can be used to answer multiple types of spatial relation questions. In the end, we draw a conclusion by listing several challenges of GeoQA which have not been solved in this dissertation and point out some future research directions. We hope this dissertation can reveal the importance of GeoQA and demonstrate the usefulness of spatially-explicit machine learning models on geospatial problems. We also hope GeoQA will become a unique research domain and serve as an important part of Geographic Artificial Intelligent (GeoAI) research.

Contents

Chapter 1

Geographic Question Answering: Challenges, Uniqueness, Classification, and Future Directions

As the general introduction to this dissertation, this chapter introduces the problem of geographic question answering (GeoQA) and investigates it from a conceptual perspective. We motivate the necessity of GeoQA by showing several challenging yet commonly asked geographic questions that are difficult to answer even for the current state-of-the-art question answering (QA) systems such as Google's QA system. Then we discuss the challenges and uniqueness of GeoQA compared with the common (open-domain) question answering problem including various spatial representation manipulation, spatial operator interpretation and selection, spatial language variability, geometric uncertainty, and vagueness of geographic information. Next, we review the current landscape of GeoQA research based on which we provide a generic classification framework for geographic questions. Several research directions are identified some of which are the focus of this dissertation. Last but not least, we discuss the three major research questions and the synopsis of this dissertation. These three questions will, later on, be answered by Chapter 2 (RQ1), Chapter 3 (RQ2), Chapter 4 (RQ2), and Chapter 5 (RQ3). This chapter is an extended version of the published paper shown in Table 1 by adding Section 1.8 which provides the structure of this dissertation.

Peer Reviewed Publication	
Title	Geographic Question Answering: Challenges, Uniqueness, Classification, and Future Directions
Authors	Gengchen Mai, Krzysztof Janowicz, Rui Zhu, Ling Cai, Ni Lao
Venue	AGILE 2021: Geospatial Technologies: On The Verge Of Change
Editors	Panagiotis Partsinevelos, Phaedon Kyriakidis, Marinos Kavoyras
Publisher	AGILE: GIScience Series of Copernicus Publishers
Submit Date	March 12, 2021
Accepted Date	April 9, 2021
Publication Date	June 8, 2021
Copyright	Reprinted with permission from AGILE 2021
DOI	https://doi.org/10.5194/agile-giss-2-8-2021

Abstract: As an important part of Artificial Intelligence (AI), Question Answering (QA) aims at generating answers to questions phrased in natural language. While there has been substantial progress in open-domain question answering, QA systems are still struggling to answer questions which involve geographic entities or concepts and that require spatial operations. In this paper, we discuss the problem of geographic question answering (GeoQA). We first investigate the reasons why geographic questions are difficult to answer by analyzing challenges of geographic questions. We discuss the uniqueness of geographic questions compared to general QA. Then we review existing work on GeoQA

and classify them by the types of questions they can address. Based on this survey, we provide a generic classification framework for geographic questions. Finally, we conclude our work by pointing out unique future research directions for GeoQA.

1.1 Introduction

"Another example of a good language problem is question answering, like 'What's the second-biggest city in California that is not near a river?' If I typed that sentence into Google currently, I'm not likely to get a useful response."[1] – Dr. Michael Jordan, UC Berkeley [1]

Question Answering (QA) lies at the intersection of natural language processing (NLP), information retrieval (IR), knowledge representation, and computational linguistics. It aims at generating or retrieving answers to questions asked in natural language [2]. Question answering is an important part of artificial intelligence (AI) research [3] and has recently permeated to our daily lives. Many commercial language understanding systems or voice control systems are widely adopted by the general public such as Apple Siri, Amazon Alexa, Google's assistant, Xiaomi Xiaoai, and so on.

Generally speaking, question answering systems can be classified into three categories based on the types of data sources [2]: unstructured data-based QA [4, 5, 6, 7, 8], semi-structured table-based QA [9], and structured data source-based QA (so-called semantic parsing) [10, 11, 12, 13, 14, 15]. Thanks to the recent development of multiple open domain QA datasets such as HotpotQA [16], SQuAD Open [7], and Natural Questions Open [17], research on unstructured data-based QA has made substantial progress [18, 19, 20]. Recently, we have also seen remarkable advancements in hybrid QA models

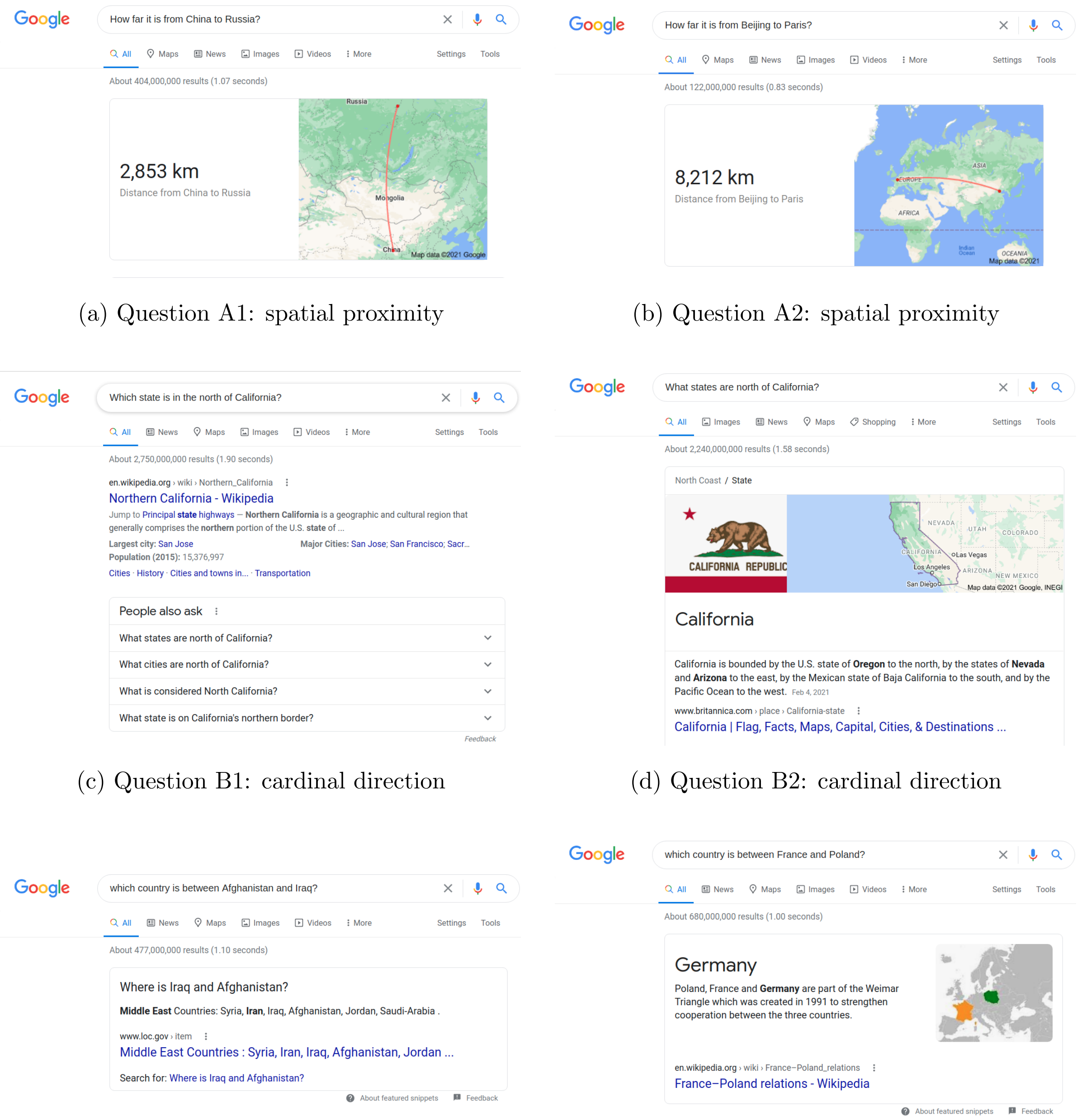
[1] Interestingly, now Google can correctly answer this geographic question based on reading comprehension over an Wikipedia article. Nevertheless, using reading comprehension to answer this kind of geographic questions is problematic and suffers from data sparsity issue (See Section 1.2).

which rely on different data sources, such as hybrid QA models based on both knowledge graphs and unstructured texts [21, 22].

Although the performance gap between human's and deep neural network-based QA models has been significantly reduced on reading comprehension style QA tasks [4], we still get a fairly poor performance when applying these models in the wild. Even commercial QA products such as Google question answering system are struggling to answer many simple geographic questions. Figure 1.1 shows several challenging geographic questions which shows the limitation of Google QA system that is powering their search.

In this work, we define *geographic questions* as questions that involve geographic entities (e.g., Los Angeles, Eastern Sierra), geographic concepts (e.g., feature types such as Building, City, State), or spatial relations (e.g. near to, north of, between) as parts of the natural language questions. Note that this definition is rather broad compared to related notions such as geo-analytical questions [23] which require geo-analytical workflows (in GIS) to answer them. The corresponding QA systems and processes are named geographic question answering (GeoQA). While some geographic questions are easy to answer such as *what is the population of London* or *where is Los Angeles* as they only require a simple property fact lookup in a knowledge base/graph, other geographic questions are more challenging to handle even for state-of-the-art (SOTA) question answering systems.

Figure 1.1 shows three pairs of geographic questions which demonstrate the limitation of Google QA. Question A1 & A2, B1 & B2, and C1 & C2 involve three different types of spatial operations in order to answer geographic questions, namely spatial proximity, cardinal direction, and projective ternary relation (e.g., betweenness) [24]. While Google QA can provide meaningful answers to Question A2, B2, and C2 as shown in Figure 1.1b, 1.1d, and 1.1f, it can not handle simple variations of them (Question A1, B1, and C1 as shown in Figure 1.1a, 1.1c, and 1.1e). A1, A2, B1, and B2 are simple questions or



How far it is from China to Russia?

About 404,000,000 results (1.07 seconds)

2,853 km

Distance from China to Russia

(a) Question A1: spatial proximity

How far it is from Beijing to Paris?

About 122,000,000 results (0.83 seconds)

8,212 km

Distance from Beijing to Paris

(b) Question A2: spatial proximity

Which state is in the north of California?

About 2,750,000,000 results (1.90 seconds)

en.wikipedia.org › wiki › Northern_California

Northern California - Wikipedia

Jump to Principal **state** highways — **Northern California** is a geographic and cultural region that generally comprises the **northern** portion of the U.S. **state** of ...

Largest city: San Jose
Major Cities: San Jose; San Francisco; Sacr...
Population (2015): 15,376,997

Cities · History · Cities and towns in... · Transportation

People also ask

What states are north of California?

What cities are north of California?

What is considered North California?

What state is on California's northern border?

(c) Question B1: cardinal direction

What states are north of California?

About 2,240,000,000 results (1.58 seconds)

North Coast / State

California

California is bounded by the U.S. state of **Oregon** to the north, by the states of **Nevada** and **Arizona** to the east, by the Mexican state of Baja California to the south, and by the Pacific Ocean to the west. Feb 4, 2021

www.britannica.com › place › California-state

California | Flag, Facts, Maps, Capital, Cities, & Destinations ...

(d) Question B2: cardinal direction

which country is between Afghanistan and Iraq?

About 477,000,000 results (1.10 seconds)

Where is Iraq and Afghanistan?

Middle East Countries: Syria, **Iran**, Iraq, Afghanistan, Jordan, Saudi-Arabia .

www.loc.gov › item

Middle East Countries : Syria, Iran, Iraq, Afghanistan, Jordan ...

Search for: Where is Iraq and Afghanistan?

(e) Question C1: betweenness

which country is between France and Poland?

About 680,000,000 results (1.00 seconds)

Germany

Poland, France and **Germany** are part of the Weimar Triangle which was created in 1991 to strengthen cooperation between the three countries.

en.wikipedia.org › wiki › France–Poland_relations

France–Poland relations - Wikipedia

(f) Question C2: betweenness

Figure 1.1: Three pairs of geographic questions that show the limitation of Google's question answering system. Google's question answering system fails to answer Question A1, B1, C1 while being able to handle A2, B2, C2 even if these three pairs A1 - A2, B1 - B2, C1 - C2 are quite similar to each other. All screenshots obtained on Feb. 17th, 2021.

so-called single-relation factoid questions [25] which can be answered by using a single triple in a Knowledge Graph (KG), if available. C1 and C2 are expected to be answered based on two triples in a KG. These questions show interesting properties shared by geographic questions and give us hints about why geographic questions are difficult to handle.

In this paper, we aim at answering the following three research questions:

1. *Why* are geographic questions difficult to answer compared to generic questions?
2. *How* to classify geographic questions?
3. *What* unique contributions can GIScience make in GeoQA in addition to SOTA approaches instead of reinventing the wheel?

In the following, we will go through those geographic questions in Figure 1.1 and discuss the reason why current QA system fail. Next, we discuss the uniqueness of geographic questions and GeoQA in Section 1.3 from a conceptual level. Then, in Section 1.4, we present existing work on GeoQA by classifying them into different groups based on the types of questions they can handle and discuss pros and cons of them. Section 1.5 provides a detailed classification of geographic questions and discusses the possible solutions and challenges of GeoQA for each question type. Last, we conclude this paper by discussing possible future research directions in GeoQA.

1.2 Why Geographic Questions are Difficult to Answer?

In this section, we discuss the reasons why geographic questions are hard to answer by using the three pairs of geographic questions presented in Figure 1.1.

1. *QA systems usually lack proper spatial representations (i.e., points, polylines, or polygons) for geographic entities.* Question A1 shown in Figure 1.1a is actually a brain teaser question. The correct answer is 0 since China is adjacent to Russia [26]. Although Google QA successfully recognizes the geographic entities involved in the question – China and Russia, it picks the wrong spatial representation (i.e., points) for spatial proximity computation. In fact, it is common practice for many widely used knowledge graphs such as Wikidata and DBpedia to represent all geographic entities as points regardless of their scale. Consequently, many QA systems based on these KBs would inherit this limitation.

2. *Polygon-based spatial operations, such as the calculation of spatial proximity and topological relations between geographic entities, are computationally expensive.* Many geographic entities are represented by polygons with thousands of vertices, and, thus, spatial operations performed on them are difficult to carry out on demand. For Question A1, although Google Maps has the polygon representations for China and Russia, it seems to always pick point geometries for the sake of fast response time.

3. *The selection of spatial operator is subject to context* - *where* a user asks a question, *when* they ask it, *which* geographic entities they are comparing. Both Question A1 and A2 have exactly the same query template - *how far it is from X to Y*. The reason why Google QA can successfully answer Question A2 but not A1 is because the scales of the compared geographic entities are different. For A2, Paris and Beijing are far enough and thus can be presented at a small map scale. Their fine-grained geometries, i.e., polygons, can be "safely" ignored and we can use points to represent their locations. However, as for Russia and China in A1, since they are adjacent to each other, their polygon representations are too large to be ignored.

How to pick the correct spatial representations and their corresponding spatial operators is challenging and depends on the map scale tied to the question[2].

4. *Reading comprehension based QA cannot easily handle geographic questions.* Instead of computing the answers based on the geometries of geographic entities, many SOTA QA systems try to answer geographic questions by answering questions based on text corpus [19] which suffer from data sparsity. For example, Google QA tries to answer cardinal direction questions such as Question B1, B2 in Figure 1.1c, 1.1d and projective ternary relation questions such as Question C1, C2 in Figure 1.1e, 1.1f by searching the answers from a text corpus (e.g., websites) instead of computing answers based on geometries. Sometimes text-corpus-based QA can work (Question B2, C2) if relevant information happens to exist in the corpus, but many times it fails (Question B1, C1). As for those binary spatial relation-based questions such as *which city/county/state is in the north/south/east/west of X*, one cannot pre-compute all possible pairs of places for their cardinal direction relations since this leads to a combinatorial explosion. The situation gets even worse when we consider projective ternary spatial relations (e.g., betweenness) or n-ary spatial relations (e.g., surrounded by).

5. *It is difficult to identify the correct spatial relations given the large spatial language variability.* This can be clearly seen in Figure 1.1c in which "north of California" is misinterpreted as "Northern California" which in turn causes the QA failure. In fact, the difficulty of recognizing spatial relations from natural language sentences has attracted a lot of attention from the NLP and machine learning community [27], especially in the domain of visual question answering [28]. Many papers are focusing

[2] Note that we assume all geometries in the underlying geospatial knowledge base of a GeoQA system share the same coordinate system such as WGS84. This is a common practice used by many geographic knowledge graphs and geospatial ontologies such as GeoSPARQL. If two geographic entities have different coordinate systems, we need to do coordinate system transformation before the QA process.

on recognizing spatial relations which are viewpoint dependent [29] such as *on the left of this door*, *on the right of this building*, *behind this desk.* As for topological and cardinal direction relations, researchers still rely on rule-based methods [30, 31].

6. *Many spatial relations are conceptually vague and therefore difficult to represent computationally in structures like knowledge graphs and difficult to learn.* A typical example of vague spatial relations is *near* [32, 33]. The search radius for the nearby geographic entities varies according to the map scale of the center entity. For example, Question *Find restaurants near Marriott hotel* should use a smaller radius than Question *Find small towns near London.* Another example of vaguely defined spatial relations are cardinal directions (e.g., Question B1, B2) and ternary relations (e.g., Question C1, C2) between/among polygonal geographic entities. Is Nevada in the east or northeast of California? Moreover, the computation of cardinal directions between polygons is complex. Regalia et al. [34] proposed a grid-point-based method which has $O(n^2)$ complexity [3]. As for Question B1 and B2 which search for all states north of California, this computation becomes prohibitively complex. Moreover, we cannot materialize all these cardinal direction relations in a KG beforehand either since this leads to a combinatorial explosion as we discussed above. Similarly, the betweenness relation among geographic entities is also vague and has high computation complexity.

7. *There is a spurious program issue mentioned by Liang et al. [12].* A *spurious program* is a program produced by a semantic parser which accidentally produces the correct answer but with the wrong QA logic, and thus does not generalize to other questions. For example, when we ask for *PlaceOfBirth* of a person, a spurious program may instead ask for *PlaceOfDeath* while these two places are the same for

[3] n is the number of grid points in each polygon

this person. Although a correct QA logic is vital, this kind of QA logic errors is hard to detect by the current standard QA evaluation protocol which is only based on answer comparison. In a weak supervision setting as Liang et al. [12] did, it is hard to distinguish *spurious programs* from the correct program since the only QA annotations are the answers. Similarly, to improve the generalizability of a GeoQA system, it requires not only the correct answer but also the correct computational logic/spatial logic. For example, although Google QA correctly answers Question C2 shown in Figure 1.1f, the answer "Germany" is extracted from a web page about the political and social cooperation of France, Poland and Germany, not a web page about the spatial configuration among these countries. Thus the logic used to answer this question is wrong and slightly changing the question may break the QA process. In other words, the generalizability of this QA model is low. The same issue exists in Question B2 as shown in Figure 1.1d. Although the correct answer "Oregon" is highlighted in the text snippet, several other incorrect answers are also highlighted such as "Nevada" and "Arizona", which also indicates an incorrect QA logic. How to overcome the QA logic error and let the model really understand questions are interesting research directions for GeoQA and QA in general.

1.2.1 Uncertainty and Vagueness of Geographic Information

One may further ask whether the problems shown in Figure 1.1 would be alleviated if we had a GeoQA system which can successfully recognize the correct and efficient spatial relation/operator as well as the correct geographic entities and use their polygon geometries (if necessary) to compute the answer. The answer is still no because of the uncertainty of geometries [35] and the vagueness of geographic concepts/entities [36] which usually exists in real-world geographic datasets.

Geometric Uncertainty

Geometric uncertainty refers to the fact that the precise geometry of one geographic entity may vary according to the map scale, the data source, and map digitization process. According to the famous *coastline paradox*[4], *the coastline of a landmass does not have a well-defined length.* Uncertainty of geometries is in fact caused by the coastal paradox. Because of the uncertainty, sometimes we cannot get the correct spatial relationships between/among geographic entities based on their (polygon) geometries which might be derived from one or several geographic datasets such as OpenStreetMap.

Figure 1.2 uses three examples from OpenStreetMap to show the problem of geometric uncertainty. Each of these three examples consists of a pair of geographic entities who are represented by a red polygon and a blue polygon. By using the Region Connection Calculus 8 (RCC8) [37], the expected spatial relations between these three pairs are *equal* (OE), *tangential proper part* (TPP), and *externally connected* (EC) respectively. However, because of the geometric uncertainty, if we compute their spatial relations based on their polygonal geometries, in all these three examples, their spatial relations become *partially overlapping* (PO). As shown in those zoom-in windows in Figure 1.2b and 1.2c, these unwanted small polygons which break the topological relations between regions are also called "sliver polygon"[5]. For example, in Figure 1.2b, Powellton, West Virginia (the red polygon) should be a subdivision of Fayette County, West Virginia (the blue polygon). However, because of the small sliver polygon shown in the enlarged window, their relations become *partially overlapping* (PO) if we strictly compute the spatial relation based on their geometries and without pre-processing, e.g., by using GeoSPARQL spatial relation functions [38].

Regalia et al. [39] also recognized the effect of geometry uncertainty on the spa-

[4] https://en.wikipedia.org/wiki/Coastline_paradox

[5] https://en.wikipedia.org/wiki/Sliver_polygon

tial relationship computation. To overcome this problem, Regalia et al. [39] proposed to precompute metrically-refined topological relations [40] between geographic entities and materialize them as triples in a geographic knowledge graph. So a GeoQA system only needs to do triple lookup for question answering instead of computing topological relations on-the-fly. However, except for the problem of a substantial larger triple set, how to decide thresholds for metrically-refined topological relation computation is still a big question since these thresholds vary according to the geographic feature types under consideration and the map scale of these geometries.

Vagueness of Geographic Concepts and Entities

However, even if we can fix the problem of geometric uncertainty, a GeoQA system can still fail to answer many geographic questions because of the inherent vagueness of many geographic concepts such as forest, lake, desert, swamp [36, 41], or even coastline. For instance, aside from the geometric uncertainty when digitizing the coastline of Great Britain, the concept "coastline" is conceptually vague. The exact coastline of Great Britain varies according to the time of the day and the season when we measure it. The spatial extent of Amazon forest really depends on the definition of "forest" and can be potentially controversial. Bennett et al. [36] has listed 12 main aspects of vagueness associated with the term "forest" such as *How dense must the vegetation be* and *How large an area must a forest occupy*. Given the vagueness of geographic concepts, it is particularly challenging to pick a correct spatial representation for a geographic entity associated with these concepts. So answering geographic questions that involve these concepts is prone to errors, such as *How many lakes there are in Michigan*, *What is the total area of Amazon forest*, *How far it is from Rocky Mountain to Denver*, and so on.

Interestingly, the vagueness of a geographic entity can not only come from its vaguely defined geographic feature types/concepts, but also come from its own definition such as vague cognitive regions [42]. Good examples are Downtown Santa Barbara [43] and Northern California [42, 44]. It is hard to represent their spatial footprints as polygons with crisp boundaries. Instead, they are usually represented by fuzzy boundaries or membership scores. Answering geographic questions involving these kind of entities is also challenging, i.e., *Is San Luis Obispo part of Southern California?*

.

1.3 Uniqueness of Geographic Questions and GeoQA

Based on the above discussions, the key challenges of GeoQA are summarized as follows. Some general challenges are shared with other QA systems:

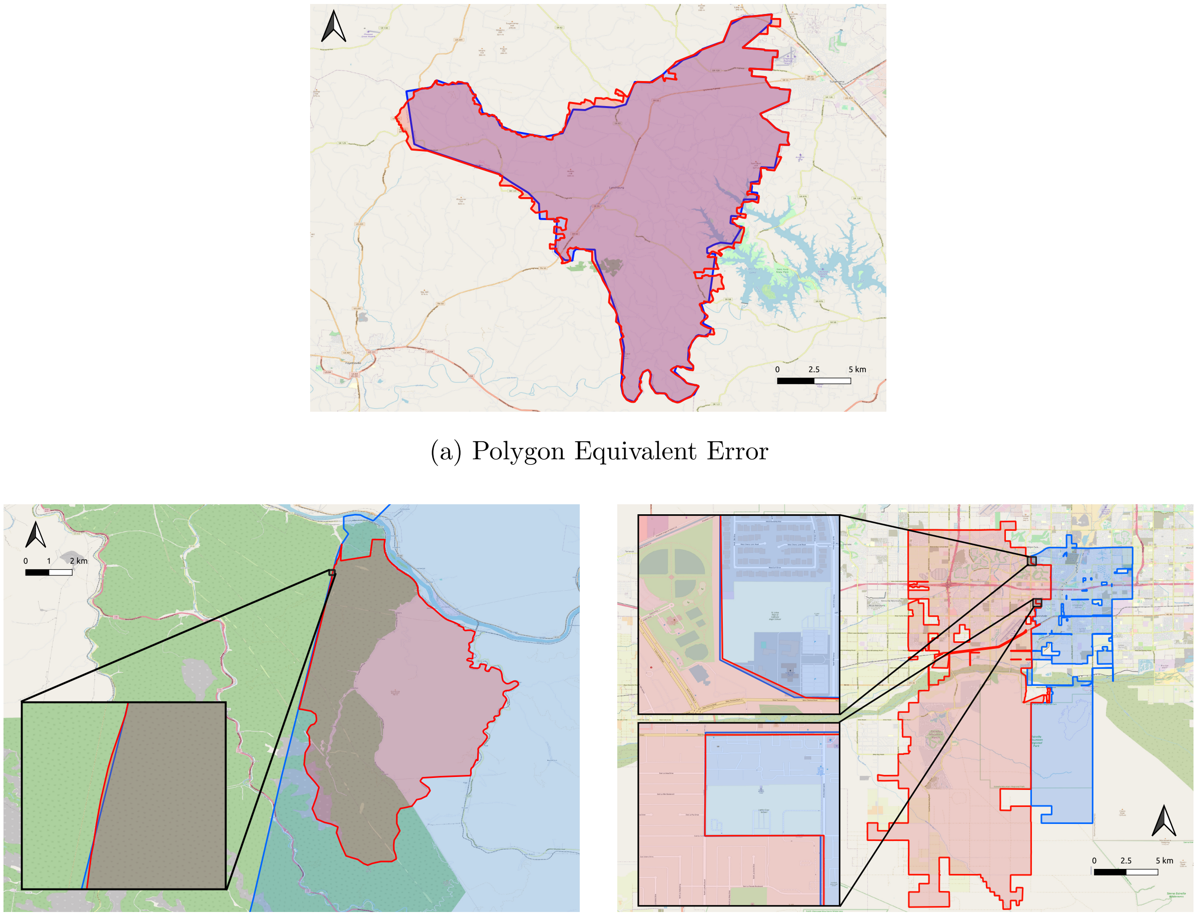
1. **Linguistic variability**: the same question can be expressed in different ways. Paraphrase, hyponym, and synonymy cause a large linguistic variability of (geographic) questions [13].

2. **Program variability**: there are many possible *programs*[6] [12] to answer a given (geographic) questions and each of them are correct. This increases the search space and makes a QA model difficult to train.

3. **Question complexity**: there are various types of geographic questions [31, 46]. Different question types require different data sources and QA techniques to represent the answer. In the first step, it is better to narrow down the scope of the QA systems, i.e., the types of questions the QA system can handle.

4. **Data source diversity**: there are various data sources which can be used as knowledge bases for QA such as knowledge graphs, semi-structured tables, text

[6] In semantic parsing and structured data source QA research [9, 12], programs indicate queries such as SPARQL queries, SQL queries, and λ-calculus [45] which are *translated* from natural language questions and can be executed on the underlining knowledge base to retrieve the answer.



(a) Polygon Equivalent Error

(b) Polygon Containment Error

(c) Polygon Touch Error

Figure 1.2: Three examples to show the geometry uncertainty problem in OpenStreetMap: (a) Lynchburg, Tennessee (the red polygon) is a consolidated city-county whose boundaries are identical to Moore County, Tennessee (the blue polygon). However, the answer to Question *Is Lynchburg, Tennessee equivalent to Moore County, Tennessee* is No, if we compute the spatial relation between these two polygon geometries based on GeoSPARQL function *geof:sfEquals*. (b) Powellton, West Virginia (the red polygon) is a census-designated place inside of Fayette County, West Virginia (the blue polygon). However, for Question *is Powellton, West Virginia inside of Fayette County, West Virginia*, the answer is No if we use GeoSPARQL function *geof:sfWithin* to compute the spatial relation between their polygon geometries. (c) Avondale, Arizona (the blue polygon) is a nearby city of Goodyear, Arizona (the red polygon). However, As for Question *Does Avondale, Arizona touch Goodyear, Arizona* or Question Is Avondale, Arizona externally connected to Goodyear, Arizona, if we compute the answer based on GeoSPARQL function *geof:sfTouches*, their answers are both No because their OpenStreetMap polygons intersect with each other.

corpus. Sometimes it is necessary to answer questions based on multiple data sources. It becomes more demanding in the GeoQA context since most geographic questions have to be answered based on a combination of multiple data sources such as raster data, vector data, text corpus, geographic knowledge graphs, and so on. Hence, developing QA systems based on multiple data sources is particularly challenging.

There are unique challenges which are specific for geographic question answering. Based on Section 1.2 and Mai et al. [47], these unique challenges can be summarized as follows:

1. Answering geographic questions relies on **appropriate spatial information** such as geometries (e.g., points, polylines, and polygons). Inappropriate selection of spatial footprints will lead to wrong answers as shown in Figure 1.1a and 1.1b.

2. A GeoQA system should be robust in handling **the vagueness and uncertainty of geographic information**. For example, a lake can have different definitions and different polygonal representations at different map scales. These uncertainties and vagueness might change the spatial relations between these polygon geometries as shown in Figure 1.2 and discussed in Section 1.2.1. A GeoQA system should be able to handle this.

3. Answers to many geographic questions are best derived from **a sequence of spatial operations** such as proximity (Figure 1.1a, Figure 1.1b), topological and cardinal direction (Figure 1.1c, Figure 1.1d), and routing computation rather than being directly extracted from a piece of unstructured text [18] or retrieved from Knowledge Graphs (KG) [13], which are the normal procedures in current QA systems.

4. Compared with the general QA, answering geographic questions requires **a substantially larger set of programs/operators**, especially a large set of spatial operators. This increases the program search space exponentially. For example, PostGIS has 21 spatial relationship functions (e.g., ST_Within), 27 measurement functions (e.g., ST_Azimuth), and 25 geometry processing functions (e.g., ST_Buffer) [7]. In contrast, in the general QA research, the current semantic parser [11, 12] or reading comprehension QA [15] usually only utilize a small set of operators to make the whole model trainable. For instance, Neural Symbolic Machine (NSM) [12], as a neural sequence-to-sequence semantic parser, automatically translate a question into a program that can be executed on the KG and retrieve answers with the support of a Lisp interpreter. This Lisp interpreter only supports 4 operators - *Hop*, *ArgMax*, *ArgMin*, and *Filter*. Neural Symbolic Reader (NeRd) [15], as a scalable reading comprehension QA, only supports 11 different operators. The total number of possible programs that can be generated grows exponentially with respect to the number of operators we consider. So the large number of spatial operators makes this program generation task extremely complex.

5. Geographic question answering can be **subjective and context dependent**, i.e., depending on when and where this question is asked, who ask it, and what this question is asked about. Some examples are *Is California (the territory) part of the United States* (time-dependent), *which country contains the largest proportion of the Kashmir region* (location-dependent and subject-dependent). The answer to the first question can be USA or Mexico depending on the temporal scope of this question. The answer to the second question can be India or Pakistan depending on when, where, and who you ask this question [26].

[7] https://postgis.net/docs/reference.html

6. Geographic questions can be **vague in terms of the involved spatial relations and geographic concepts**. For instance, the answer to Question *In what direction is France located to Italy* can be either east or southeast depending on the definition of cardinal directions between polygons. Moreover, for Question *What is the total area of forest in Brazil*, the answer depends on the definition of forest [41].

1.4 Existing Work on GeoQA

Although QA has been a long-standing research topic, geographic question answering (GeoQA) remains less studied. In this section, we discuss some important existing work on GeoQA. Based on the types of geographic questions they focus on, we classify existing GeoQA research into four types: factoid, geo-analytical, scenario-based, and visual.

1.4.1 Factoid Geographic Question Answering

Factoid GeoQA focuses on answering questions based on geographic facts. To the best of our knowledge, Zelle et al. [10] presented the first GeoQA system, which uses CHILL parser to answer natural language geographic questions based on the *Geoquery* query language. They defined 20 relations such as *capital*, *area*, *next_to*, *traverse*, and so on, which indicate different types of geographic questions that Geoquery supports. Although some relations are spatial such as *next_to* and *traverse*, all relations have been materialized as 800 Prolog facts. Then the QA system only needs to perform a question-query translation and an answer lookup. Namely, no on-the-fly spatial computation is required. Although this work focused on answering geographic questions, a standard QA pipeline was adopted and the uniqueness of geographic questions was not considered.

Chen et al. [48] proposed a geographic question answering framework to answer five types of geographic questions based on the spatial operators supported by PostGIS. An input geographic question first goes through a linguistic analysis so as to be classified into one of the predefined query templates. Then the spatial SQL query template is filled by using the parsed data such as spatial operators (e.g., ST_Within, ST_Buffer), place name, quantity constraints, and so on. Subsequently, the answer is retrieved by executing this query on the underlining PostGIS database. This GeoQA framework can support five simple geographic question types: 1) location questions, e.g., *where is Columbus*; 2) direction & distance questions, e.g., *where is Columbus perspective to Cleveland*; 3) distance questions, *how far is it from Columbus to Cleveland*; 4) nearest questions, e.g., *which city is the nearest to Columbus*; 5) buffer questions, e.g., *which cities are within 5 miles from Columbus.* We can see that except for the first type of questions, the rests require spatial operators. Compared with Zelle et al. [10] who materialized all spatial relations as facts beforehand, this system is able to utilize spatial operators to answer geographic questions on-the-fly. However, it simply utilizes points to represent geographic entities and thus inherits the limitation we have discussed in Section 1.2. The limited number of question types and the small size of the underlying database restrict the number of geographic questions it can handle.

Punjani et al. [31] proposed a template-based GeoQA system as Chen et al. [48] did. Instead of relying on a PostGIS database, this GeoQA system is based on a GeoSPARQL-enabled geographic knowledge graph created from DBpedia, GADM database of global administrative areas, and OpenStreetMap. This GeoQA system mainly focuses on seven types of factoid geographic questions which can be answered based on several hand-crafted GeoSPARQL query templates. These question types include various numbers of geographic entities, concepts, or spatial relations. First, geographic entities, concepts, and spatial relations are extracted from a natural language geographic question asked by users. Then this question is mapped to one of the query templates. The generated GeoSPARQL query is then executed on the underlining KG to obtain answers. This

GeoQA system is able to handle different spatial relations such as topological relations and cardinal direction relations by using the polygon geometries of each geographic entity. However, the deterministic spatial operations supported by GeoSPARQL suffer from uncertainty of the polygon geometries we have discussed in Section 1.2.1.

As a prerequisite of GeoQA, Hamzei et al. [46] carried out a data-driven place-based question analysis using a large-scale QA dataset generated from Microsoft Bing - MS MARCO V2.1. They used linguistic analysis to translate questions and answers into their semantic encodings based on six primary elements: place names, place types, activities (e.g., buy), situations (e.g., live), qualitative spatial relationships, and qualities. Then they used a string similarity measure (Jaro similarity) as well as k-means to cluster the encoded questions and answers into different clusters. Experimental results showed that place-based questions can be clustered into three types: 1) non-spatial questions - questions not aiming at localization of places (e.g., *In which county is Grand Forks, North Dakota located*); 2) spatial questions - questions about locations of place (e.g., *where is Barton County, Kansas*); 3) non-geographical and ambiguous questions (e.g., *where are ores located*). The proposed semantic encoding approach benefits our understanding of the intent of geographic questions. However, this classification is rather coarse. The non-spatial question type still contains various types of factoid geographic question. Moreover, this classification is still based on the syntactic structures of questions rather than their semantic interpretations. The geographic question types discussed in Hamzei et al. [46] are only factoid questions. In contrast, we provide a classification of geographic questions in Section 1.5 based on their semantic interpretations which cover a wider range of question types.

Based on the above discussion, we can see that although there is some research on factoid GeoQA, most existing GeoQA models [10, 48, 30, 31] are template-based and can only handle limited types of geographic questions. Commonly, they adopted a two-step strategy to answer geographic questions – a question classification step and an answering step. A natural language question is first classified into one predefined query template, which then is used in QA system to seek the answer. This indicates that, these models are not directly trained on the labeled data, namely question-answer pairs. Instead, they are usually trained on the intermediate question type labels which does not guarantee for a correct final answer while this error cannot propagate back to the whole QA framework. Therefore, existing GeoQA models can hardly be trained in an end-to-end manner as many reading comprehension QA models do [12, 18] and cannot be easily generalized to other datasets as well. In short, there is still a lack of efficient large-scale end-to-end GeoQA systems which can handle various types of geographic questions.

1.4.2 Geo-analytical Question Answering

Compared with the above GeoQA work that mainly focus on answering factoid geographic questions, geo-analytical question answering proposed by Scheider et al. [23] went beyond simple geographic facts but focuses more on questions with complex spatial analytical intents [49]. A simple factoid geographic question such as Question A1 can be answered by executing one or two spatial operations on the respective spatial footprints of geographic entities. In contrast, geo-analytical questions usually require generating a GIS analytic workflows. Example questions include *how much green space will Tom see while running through Amsterdam* (Question M) Scheider et al. [23] and *what is the best site for a new landfill in the Netherlands* (Question N) [49].

The aim of geo-analytical question answering also shifts from retrieving simple answers to formulating the answer through analytical workflows which might be generated on-the-fly or retrieved from a GIS workflow corpus shared by other GIS users [50].

Despite the interesting nature of geo-analytical QA, several challenges need to be solved in order to develop a full-functional geo-analytical QA system. Firstly, in contrast to all current QA systems which are built on *predefined knowledge bases* (e.g., knowledge graphs, text corpus, and semi-structured tables), geo-analytical question answering does not have well-defined knowledge bases. Different geo-analytical questions might require different kinds of knowledge bases. Scheider et al. [23] turned to treat a portal of different GIS datasets as the knowledge base of geo-analytical QA. However, all current Geoportals such as ArcGIS Online [51, 52] and NASA Physical Oceanography Distributed Active Archive Center (PO.DAAC) [53] only support search functionality over different datasets on the metadata level and cannot be directly used for geo-analytical QA which requires a deep assessment of the analytic potential of a GIS dataset for a given question. Secondly, geo-analytical questions are mostly vaguely defined and can be answered based on different combinations of data sets and GIS tools (spatial operators). For example, as shown in Scheider et al. [23], to answer Question M, one option is to use a vector map of urban trees in Amsterdam overlaid on Tom's running trajectory, based on which the number of trees within the buffer of the trajectory can be computed to answer the question. Another option is to use a raster map of green space in Amsterdam and computing the answer based on kernel density estimation and map algebra. Different data set options make it difficult to design a knowledge base for geo-analytical QA. Different possible solutions lead to a growing solution space and therefore make it harder to construct a fully automatic QA pipeline. It is these difficulties that make geo-analytical QA challenging and worth investigating at the same time.

1.4.3 Scenario-based Geographic Question Answering

In scenario-based GeoQA (GeoSQA), a question is always associated with a scenario described by a map or a paragraph. Huang et al. [54] presented a GeoSQA dataset which consists of 1,981 scenarios and 4,110 multiple choice questions in geography domain. These scenarios and multiple choice questions are collected from Gaokao, China's version of SAT, and mock tests at high school level. So all scenario-based geographic questions are textbook-like questions. An example scenario can be a map showing the urban planning of a city as well as its textual description. The associated question asks for the possible usage of a location presented on the map. Answering this kind of questions requires some commonsense knowledge in geography as well as a deep understanding of the scenario. Huang et al. [54] showed that the state-of-the-art reading comprehension and textual entailment models perform no better than random guess on this task which illustrates the challenges of this kind of GeoQA.

In contrast to the textbook-like scenario-based QA, Contractor et al. [55] presented a tourism oriented scenario QA task and a GeoQA pipeline. The target QA dataset - Tourism Questions [56] consists of over 47,000 real-world tourism questions that seek for Points-of-Interest (POI) recommendations together with a universe of nearly 200,000 candidate POIs. These questions are long paragraphs which describe a tourism scenario asking for POI recommendation. An example question is *I am outside of Universal Studio, Los Angels, please recommend good Chinese restaurants nearby*[8]. The answer to these questions are usually a ranked list of POIs. To tackle this task, Contractor et al. [55] proposed a spatio-textual reasoning network which jointly considers the spatial proximity between candidate POIs and the target POIs in the question as well as the semantic similarity between questions and the reviews of candidate POIs. The distances between candidate POIs and the target POIs mentioned in the question are explicitly encoded by a geo-spatial reasoner module which produces the spatial relevant scores between questions and candidate POIs. The semantic relevant scores are computed by

[8] Since the original question example is very long. We formulate a rather short and simple tourism scenario question here.

a textual reasoning sub-network. These two scores are then combined to produce the final relevant scores between questions and each candidate POI. The approach indeed shows a great potential of spatial reasoning in GeoQA. However, since distances need to be computed for each pair of candidate POIs and target POIs in the questions, the presented spatio-textual reasoning network is not suitable for open-domain QA where we can have a richer pool of candidate POIs to search from.

1.4.4 Visual Geographic Question Answering

Visual question answering [28] is another rapidly developing QA research direction in which each question is paired with an image as the context. An example question that could be asked about an image showing a child is *where is the child sitting*. Lobry et al. [57] adopted this idea and proposed the task of visual question answering for remote sensing data (RSVQA) in which a remote sensing image is paired with a question asking about the content of this image. Example questions include *how many buildings are there*, and *what is the area covered by small buildings*. To answer this kind of questions, Lobry et al. [57] utilized a Convolutional Neural Networks (CNN) as the image encoder and a Recurrent Neural Network (RNN) as the question encoder. The encoder outputs are concatenated and fed to a fully connected layer which is followed by an answer classification layer. Although this work mainly focuses on capturing computer vision features, spatial knowledge is minimally utilized in the RSVQA model design. Consequently, the presented RSVQA model shows little difference compared to normal VQA models. How to incorporate spatial thinking into the RSVQA model design to develop spatially-explicit [58] QA models is a promising future research direction.

1.5 The Classification of Geographic Questions

Section 1.4 discussed key work on GeoQA which focus on certain types of geographic questions. Some of them [48, 31, 46, 49] provided a classification of geographic questions within the scope of question types they can handle. In this section, we provide a general classification of geographic questions which attempts to cover all aspects of GeoQA. We hope this classification can comprehensively reveal the landscape of GeoQA and serve as a guideline for future GeoQA-related research.

In fact, Mishra et al. [2] provided a survey for question answering systems and classified QA systems based on multiple criteria including application domains, question types, types of analysis on questions, types of data sources, retrieval methods, and answer types. According to Mishra et al. [2], questions can be classified as factoid type questions [what, when, which, who, how], list type questions, hypothetical type questions, causal questions [how and why], and confirmation questions. Although the classification covers most of the questions asked in a normal QA system, it does not consider many important types that we often see in geographic questions such as questions about spatial relations, routing questions, prediction-based questions, and so on.

Following classification by Mishra et al. [2], we classify geographic questions into the following categories:

1. **Factoid geographic questions**: geographic questions that can be answered based on the factoid geographic knowledge, e.g., *which state is Houston located in.*

2. **Prediction-based geographic questions**: geographic questions should be answered based on the prediction of facts, e.g., *what will be the average temperature in Las Vegas next Monday.*

3. **Opinion geographic questions**: geographic questions which require subjective

information or opinions about some geographic facts, e.g., *what is the best trail in the Grand Canyon National Park.*

4. **Hypothetical geographic questions**: geographic questions that ask for information related to any hypothetical events, e.g., *what would California look like if the United States had not acquired it in 1848.*

5. **Causal geographic questions**: geographic questions which require explanations about geographic facts, e.g., *why and how did Los Angeles become famous for its film industry.*

6. **Geo-analytical questions**: geographic questions which require complicated geoprocessing workflows to answer, e.g., *where is the best location for my new house in San Diego with a quiet neighborhood, lower crime rate, good accessibility to grocery stores and beach.*

7. **Scenario-based geographic questions**: geographic questions that are associated with a scenario described by textual description or a map. An example question is *we just arrived at London and currently stay at a hotel close to London King's Cross train station. Can you recommend a good Italian restaurant nearby which serves vegan pizza?*

8. **Visual geographic questions**: geographic questions paired with remote sensing images or maps whose contents are the focus of these questions.

In the following, we will discuss each question type in detail.

1.5.1 Factoid Geographic Questions

In contrast to the factoid type questions defined by Mishra et al. [2] that require answers in a single short phrase or sentence and whose expected answer types are named entities, we define factoid geographic questions in a broader sense in terms of the answer types. Any questions that can be answered based on the real-world factoid geographic knowledge can be treated as factoid geographic questions. The factoid type questions and list type questions[9] [2] are included in this question type if they are geographic questions.

Factoid geographic questions are the most typical question type that existing GeoQA systems focus on. We further classify this type into the following sub-types:

1. **Single geographic entity attribute questions**: This type refers to questions about attributes of one single geographic entity such as its geographic coordinates, population, elevation, area, temperature, and so on. This question type does not require any spatial operations and thus can be answered via a datatype property[10] triple fetched from a GeoKG or extracted from a description of a place. Examples include *where is London*, *what is the total population of Phoenix, Arizona*, and *what is the annual precipitation in Seattle, Washington.*

2. **Spatial relationship questions**: These are questions that involve spatial relations such as spatial proximity, topological relations, cardinal directions, ternary projective relation, and n-ary spatial relations between/among (two or more) geographic entities. Examples of this type include: *how far is it from New York to Washington D.C.* (spatial proximity), *how much does it cost to take a Uber from Stanford University to Pier 33* (time dependent spatial proximity), *does King Canyon National Park touch Inyo County, California* (topological relations), *What is the cardinal direction between Los Angles and San Diego* (cardinal directions), *which country sits between China and Russian* (ternary projective relation), and *which countries surround Switzerland* (n-ary spatial relation).

[9] list type questions are questions whose answer are a list of entities. This question type is still based on factoid knowledge.

[10] `https://www.w3.org/TR/owl-ref/#DatatypeProperty-def`

3. **Spatial/non-spatial qualifier questions**: This refers to those questions that are asked about one or a set of geographic entities which satisfy one or several spatial (e.g., in City A) or non-spatial (e.g., highest elevation) qualifiers. Examples include: *What is the largest city in United States in terms of population*? *Which province in China has the highest average elevation*? *Which coastal cities are within 20 miles from Seattle*? *Which churches are near a castle in Scotland*, and *Which city in France has the largest COVID-19 case count.*

4. **Routing questions**: This type of questions is frequently asked in navigation guidance services and mainly asks about the routing between places. The answer is, therefore, a route displayed on the map or a voice/text-based step-by-step instruction. One example is: *how to get from Hollywood to LAX airport?*

These sub-types comprehensively cover geographic question types that have been discussed in Zelle et al. [10], Chen et al. [48], Chen et al. Chen et al. [30], Punjani et al. [31], Hamzei et al. [46], and Xu et al. [49].

1.5.2 Prediction-based Geographic Questions

Factoid geographic questions ask about historical or present geographic knowledge, while prediction-based geographic questions ask about the future. Hence answers should be generated based on predictions of real-world geographic facts such as population, temperature, future events, and so on.

In some cases, the predictions have been precomputed and stored in a knowledge base. Then the QA process of prediction-based geographic questions can be done in exactly the same way as that of factoid geographic questions. We classify prediction-based geographic questions as follow:

1. **Single geographic entity attribute prediction question**: Questions about the prediction of attributes of one single geographic entity such as population, air quality, temperature, and so on. e.g., *what will be the air quality like in Los Angels in the following two weeks*, *where this iceberg will be in two months after its recently separation from the Antarctic glacier*.

2. **Spatial/non-spatial qualifier prediction questions**: Questions asked about one or a set of geographic entities which satisfy one or several spatial or non-spatial qualifiers in the future.

   (a) **Prediction-based non-spatial qualifier questions**: These questions have non-spatial qualifiers which are based on the predictions of the attributes of geographic entities. Examples are *which country in the world will have the largest population in 10 years*, *which state in the US will have the largest total COVID-19 case count once this current pandemic ends*, *which university in Australia will have the largest proportion of international students in 5 years*.

   (b) **Prediction-based spatial qualifier questions**: These prediction questions have spatial qualifiers for geographic entities whose locations may or may not change, e.g., *which nearby house will have the largest increase in its price after the construction of this subway station that will be finished in two years*.

If the predictions are not available beforehand, the GeoQA system should be able to understand the question intent and generate a program to compute the answer which might involve some prediction functions. As far as we know, there are no QA systems available to date that address this type of GeoQA.

1.5.3 Opinion Geographic Questions

Opinion geographic questions involve personal opinions with subjective terms such as the *best* hotels, the *most beautiful* city, the *most atmospheric* restaurant, and so on. These subjective terms can be interpreted in different ways by different people which complicates the question answering process. For example, as for Question *what is the largest city in Texas*, "the largest city" can be interpreted as the city with the largest population, or with the largest area. Some subjective terms can be approximated based on existing quantitative measures. For example, as for Question *what is the most popular restaurant in San Jose, California*, we can use the Yelp rating as a proxy to measure the popularity of a restaurant. In this case, the QA can be done in the same way as that of the factoid geographic questions. Nevertheless, opinion detection itself which classifies text as subjective or objective is still a research problem [59].

1.5.4 Hypothetical Geographic Questions

Similar to the definition provided by Mishra et al. [2], hypothetical geographic questions ask for information related to any hypothetical event or condition. The question is usually formulated as "what would happen if...". Example questions are *what would California look like if the United States had not acquired it in 1848*, *which nearby cities would have been flooded if the dike at Huayuankou, Henan would have been breached again*[11].

At first glance, hypothetical geographic questions might look similar to prediction-based geographic questions. However, they are different question types. The former asks for a hypothetical situation and the answers are usually derived from an educated guess based on commonsense. In contrast, the later asks for a scientific prediction based on the observation data.

[11] https://en.wikipedia.org/wiki/Huayuankou,_Henan

Since there are no 100% correct answer for these questions, the QA reliability is low and the QA technique adopted by factoid question answering will not work. Some expert knowledge and commonsense knowledge may need to be involved during the QA process. This question type might be one of the most difficult one to handle and need to be investigated further.

1.5.5 Causal Geographic Questions

Causal geographic questions ask for explanations about geographic facts. Example questions are *why are there a lot of places along the west coast of the Atlantic Ocean named after Alexander von Humboldt* and *why are there a lot of places in South America named San Jose*.

The answer to a causal geographic question is usually a passage about the geographic facts under discussion. So we can adopt some text-corpus-based extractive question answering techniques [7, 19] to "approximately" answer this kind of questions. However, almost all the current deep neural network based extractive QA models [7] [12] can only do fact lookup from text corpus while causal geographic questions require a deep understanding of the causality relationship in the questions and reasoning on commonsense knowledge. So simply applying extractive QA models on causal questions will lead to much lower performance.

1.5.6 Geo-analytical Questions

Section 1.4 provides a detailed description of geo-analytical QA and discusses about the challenges we might meet when developing a geo-analytical QA system - uncertain choices of knowledge bases and exploded solution space.

[12] Given a question, an extractive QA model search for the possible paragraphs which might contain the answer. And then it reads these paragraph sand *extract* text spans from them as the answers.

The reasons why we separate geo-analytical questions from other types of questions are two-fold: 1) unlike other types of questions that aim at generating compact answers, geo-analytical question answering focuses more on generating or retrieving the geoprcessing workflows [23] that can be used to obtain answers; 2) in contrast to other question types that have relatively limited answer types, the answer types of geo-analytical questions are very diverse. Example answer types include raster maps, geometries, numerical values, geographic entities, text, and so on.

Despite its difficulty, geo-analytical QA actually points out an exciting future direction of GIS technology which can automate the spatial analysis process without any human intervention. So we still advocate this idea and expect a major advancement along this research direction in the early future.

1.5.7 Scenario-based Geographic Questions

As we discussed in Section 1.4.3, a scenario-based geographic question is usually associated with a scenario depicted by either a map or a textual description. Classical scenarios used in GeoQA include the textbook-like scenario such as the GeoSQA dataset [54] and the tourism scenario such as Tourism dataset [56, 55]. As for Tourism datasets [56], only simple spatial reasoning, e.g., distance between candidate POIs and POIs mentioned in the scenario, is required. However, as for the GeoSQA dataset [54], different textbook scenarios require different spatial reasoning such as cardinal directions, proximity, and topological reasoning. Moreover, commonsense knowledge is required to correctly answer this type of questions. Therefore, designing a spatial-aware QA model for GeoSQA is challenging.

So for scenario-based geographic questions, the design of GeoQA model varies from case to case and depends on the nature of the questions and what scenario the questions are based on.

1.5.8 Visual Geographic Questions

Visual geographic questions are different from other question types because each question is paired with a remote sensing image [57] or a map. These images or maps can be seen as the restricted knowledge base for corresponding questions. The map can be a historic map or a narrative map. They can also be obtained from some fictions, such as Marauder's Map from Harry Potter, *Atlas of the European novel, 1800-1900* [60], and *A Literary Atlas of Europe*[13]. However, to the best of our knowledge, these narrative maps have not been used for the GeoQA purpose and there is no visual GeoQA work focusing on fictional maps.

Promising research questions for visual geographic questions answering include issues such as what makes visual GeoQA different from normal visual QA? What are the benefits to incorporate spatial knowledge into Visual GeoQA models? One possible direction lies in the difference between the spatial relations used in general VQA and geographic VQA. The spatial relations studied in the current VQA [29] are like *on the left of*, *in front of*, and *on top of* which is very different from the spatial relations we would have among geographic entities, e.g. cardinal direction, topological relations. Whether this difference leads to some difference in the GeoQA model design needs to be investigate further.

1.5.9 Discussion about the Question Classification

The proposed question classification is an integration and extension of multiple existing question classification work [2, 31, 46]. In fact, these question types are classified from different aspects: factoid vs. non-factoid questions, objective vs. subjective/opinion

[13] `http://www.literaturatlas.eu/en/`

questions, geo-analytical vs. knowledge lookup questions, textual vs. visual questions, and so on. More specifically, the first five question types are classified based on the types of knowledge that a question focuses on - factoid knowledge, the knowledge about future, the knowledge about people's opinions, common sense knowledge about hypothetical events, or knowledge about the explanations for geographic facts. Geo-analytical questions are listed as one specific type because of its specific focus on GIS workflow synthesis. The scenario-based and visual geographic question types emphasize the context (e.g., text description, images) associated with the question. Basically, these question types reflect different aspects and focuses of GeoQA.

These question types are not necessarily mutually exclusive from each other. For example, as for Question *What would be the best location if we want to build a new elementary school in Seattle*, it is both a hypothetical geographic question and a geo-analytical question because this question follows the "what would happen if..." hypothetical question pattern and answering it requires GIS workflow synthesis (e.g., site selection analysis). Question *How many buildings are in the current remote sensing image* is both a factoid geographic question and a visual geographic question.

Moreover, this question classification only reflects our current understanding of GeoQA research and is by no means a final and complete system for geographic question classification. With the advancement of the GeoQA research, we might see new types of geographic questions which have not been covered by the presented classification system.

Nevertheless, we believe the presented geographic question classification is useful since it can help a GeoQA researcher to narrow down the focus and find an appropriate GeoQA dataset that fits into their research scope. It can also guide them in the process of GeoQA benchmark dataset construction and analysis as Hamzei et al. [46] did. Last but not least , a question classification system helps identify the challenges and future research directions for GeoQA.

1.6 Future Research Directions for GeoQA

In this section, we will discuss some interesting research directions for GeoQA. Most importantly, we need to address the question of what unique contributions we can make in GeoQA beyond work on more general AQ systems.

Question answering is one of the most important research topics in natural language processing. Currently, there are around 30 different large-scale question answering data sets available[14]. Most of them are about reading comprehension and open-domain question answering such as HotpotQA [16], SQuAD [7], Natural Questions [17], CoQA [61] which mainly aim at unstructured-text based QA. There are also QA datasets for structured-knowledge-based QA such as QALD-9 [62].

Compared with QA, GeoQA is a smaller research topic which starts attracting attentions from QA researchers as well as GIScientists only recently. A recent review on the usage of geospatial information in virtual assistants [63] also showed that the usage of different types of geographic data and various spatial methods in virtual assistants is quite limited. *How we can show the unique contribution of GeoQA to the general QA community* is the golden question needed to be answered for GIScientists.

As far as we see, there are some interesting and unique research directions specifically for GeoQA:

1. **How to effectively utilize geographic coordinates in a GeoQA model?** As the basic element of geographic information, how to effectively utilize locations in deep learning models for any geospatial task is a fundamental problem itself. [55] presented an indirect way to encode distances among locations (e.g., POIs) for the

[14] `http://nlpprogress.com/english/question_answering.html`

GeoQA purpose. In contrast, Mac Aodha et al. [64], Mai et al. [65], and Mai et al. [66] take a more explicit approach which directly encode coordinates into location embeddings for multiple downstream tasks. Which one works better for a specific GeoQA task needs to be investigated.

2. **How to effectively utilize complex spatial footprints of geographic entities such as polygons, multipolygons, and polylines in a GeoQA model? How to design efficient "fuzzy spatial operators" which are robust to the geometry uncertainty problem?** These complex spatial footprints are essential for many geographic question types. However, as we discussed in Section 1.2.1, directly utilizing deterministic spatial operators such as GeoSPARQL functions as Punjani et al. [31] did will suffer from the known problems with using raw geometries which will affect the performance of GeoQA. A more proper way is to design an efficient neural-network-based "fuzzy spatial operator" which is robust to the geometric uncertainty problem. This "fuzzy spatial operator" takes these complex polygon geometries as input and outputs their spatial relations. At the training phase, this operator automatically learns the concept of thresholds implicitly based on the training labels and we do not need to specify thresholds explicitly as Regalia et al. [39] did. This might be an interesting research direction.

3. **How to define a compact but effective set of spatial operators for GeoQA? Furthermore, how to define a program language similar to Lisp [12] and Prolog [10] but for spatial computing which will make GeoQA easier?** As we discussed in Section 1.3, given the large number of spatial operators, we need to derive a small subset which can be used to answer most of the geographic question types. The core concepts of spatial information research [67] may be a great starting point since it provides a list of core spatial operators/computations and defines

a high-level language for spatial computing [68]. However, several issues need to be investigated further - How good are these spatial operators? How easily can they be applied to GeoQA? And how many question types can they support?

4. **How to handle the vagueness of spatial relations as well as geographic concepts in a GeoQA model?** The selection of spatial operators should be aware of the vagueness of geographic concepts and geographic entities during question answering process. For example, Question *Is San Luis Obispo part of Southern California* and Question *Is San Luis Obispo part of California* should be handled differently. Unlike California, Southern California is a vague cognitive region which does not have a crisp boundary. The ordinary topological relation operators cannot deal with this. It might be complicated to design a GeoQA model to directly interpret the vagueness of geographic concepts and entities. A simple yet effective approach is to collect annotated data for QA pairs which contain these spatial operators and concepts and develop an end-to-end model to learn from them.

In this paper, we attempt to provide a holistic view of the current landscape of GeoQA research as well as its challenges and uniqueness. We hope the GeoQA problem mentioned by Jordan can be solved and a real geospatial artificial intelligence agent can be built in the coming years.

1.7 Software and Data Availability

The data utilized in this paper are downloaded from OpenStreetMap and visualized using QGIS. All data and software used are open source.

Acknowledgement This work is funded by the NSF award 2033521 A1: *KnowWhereGraph: Enriching and Linking Cross-Domain Knowledge Graphs using Spatially-Explicit*

AI Technologies.

1.8 Research Questions and Dissertation Synopsis

1.8.1 Research Questions

Section 1.6 lists several interesting research directions based on the uniqueness of GeoQA discussed in Section 1.3. Although each of them focuses on different aspects, they jointly ask one question - *how to design spatially-explicit machine learning models [58, 69] for geographic question answering such that it is able to capture the spatial representations of geographic entities (e.g., point coordinates, polylines, and polygons) and the spatial relations among them.* This question is the core question we will address in this dissertation. Because of the complex nature of this question, we divide it into three research questions that deep dive into the problem of GeoQA step by step.

- **RQ1** - Geographic Question Relaxation: *How to develop a spatially-explicit query relaxation and rewriting model which utilizes spatial knowledge to handle those unanswerable geographic questions - geographic questions that are unanswerable for a normal question answering system?*

Although Section 1.3 discusses the uniqueness of geographic questions, it is still unclear whether we can use spatial thinking and spatial knowledge to develop a spatially-explicit QA model that can handle these uniquenesses. An initial and important step is to develop a spatially-explicit query relaxation model that can handle those unanswerable geographic questions. RQ1 focuses on this question to test whether utilizing spatial knowledge can help to answer more geographic questions correctly.

- **RQ2** - Location Encoding: *How to design a general-purpose location encoding model which can encode a (geographic) location into the embedding space such that it can* *be utilized in multiple downstream geospatial tasks? How to integrate this location encoding model into the neural network-based question answering model to make a location-aware QA framework?*

RQ2 is a further question asked based on RQ1. If we can prove that utilizing spatial knowledge is useful for GeoQA, the next step is to find a fundamental way to incorporate spatial information, especially geographic locations, into the current neural network-based QA models.

There are multiple ways to incorporate spatial information or spatial knowledge into deep learning models such as distance-based spatial context resampling [70, 69], distance-based spatial context reordering [71], distance-based triple resampling [72, 73, 47], and distance-based spatial reasoning [55]. Most of these approaches utilize pairwise distances among geographic entities/places to design specialized spatially-explicit models for given tasks.

Instead of following this research direction, we are interested in a more fundamental approach that encodes geographic location directly into the embedding space and utilizes it in multiple downstream tasks. This approach has two advantages: 1) given a dataset with N geographic entities, by encoding geographic locations instead of their pairwise distances, we effectively reduce the problem from $O(N^2)$ to $O(N)$; 2) In contrast to a specialized spatially-explicit model which is designed for a specific task, a general-purpose location encoding model can be utilized in many geospatial tasks including GeoQA.

So RQ2 focuses on developing a general-purpose location encoding model and utilizing it to develop a location-aware QA model.

- **RQ3** - Polygon Encoding: *How to design a general-purpose representation learning model for polygonal geometries, i.e., polygon encoder, for polygon-based geospatial tasks? How to utilize it in a GeoQA model to answer spatial relation questions* *which otherwise cannot be answered only based on point-based spatial footprints?*

RQ2 focuses on representing geographic locations, i.e., points, into the embedding space. However, many geographic questions (i.e., topological relation questions, distance questions, cardinal relation questions) require more detailed spatial representations of geographic entities such as polygons and multipolygons to answer these questions correctly. For example, Question A1 in Figure 1.1a needs polygon representations of China and Russia. Question B1 and B2 in Figure 1.1c and 1.1d also require polygonal geometries of states of the US. Otherwise, based on a point representation of California and Nevada which might be sampled anywhere within their polygon representations, a deterministic cardinal direction operator will indicate Nevada is in the southeast (or even south) of California. However, how to handle geometric uncertainty as well as vagueness of geographic concepts and entities (See Section 1.2.1) when we use the polygonal representation of geographic entities also needs to be solved.

RQ3 focuses on this problem and calls for a general-purpose polygon encoder that can encode polygonal geometries (e.g., simple polygons, polygons with holes, or multipolygons) into the embedding space. This polygon encoder can be utilized to answer those geographic questions that require complex spatial footprints.

1.8.2 Dissertation Synopsis

This dissertation is organized based on an accumulation of five individual but related publications which are included as individual chapters in this dissertation: Chapter 1, Chapter 2, Chapter 3, Chapter 4, and Chapter 5.

As the introduction of the whole dissertation, the current chapter (Chapter 1) is an extended version of Mai et al. [74] which describes the challenges, uniqueness of GeoQA and provides a classification for geographic questions. We also list three research questions that will be answered in the following four chapters. Chapter 2 focuses on RQ1 - geographic question relaxation. Chapter 3 and Chapter 4 target RQ2 - the location encoding problem while Chapter 5 answers RQ3 - the question of polygon encoding.

Chapter 2 presents a spatially-explicit knowledge graph embedding model which is developed for geographic question relaxation. We show that by utilizing spatial knowledge, more specifically the distance decay effect, we can allow the QA model to correctly answer more geographic questions.

Chapter 3 targets the first part of RQ2 and presents a general-purpose location encoding model called Space2Vec. The generalizability of Space2Vec is shown in two geospatial tasks - POI type classification and geo-aware image classification.

Based on Chapter 3, Chapter 4 utilizes Space2Vec location encoder to design a location-aware knowledge graph embedding model called SE-KGE. We further apply it to two tasks - knowledge graph-based GeoQA and spatial semantic lifting. We show that by directly encoding the point locations as well as the bounding boxes of geographic entities into the knowledge graph embedding space, the resulting KG embedding model is able to outperform multiple baselines on both tasks.

Based on the GeoQA result analysis in Chapter 4, we find out that only encoding point locations and the bounding boxes of geographic entities is not sufficient to answer many geographic questions such as topological relation questions, cardinal direction questions, and so on. We need to encode precise geometry representations (e.g., polygons) into the embedding space for the GeoQA task. Chapter 5 presents a polygon encoding model which can encode polygonal geometries into the embedding space such that it can be utilized in many downstream tasks including GeoQA. We show the effectiveness of this model on the polygon-based spatial relation prediction task which is an important task for GeoQA.

Finally, Chapter 6 concludes this dissertation by summarizing the previous chapters and listing the theoretical and practical research contributions of this dissertation. We also discuss the limitations of this dissertation and point out several future research directions.

Chapter 2

Relaxing Unanswerable Geographic Questions Using A Spatially Explicit Knowledge Graph Embedding Model

Many geographic questions become unanswerable when they are handled by a normal question answering system. In order to take into account the uniqueness of geographic questions, this chapter presents a spatially-explicit knowledge graph embedding model, called TransGeo, for unanswerable geographic question relaxation. The distance decay effect has been considered in the embedding model training stage through distance-aware triple weight calculation and entity context sampling. The trained TransGeo model is applied to relax and rewrite unanswerable geographic questions. To evaluate this model, we construct a geographic knowledge graph - DB18 and an unanswerable geographic query dataset - *GeoUQ* based on DBpedia Knowledge Graph. Evaluation results show that our TransGeo model can outperform multiple existing knowledge graph embedding models on both the link prediction and geographic query relaxation task. Relaxing and rewriting unanswerable geographic questions to allow the QA system to be able to correctly answer more questions is the first step towards a fully functional GeoQA system. In this chapter, we show the importance of spatial thinking in the GeoQA system development process.

Peer Reviewed Publication	
Title	Relaxing Unanswerable Geographic Questions Using A Spatially Explicit Knowledge Graph Embedding Model
Authors	Gengchen Mai, Bo Yan, Krzysztof Janowicz, Rui Zhu
Venue	AGILE 2019: Geospatial Technologies for Local and Regional Development
Editors	Phaedon Kyriakidis, Diofantos Hadjimitsis, Dimitrios Skarlatos, Ali Mansourian
Publisher	Springer Nature Switzerland AG
Pages	21-39
Submit Date	December 9, 2018
Accepted Date	January 20, 2019
Publication Date	April 16, 2019
Copyright	Reprinted with permission from Springer Nature Switzerland AG
DOI	https://doi.org/10.1007/978-3-030-14745-7_2

Abstract: Recent years have witnessed a rapid increase in Question Answering (QA) research and products in both academic and industry. However, geographic question answering remained nearly untouched although geographic questions account for a substantial part of daily communication. Compared to general QA systems, geographic QA has its own uniqueness, one of which can be seen during the process of handling unan-

swerable questions. Since users typically focus on the geographic constraints when they ask questions, if the question is unanswerable based on the knowledge base used by a QA system, users should be provided with a relaxed query which takes distance decay into account during the query relaxation and rewriting process. In this work, we present a spatially explicit translational knowledge graph embedding model called TransGeo which utilizes an edge-weighted PageRank and sampling strategy to encode the distance decay into the embedding model training process. This embedding model is further applied to relax and rewrite unanswerable geographic questions. We carry out two evaluation tasks: link prediction as well as query relaxation/rewriting for an approximate answer prediction task. A geographic knowledge graph training/testing dataset, *DB18*, as well as an unanswerable geographic query dataset, *GeoUQ*, are constructed. Compared to four other baseline models, our TransGeo model shows substantial advantages in both tasks.

2.1 Introduction

In the field of natural language processing, Question Answering (QA) refers to the methods, processes, and systems which allow users to ask questions in the form of natural language sentences and receive one or more answers, often in the form of sentences [75]. In the past decades, researchers from both academia and industry have been competing to provide better models for various subtasks of QA. Nowadays, many commercial QA systems are widely used in our daily life such as Apple Siri and Amazon Alexa.

Although QA systems have been studied and developed for a long time, geographic question answering remained nearly untouched. Although geographic questions account for a large part of the query sets in several QA datasets and are frequently used as illustrative examples [11, 12], they are treated equally to other questions even though geographic questions are fundamentally different in several ways. First, many geographic questions are highly context-dependent and subjective. Although some geographic questions can be answered objectively and context independently such as *what is the location of the California Science Center*, the answers to many geographic questions vary according to when and where these questions are asked, and who asks them. Examples include *nightclubs near me that are 18+* (location-dependent), *how expensive is a ride from Stanford University to Googleplex* (time-dependent), and *how safe is Isla Vista* (subjective). Second, another characteristic of geographic questions is that the answers are typically derived from a sequence of spatial operations rather than extracted from a piece of unstructured text or retrieved from Knowledge Graphs (KG) which are the normal procedures for current QA systems. For example, the answer to the question *what is the shortest route from California Science Center to LAX* should be computed by a shortest path algorithm on a route dataset rather than searching in a text corpus. The third difference is that geographic questions are often affected by vagueness and uncertainty at the conceptual level [76], thereby making questions such as *how many lakes are there in Michigan* difficult to answer.[1].

Due to the previously mentioned reasons, it is likely to receive no answer given a geographic question. In the field of general QA such cases are handled by so-called (query) relaxation and rewriting techniques [77]. We believe that geographic questions will benefit from *spatially-explicit relaxation methods* in which the spatial adjacency and time continuity should be taken into account during relaxation and rewriting. Interestingly, only a few researchers have been working on geographic question answering [48, 78, 79]. In this paper, we will mainly focus on how to include spatial adjacency (distance decay effect) into the geographic query relaxation/rewriting framework.

The necessity of query relaxation/rewriting arises from the problem of *unanswerable*

[1] Where the answer can vary between 63,000 and 10 depending on the conceptualization of `Lake`.

questions [80]. Almost all QA systems answer a given question based on their internal knowledge bases (KB). According to the nature of such knowledge bases, current QA research can be classified into three categories: unstructured data-based QA [4, 5, 6, 7, 8], semi-structure table-based QA [9], and structured-KB-based QA (so-called semantic parsing) [11, 12, 13, 14, 11]. If the answer to a given question cannot be retrieved from these sources, this question will be called an *unanswerable question.* There are different reasons for unanswerable questions. The first reason is that the information this question focuses on is missing from the current KB. For example, if the question is *what is the weather like in Creston, California* (Question A) and if the weather information of Creston is missing in the current KB, the QA system will fail to answer it. Another reason may stem from logical inconsistencies of a given question. The question *which city spans Texas and Colorado* (Question B) is unanswerable no matter which KBs is used because these states are disjoint.

In order to handle these cases, the initial questions need to be relaxed or rewritten to answerable questions and spatial adjacency need to be considered in this process. A relaxed question to Question A can be *what is the weather like in San Luis Obispo County* because Creston is part of San Luis Obispo County. Another option is to rewrite Question A to a similar question: *what is the weather like in San Luis Obispo (City)* because San Luis Obispo is near to Creston. Which option to consider depends on the nature of the given geographic question. As for Question B, a relaxation solution would be to delete one of the contradictory conditions. Sensible query relaxation/rewriting should be based on both the similarity/relatedness among geographic entities (the distance decay effect) and the nature of the question. However, current relaxation/rewriting techniques [77, 81, 82] do not consider spatial adjacency when handling unanswerable questions, and, thus, often return surprising and counter-intuitive results.

The research contributions of our work are as follows:

1. We propose a spatially explicit knowledge graph embedding model, TransGeo, which explicitly models the distance decay effect.

2. This spatially explicit embedding model is utilized to relax/rewrite unanswerable geographic queries. To the best of our knowledge, we are the first to consider the spatial adjacency between geographic entities in this process.

3. We present a benchmark dataset to evaluate the performance of the unanswerable geographic question handling framework. The evaluation results show that our spatially explicit embedding model outperforms non-spatial models.

The remainder of this work is structured as follows. In Section 2.2, several works about unanswerable question handling are discussed. Next, we present our spatially explicit KG embedding model, TransGeo, and show how to use this model to do unanswerable geographic question relaxation/rewriting in Section 2.3. Then, in Section 2.4 we empirically evaluate TransGeo against 4 other baseline models in two tasks: link predication task, unanswerable geographic question relaxation/rewriting and approximate answer prediction task. Then we conclude our work in Section 2.5 and point out the future research directions.

2.2 Related Work

The *unanswerable question* problem was recently prominently featured in the open domain question answering research field by Rajpurkar et al. [80]. The authors constructs a benchmark dataset, SQuADRUn, by combining the existing Stanford Question Answering Dataset (SQuAD) with over 50,000 unanswerable questions. These new unanswerable questions are adversarially written by crowd-workers to *look similar to* the original answerable questions. In their paper, the unanswerable questions are used as negative samples to train a better QA model to discriminate unanswerable questions from answerable ones. In our work, we assume the question has already been parsed (e.g. to a SPARQL query) by a semantic parser and resulted in an empty answer set. The task is to relax or rewrite this question/SPARQL query and to generate a related query with its corresponding answer. In the Semantic Web field, SPARQL query relaxation aims to reformulate queries with too few or even no results such that the intention of the original query is preserved while a sufficient number of potential answers are generated [77].

Query relaxation models can be classified into four categories: similarity-based, rule-based, user-preference-based, and cooperative techniques-based models. Elbassuoni et al. [77] proposed a similarity-based SPARQL query relaxation method by defining a similarity metric on entities in a knowledge graph. The similarity metric are defined based on a statistic language model over the context of entities. The relaxed queries are then generated and ranked based on this metric. This query relaxation method is defined purely based on the similarity between SPARQL queries. In contrast, our model jointly considers the similarity between queries and the probability that a selected answer to the *relaxed* query is, indeed, the answer to *original* query. This is possible due to the so-called Open World Assumption (OWA) commonly used by Web-scale KG by which statements/triples missing from the knowledge graph can still be true unless they are explicitly declared to be false within the knowledge graph. Our model aims at relaxing or rewriting a query such that the top ranked rewritten queries are more likely to generate the correct answer to the original one if it would be known.

With the increasing popularity of machine learning models in question answering and the Semantic Web, knowledge graph embedding models have been used to either predict answers for failed SPARQL queries [83] or recommend similar queries [84, 82]. KG embedding models aim to learn distributional representations for components of a knowledge graph. Entities are usually represented as continuous vectors while relations, i.e., object properties, are typically represented as vectors (such as in TransE [85], TransH [86], and TransRW [87]), matrices (e.g. TransR [88]), or tensors. For a comprehensive explanation of different KG embedding models, readers are referred to a recent survey by Wang et al. [89].

Hamilton et al. [83] proposes a graph query embeddings model (GQEs) to predict answers for conjunctive graph queries in incomplete knowledge graphs. GQEs first embeds graph nodes (entities) in a low-dimensional space and represents logical operators as learned geometric operations (e.g., translation, rotation) in the embedding space. Based on the learned node embeddings and geometric operations, each conjunctive graph query can be converted into an embedding in a same embedding space. Then cosine similarity is used to compare the query embeddings and node embeddings, and subsequently rank the corresponding entities as potential answers to the current query. While GQEs have been successfully applied to representing conjunctive graph queries and entities in the same embedding space, they have some limitations. For instance, GQEs can only handle conjunctive graph queries, a subset of SPARQL queries. Additionally, the predicted answer to a conjunctive graph query is not associated with a relaxed/rewritten query as an explanation for the answer.

[82] proposed an entity context preserving translational KG embedding model to represent each entity as a low-dimensional embedding and each predicate as a translation operation between entities. The authors show that compared with TransE [85], the most popular and straightforward KG embedding model, their embedding model performs better in terms of approximating answers to empty answer SPARQL queries. They also present an algorithm to compute *similar* queries to the original SPARQL queries based on the approximated answers. Our work is developed based on this work by overcoming some limitations and including distance decay in the embedding model training process.

2.3 Method

Before introducing our spatially explicit KG embedding model, we briefly outline concepts relevant to our work.

Definition 1 *Knowledge Graph: A knowledge graph (KG) is a data repository, which is typically organized as a directed multi-relational graph. Let* $G = \langle E, R \rangle$ *be a knowledge graph where* E *is a set of entities (nodes) and* R *is a set of relations (labeled edges). A triple* $T_i = (h_i, r_i, t_i)$ *can be interpreted as an edge connecting the head entity* h_i *(subject) with the tail entity* t_i *(object) by relation* r_i *(predicate).* [2]

Definition 2 *Entity Context: Given an entity* $e \in E$ *in the knowledge graph* G*, the context of* e *is defined as* $C(e) = \{(r_c, e_c)|(e, r_c, e_c) \in G \vee (e_c, r_c, e) \in G\}$.

Definition 3 *Basic Graph Pattern (BGP): Let* V *be a set of query variables in a SPARQL query (e.g., ?place). A basic graph pattern in a SPARQL query is a set of triple patterns* (s_i, p_i, o_i) *where* $s_i, o_i \in E \cup V$ *and* $p_i \in R$*. Put differently, we restrict triple patterns and thus BGP to cases where the variables are in the subject or object position.*

Definition 4 *SPARQL select query: For the purpose of this work, a SPARQL select*[3] *query* Q_j *is defined as the form:* $Q_j =$ `SELECT` V_j `FROM` KG `WHERE` GP *where* $V_j \subseteq V$ *and* KG *is the studied knowledge graph and* GP *is a BGP.*

The SPARQL query 2.1 shows an example which corresponds to the natural language question: *In which computer hardware company located in Cupertino is/was Steve*

[2] Note that in many knowledge graphs, a triple can include a datatype property as the relation where the tail is a literal. In our work, we do not consider these kind of triple as they are not used in any major current KG embedding model. We will use head (h), relation (r), and tail(t) when discussing embeddings and subject (s), predicate (p), object (o) when discussing Semantic Web knowledge graphs to stay in line with the literature from both fields.

[3] We ignore `ASK`, `CONSTRUCT`, and `DESCRIBE` queries here as they are not typically used for question answering, and, thus, also not considered in related work.

Jobs a board member. The answer should be dbr:Apple_Inc. If the triple (dbr:Apple_Inc,dbo:locationCity,dbr:Cupertino,_California), however, is missing from current KG, this question would become an unanswerable geographic question. Compared to the full SPARQL 1.1 language standard, two limitations of the given definition of a SPARQL query should be clarified:

1. Predicates in a SPARQL 1.1 BGP can also be a variables. Hence, Definition 3 presents a subset of all triple patterns, which can appear in a standard SPARQL query.

2. SPARQL 1.1 also contains other operations (UNION, OPTION, FILTER, LIMIT, etc.) not considered here and in related state-of-the-art work [82, 83] .

```
SELECT ?v
WHERE {
 ?v dbo:locationCity dbr:Cupertino,_California .
 ?v dbo:industry dbr:Computer_hardware .
 dbr:Steve_Jobs dbo:board ?v .
}
```

Listing 2.1: An example SPARQL query generated by a semantic parser.

Given a SPARQL query Q_j parsed from a natural language geographic question, if executing Q_j on the current KG yields an empty answer set, our goal is: **1)** learn a spatially explicit KG embedding model for the current KG which takes distance decay into account; **2)** use the embedding model to infer a ranked list of approximated answers to this question; and **3)** generate a relaxed/related SPARQL query for each approximate answer as an explanation for the query relaxation/rewriting process.

2.3.1 Modeling Geographic Entity Context in Knowledge Graphs

Based on the examples about relaxing or rewriting Question A and Question B in the introduction, we observe that a suitable query relaxation/rewriting for an unanswerable geographic question should consider both the similarity/relatedness among geographic entities (e.g., the distance decay effect) as well as the nature of the question. In terms of measuring semantic similarities among (geographic) entities in a knowledge graph, we borrow the assumption of distributional semantics from computational linguistic that *you shall know a word by the company it keeps* [90]. In analogy, the semantic similarity among (geographic) entities can be measured based on their contexts [70].

With regards to measuring the similarity/relatedness between general entities in a knowledge graph, both Elbassuoni et al. [77] and Wang et al. [82] consider the one degree neighborhood of the current entity as its context, which is shown in Definition 2. However, **this entity context modeling falls apart when geographic entities are considered in two ways**. First, this geographic entity context modeling does not fully reflect *Tobler's first law of geography*, which indicates that *near things are more related than distant things*. Since Definition 2 only considers object property triples as the entity context and disregard all datatype properties, all positional information, e.g., geographic coordinates, would not be considered in the context modeling. Although the place hierarchy is encoded as object property triples in most KG, e.g., GeoNames, GNIS-LD, and DBpedia, and these triples can also indirectly introduce distance decay effects into the context modeling, such contextual information is far too coarse. For example, Santa Barbara County, Los Angeles County, and Humboldt County are all subdivisions of California. From a place hierarchy perspective, all three should have the same relatedness to each other. But Santa Barbara County is more related to Los Angeles County rather than Humboldt County.

The second reason is due to the way geographic knowledge is represented in Web-scale knowledge graphs. For any given populated place, the place hierarchy of administrative units is modeled using the same canonical predicates. Put differently, even if no other triples are known about a small settlement, the KG will still contain at least a triple about a higher-order unit the place belongs to, e.g., a county. Consequently, all populated places in, say, Coconino County, Arizona, will share a common predicate (e.g., `dbo: isPartOf`) and object (e.g., `dbr:Coconino_County,_Arizona`) . For tiny deserted settlements such as Two Guns, AZ this may also be the sole triple known about them. In contrast, major cities in the same county or state, e.g., Flagstaff, will only have a small percentage of their total object property triples be about geographic statements. This will result in places about which not much is known to have an artificially increased similarity.

These aforementioned two reasons demonstrate the necessity to model geographic entity context in a different way rather than Definition 2. In this work, we redefine Definition 2 by combing an edge-weighted PageRank and a sampling procedure. The underlying idea is to assign larger weights to geographic triples in an entity context where the weights are modeled from a distance decay function.

To provide a final and illustrative example of the problems that arise form embedding models that are not spatially explicit, consider the work by Wang et al. [82]. Their query example is *which actor is born in New York and starred in a United States drama film directed by Time Burton.* After passing the SPARQL version of this question to their query relaxation/rewriting model, the model suggests to change the birthplace from New York to Kentucky which is certainly a surprising relaxation from the original query. Although Kentucky is also a place as New York, it is too far away from the brithplace, New York, the QA system user is interested in. A more reasonable relaxed/rewritten query should replace New York City with its nearby places, e.g. New Jersey.

2.3.2 Spatially Explicit KG Embedding Model

Given a knowledge graph $G = \langle E, R \rangle$, a set of geographic entities $P \subseteq E$, and a triple $T_i = (h_i, r_i, t_i) \in G$, we treat G as an undirected, unlabeled, edge-weighted multigraph MG, which means that we ignore the direction and label (predicate) for each triple in G. The weight $w(T_i)$ for triple T_i is defined in Equation 2.1, where D is the longest (simplified) earth surface distance which is half of the length of the equator measured in kilometer; $dis(h_i, t_i)$ is the geodesic distance between geographic entity h_i and t_i on the surface of an ellipsoidal model of the earth measured in kilometer. The ε is a hyperparameter to handle the cases where h_i and t_i are collocated; and l is the lowest edge weight we allow for each triple. If the head place and tail place of a geographic triple are too far apart, we set its weight as the lower bound l, indicating that we do not expect strong spatial interaction at this distance. [4]

$$w(T_i) = \begin{cases} \max(\ln \frac{D}{dis(h_i,t_i)+\varepsilon}, l) & \text{if } h_i \in P \wedge t_i \in P \\ l & \text{otherwise} \end{cases} \tag{2.1}$$

The location of h_i and t_i are represented as their geographic coordinates stored in a knowledge graph, which are usually points. In this work, we use the `geo:geometry` property to get the coordinates of all geographic entities in DBpedia.

After we compute weights for each triple in MG, an edge-weighted PageRank is applied to this weighted multigraph, where edge weights are treated as the transition probability of the random walker from one entity node to its neighboring entity node. In order to prevent the random walker to get stuck at one *sinking node*, the PageRank algorithm also defines a teleport probability, which allows the random walker to jump to

[4] We leave the fact that interaction depends on the travel mode and related issues for further work. Similarity, due to the nature of existing knowledge graphs, we use point data to represent places despite the problems this may introduce. Work on effectively integrating linestrings, polygons, and topology into Web-scale knowledge graphs is ongoing [35].

a random node in MG with a certain probability at each time step. Let $PR(e_i)$ be the PageRank score for each entity e_i in the knowledge graph, then $PR(e_i) \in (0, 1)$ represents the probability of a random walker to arrive at entity e_i after n time steps. If e_i had a lot of one degree triples (i.e., $|C(e_i)|$ is large), then e_i would have a larger $PR(e_i)$. Since $\sum_i PR(e_i) = 1$ and $|C(e_i)|$ have a long tail distribution, $PR(e_i)$ will also have a long tail distribution with few very large values but many small values. This skewed distribution would affect the later sampling process. In order to normalize $PR(e_i)$, we apply a *damping* function (Equation 2.2). In Equation 2.2, ln is the natural log function; N is the number of entities in the knowledge graph G. This function has the nice property that $w(e_i)$ increases monotonically w.r.t. $PR(e_i)$ and the distribution of $w(e_i)$ is more normalized than $w(e_i)$. Therefore, $w(e_i)$ encodes the structural information of the original knowledge graph and the distance decay effect on interaction (and similarity/relatedness more broadly) among geographic entities. The more incoming and outgoing triples one entity e_i has, the larger its $w(e_i)$ will be. Also, the closer two geographic entities $e_i, e_j \in P$ are, the larger $w(e_i)$ and $w(e_j)$ would be.

$$w(e_i) = N \cdot \frac{\frac{1}{-\ln PR(e_i)}}{\sum_i \frac{1}{-\ln PR(e_i)}} \tag{2.2}$$

Next, we introduce the knowledge graph embedding model, which utilizes $w(e_i)$ as the distribution from which the entity context is sampled. Since $w(e_i)$ directly encodes the distance decay information among geographic entities, we call our model spatially explicit KG embedding model, denoted here as TransGeo.

Translation-based KG embedding models embed entities into low-dimensional vector spaces while relations are treated as translation operations in either the original embedding space (TransE) or relation-specific embedding space (TransH, TransR). This geometric interpretation provides us with a useful way to understand the embedding-based

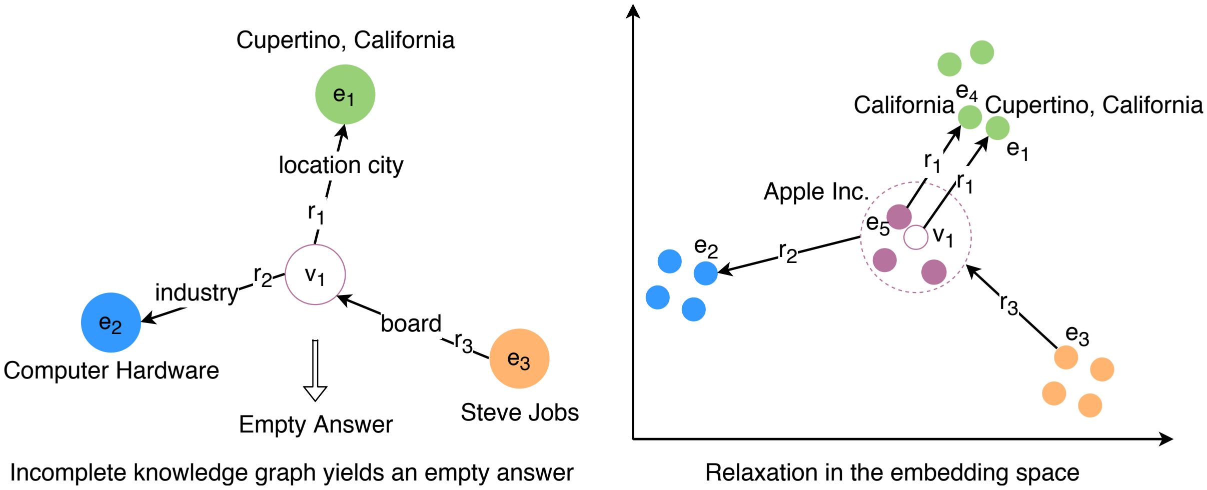


Figure 2.1: An unanswerable geographic query example and its corresponding KG embedding query relaxation/rewritten process.

Figure 2.1 shows the basic graph pattern of Query 2.1 and their vector representations in KG embedding space. If triple (dbr:Apple_Inc,dbo:locationCity,dbr:Cupertino,_California) is missing from the current KG, this query becomes an unanswerable query. However, if we already obtained the learned embeddings for e_1, e_2, e_3, r_1, r_2, and r_3, we could compute the embedding of the query variable $?v$ with each triple pattern. Next, we can compute the weighted average of these embeddings to get the final embedding of $?v$, which is denoted as $\mathbf{v}$. Next, the k-nearest neighbor entities of $\mathbf{v}$ can be obtained based on the cosine similarity between their embeddings. These k-nearest neighbor entities are treated as approximated answers to the original query 2.1. Based on each of these candidate answers, we cycle through each triple pattern in the original Query 2.1 to see whether they need to be relaxed or not, which is the major procedure for embedding-based query relaxation/rewritten.

In order to make the embedding-based query relaxation/rewriting process work well, the KG embedding model should be an entity context preserving model. However, one problem for the original TransE model is that each triple is treated independently in the training process which does not guarantee its context preservation. Inspired by the Continuous-Bag-of-word (CBOW) word embedding model [91], [82] proposed an entity context preserved KG embedding model which predicts the *center* entity based on the entity context (Definition 2). However, as we discussed in Section 2.3.1, the geographic entity context can not be fully captured by using Definition 2 and we need another method to capture the distance decay effect, where $w(e_i)$ plays a role. Another shortcoming of the embedding model proposed in Wang et al. [82] is that the size of entity context $|C(e_i)|$ varies among different entities which will make the number of triples trained in each batch different. This will have a negative effect on the model optimization process. Some entities may have thousands of incoming and outgoing triples, e.g., `dbr:United_States` has 232,573 context triples. This will imply that the model parameters will only update once all these triples are processed which is not a good optimization technique.

Based on this observation, we define a hyperparameter d as the context sampling size for each entity. If $|C(e_i)| > d$, then the context $C(e_i)$ of entity e_i would not be fully used in each KG embedding training step. Instead, the training context $C_{samp}(e_i)$ is sampled from $C(e_i)$ ($C_{samp}(e_i) \subseteq C(e_i)$) while the sampling probability of each context item (r_{ci}, e_{ci}) is calculated based on the damped PageRank value $w(e_{ci})$. If $|C(e_i)| > d$, the training context $C_{samp}(e_i)$ is sampled without replacement. If $|C(e_i)| < d$, $C_{samp}(e_i)$ is sampled with replacement. After a certain number of epochs t_{freq}, $C_{samp}(e_i)$ will be resampled for each entity. Because of this sampling strategy, a context item (r_{ci}, e_{ci}) of e_i would have a higher chance to be sampled if $e_i \in P \wedge e_{ci} \in P$, and e_i is close enough to e_{ci} in geographic space.

$$P(r_{ci}, e_{ci}) = \frac{w(e_{ci})}{\sum_{(r_{cj}, e_{cj}) \in C(e_i)} w(e_{cj})}, \text{ where } (e_i, r_{ci}, e_{ci}) \in G \vee (e_{ci}, r_{ci}, e_i) \in G \tag{2.3}$$

Based on the definition of entity training context $C_{samp}(e_i)$, a compatibility score between $C_{samp}(e_i)$ and an arbitrary entity e_k can be computed as Equation 2.4, in which $\phi(e_k, r_{cj}, e_{cj})$ is the plausibility score function between (r_{cj}, e_{cj}) and e_k. In Equation 2.5, $\|\cdot\|$ represents the $L1$-norm of the embedding vector; $\mathbf{e_k}, \mathbf{e_{cj}}$ represent the KG embeddings for the corresponding entity e_k, e_{cj}, and $\mathbf{r_{cj}}$ is the relation embedding of r_{cj}.

$$f(e_k, C_{samp}(e_i)) = \frac{1}{|C_{samp}(e_i)|} \cdot \sum_{(r_{cj}, e_{cj}) \in C_{samp}(e_i)} \phi(e_k, r_{cj}, e_{cj}) \tag{2.4}$$

$$\phi(e_k, r_{cj}, e_{cj}) = \begin{cases} \|\mathbf{e_k} + \mathbf{r_{cj}} - \mathbf{e_{cj}}\| & \text{if } (e_i, r_{cj}, e_{cj}) \in G \\ \|\mathbf{e_{cj}} + \mathbf{r_{cj}} - \mathbf{e_k}\| & \text{if } (e_{cj}, r_{cj}, e_i) \in G \end{cases} \tag{2.5}$$

The same assumption has been used here as TransE, which is that, in the *perfect* situation, if $(h_i, r_i, t_i) \in G$, $\|\mathbf{h_i} + \mathbf{r_i} - , \mathbf{t_i}\| = 0$. Based on Equation 2.4 and 2.5, if $e_k = e_i$, each $\phi(e_k, r_{cj}, e_{cj})$ would be small and close to zero, thus $f(e_i, C_{samp}(e_i))$ would be also small and close to zero. In contrast, if $C(e_k) \cap C(e_i) = \oslash$, each $\phi(e_k, r_{cj}, e_{cj})$ would be very large and $f(e_i, C_{samp}(e_i))$ would also also large.

In order to set up the learning task, the pairwise ranking loss function has been used as the objective function like most KG embedding models do. Specifically, for each entity e_i, we randomly sample K entities as the negative sampling set $Neg(e_i)$ for e_i. Equation 4.12 shows the objective function of TransGeo, where γ is the margin and $max()$ is the maximum function.

$$\mathcal{L} = \sum\nolimits_{e_i \in G} \sum\nolimits_{e_i' \in Neg(e_i)} max\Big(\gamma + f(e_i, C_{samp}(e_i)) - f(e_i', C_{samp}(e_i)), 0\Big) \tag{2.6}$$

2.3.3 KG Embedding Model Based Query Relaxation and Rewriting

After obtaining the learned TransGeomodel, we adopt the same procedure as [82] to relax/rewrite the query. We briefly summarize the process below. We assume a SPARQL query Q with two variables $?v_1$ and $?v_2$, which are targets to be relaxed/rewritten in order to find approximated answers.

1. Given an empty answer SPARQL query Q, we partition the basic graph pattern into several groups such that all triple patterns in one group only contain one variable. Triples who have two variables $?v_1$ and $?v_2$ (connected triples) as its subject and object respectively are treated differently;

2. For each triple pattern group which contains variable $?v$, the embedding of $?v$ is first computed by each triple pattern based on the translation operations from the entity node to the variable node. Then the final embedding of $?v$ is computed as the weighted average of previous computed variable embeddings. The weight is calculated based on the number of matched triples of each triple patterns in the KG;

3. If Q has any connection triples, the embeddings of variables computed from each triple pattern group are refined based on the predicate of the connection edges. Then these embeddings will be treated as the final embeddings for each variable;

4. The approximate answers to each variable are determined by using their computed variable embeddings to search for the k-nearest embeddings of entities based on their cosine similarity. Each variable will have a ranked list of entities, e.g. $A(?v_1)$, $A(?v_2)$, as their approximated answers;

5. If Q has any connection triples, e.g. $(?v_1, r, ?v_2)$, we need to first use beam search to get top-K answer tuples for $?v_1$ and $?v_2$. And then each answer tuple (e_{1i}, e_{2j}) is checked for the condition $(e_{1i}, r, e_{2j}) \in G$. The answer tuples which satisfy this condition will be returned as a ranking list $Ans(Q)$ of approximated answers;

6. For each answer tuple $(e_{1i}, e_{2j}) \in Ans(Q)$, we enumerate each triple pattern to check the satisfaction. As for triple $(?v_1, r, e)$, if $(e_{1i}, r, e) \in G$, we do not perform any relaxation. If $(e_{1i}, r, e) \notin G$, then $(?v_1, r, e)$ will be relaxed based on Equation 4.3. However, if e_{1i} does not have any outgoing triples, this triple pattern could not be relaxed and we would delete this triple pattern from the query relaxation/rewriting result. But the similarity score of this relaxation result will be set to 0;

7. The ranked list of answer tuples as well as the relaxed queries associated with them are returned to the users.

$$(e_{1i}, r_k, e_k) = \arg\max\left(\frac{\mathbf{r} \cdot \mathbf{r_k}}{\|\mathbf{r}\| \cdot \|\mathbf{r_k}\|} + \frac{\mathbf{e} \cdot \mathbf{e_k}}{\|\mathbf{e}\| \cdot \|\mathbf{e_k}\|}\right) \tag{2.7}$$

2.4 Experiment

Since almost all the established knowledge graph training dataset for KG embedding models, e.g., FB15K, WN18, do not contain enough geographic entities, we collect a new KG embedding training dataset, *DB18*[5], which is a subgraph of DBpedia. The dataset construction procedure is as follow: 1) We first selected all geographic entities which are part of (`dbo:isPartOf`) `dbr:California` with type (`rdf:type`) `dbo:City` which yields 462 geographic entities; 2) We use these entities as seeds to get their 1-degree and 2-degree object property triples and filter out triples with no `dbo:` properties; 3) we delete

[5] `https://github.com/gengchenmai/TransGeo`

Table 2.1: Summary statistic for *DB18*

DB18	Total	Training	Testing
# of triples	139155	138155	1000
# of entities	22061	-	-
# of relations	281	-	-
# of geographic entities	1681 (7.62%)	-	-

the entities and their associated triples whose node degree is less than 10; 4) we split the triple set into training and testing set and make sure that every entity and relation in the testing dataset will appear in training dataset. The statistic of *DB18* is listed in Table 2.1. 'Geographic entities' here means entities with a `geo:geometry` property.

Following the method we describe in Section 2.3.2, we compute the edge weights for each triple in *DB18* and an edge-weighted PageRank algorithm is applied on this undirected unlabeled multigraph. Here we set l to 1 and ε to 1. We select four models as the baseline models to compare with TransGeo: **1)** *TransE*; **2)** the context preserving translational KG embedding [82]; **3)** a simplified version of TransGeo in which the entity context items are randomly sampled from a uniform distribution, denoted as $TransGeo_{unweighted}$; **4)** another simplified version of our model in which the PageRank are applied to unweighted multigraph, denoted as $TransGeo_{regular}$. We implement *TransE*, $TransGeo_{unweighted}$, $TransGeo_{regular}$, and TransGeo, in Tensorflow. We use the original Java implementation of Wang et al. [82][6]. For all five models, we train them for 1000 epochs with the margin $\gamma = 1.0$ and learning rate $\alpha = 0.001$. As for $TransGeo_{unweighted}$, $TransGeo_{regular}$, and TransGeo, we use 30 as the entity context sampling size d and 1000 as batch size. We resample the entity context every 100 epochs. As for the context preserving translational KG embedding [82], we use 10 as the entity context size cut-off

[6] `https://github.com/wangmengsd/re`

value. The embedding dimension of all these five embedding models is 50.

In order to demonstrate the effectiveness of our spatially explicit KG embedding model, TransGeo, over the other four baseline models, we evaluate these five KG embedding models in two task: the standard link predication task and an relaxation/rewriting task to predict answers to the otherwise unanswerable geographic questions. The evaluation results are listed in Table 2.2.

The common link prediction task is used to validate the translation preserving characteristic of different models. The set up of the link prediction task follows the evaluation protocol of Bordes et al. [85]. Given a correct triple $T_k = (h_k, r_k, t_k)$ from the testing dataset of DB18, we replace the head entity h_k (or tail t_k) with all other entities from the dictionary of DB18. The plausibility scores for each of those n triples are computed based on the plausibility score functions of *TransE* ($\| \mathbf{h} + \mathbf{r} - \mathbf{t} \|$). Then these triples are ranked in ascending order according to this score. The higher the correct triple ranks in this list, the better this learned model. Note that some of the corrupted triples may also appear in the KG. For example, as for triple (`dbr:Santa_Barbara,_California`, `dbo:isPartOf`, `dbr:California`), if we replace the head `dbr:Santa_Barbara,_California` with `dbr:San_Francisco`, the result corrupted triple (`dbr:San_Francisco`, `dbo:isPartOf`, `dbr:California`) is still in the DBpedia KG. These false negative samples need to be filtered out. Mean reciprocal rank (*MRR*) and *HIT@10* are used as evaluation matrics where *Raw* and *Filter* indicate the evaluation results on the original ranking of triples or the filtered list which filters out the false negative samples. According to Table 2.2, TransGeo performs the best in most of the metrics and the only metric TransGeo cannot outperform is *MRR* in the raw setting. This evaluation shows that our spatially explicit model does indeed hold the translation preserving characteristic.

For the quality of the unanswerable geographic query relaxation/rewriting results,

Table 2.2: Two evaluation tasks for different KG embedding models

	Link Prediction				SPARQL Relaxation	
	MRR		HIT@10		MRR	HIT@10
	Raw	Filter	Raw	Filter		
TransE Model	**0.122**	0.149	30.00%	34.00%	0.008	5% (1 out of 20)
Wang et al. [82]	0.113	0.154	27.20%	30.50%	0.000	0% (0 out of 20)
$TransGeo_{regular}$	0.094	0.129	28.50%	33.40%	0.098	25% (5 out of 20)
$TransGeo_{unweighted}$	0.108	0.152	30.80%	37.80%	0.043	15% (3 out of 20)
TransGeo	0.104	**0.159**	**32.40%**	**42.10%**	**0.109**	**30% (6 out of 20)**

we evaluate the results based on the ranking of the approximate answers [83]. Let's take Question 2.1 as an example. One reason which causes an empty answer is that some triples were missing from the KG, e.g., (`dbr:Apple_Inc,dbo:locationCity,dbr:Cupertino,_California`), and the current SPARQL query is overly restrictive. However, based on the KG embedding model, we can approximate the embeddings of the variables in the current query. This variable embeddings can be used to search for the most *probable* answers/entities to each variable in the embedding space. These *k-nearest* entities are assumed to be more probable to be the correct answer of the original question. The correct answer (based on the Open World Assumption) to Question 2.1 is `dbr:Apple_Inc`. If the KG embedding is good at preserving the context of entities, the embedding of `dbr:Apple_Inc` will appear close to the computed variable embedding (See Figure 2.1). So the performance of the query relaxation/rewriting algorithm can be evaluated by checking the rank of the correct answer in the returned ranking list of the approximate answers.

Based on the above discussion, we construct another evaluation dataset, *GeoUQ*, which is composed of 20 unanswerable geographic questions. Let G_{train} be a knowledge graph which is composed of all the training triples of DB18[7] and G_{all} be a knowledge graph containing all training and testing triples in DB18[8]. Both G_{train} and G_{all} can be accessed through the SPARQL endpoint. These queries satisfy 2 conditions: 1) each query Q will yield empty answer set when executing Q on G_{train}; 2) Q will return only one answer when executing Q on G_{all}. The reason for making Q a one-answer query in G_{all} is that the user also expects one answer from the QA system to the question (s)he poses. One-answer queries are also the common setup for many QA benchmark datasets, e.g. WikiMovie [5], WebQuestionsSP [11]. *MRR* and *HIT@10* are used as evaluation metrics for this task.

All five KG embedding models are evaluated based on the same query relaxation/rewriting implementation. The evaluation results are shown in Table 2.2. From Table 2.2, we can conclude that TransGeo outperform all the other baselines models both on *MRR* and *HIT@10*.

Table 2.3 show the top 3 query relaxation/rewriting results of Question 2.1 from all the 5 KG embedding models. For each query, the highlighted part in the BGP is the part where the query is changed from the original Query 2.1. Note that some of the relaxation/rewriting results have less triple patterns than the original Query 2.1. This is because the current approximate answer/entity does not have any outgoing or incoming triples to be set as the alternative to the original triple pattern. Hence, we delete this triple pattern. This has been described in Step 6 in Section 2.3.3. From Table 2.3, we can see that the correct answer `dbr:Apple_Inc` has been listed as the second approximate answer for TransGeo. However, all the 4 baseline models fail to predict this correct answer in their top 10 approximate answers list. Besides the perspective of predicting

[7] `http://stko-testing.geog.ucsb.edu:3080/dataset.html?tab=query&ds=/GeoQA-Train`

[8] `http://stko-testing.geog.ucsb.edu:3080/dataset.html?tab=query&ds=/GeoQA-All`

the correct answers, we can also evaluate the models by inspecting the quality of the relaxed/rewritten queries. For example, the top 1 relaxed query from TransGeo changes `dbr:Cupertino,_California` to `dbr:Redwood_City,_California` which is a nearby city of `dbr:Cupertino,_California`. Although the predicted answer is `dbr:NeXT` rather than `dbe:Apple_Inc`, this query relaxation/rewriting makes sense and is meaningful for the user. The 2nd relaxed result from TransGeo changes `dbr:Cupertino,_California` to `dbr:California` which is a superdivision of `dbr:Cupertino,_California`. This is indeed a real *query relaxation* which relaxes the geographic constraint to its superdivision. In short, our spatially explicit KG embedding model, TransGeo, produces better result than all baseline models.

2.5 Conclusion

In this work, we discussed why geographic question answering differs from general QA in general, and what this implies for relaxation and rewriting of empty queries specifically. We demonstrated why distance decay has to be included explicitly in the training of knowledge graph embeddings and showed cases of neglecting to do so. As a result, we propose a spatially explicit KG embedding models, TransGeo, which utilizes an edge-weighted PageRank and sampling strategy to include the distance decay effect into the KG embedding model training. We constructed a geographic knowledge graph training dataset, *DB18* and evaluated TransGeo as well as four baseline models. We also created an unanswerable geographic question dataset (*GeoUQ*) for two evaluation tasks: link prediction and answer prediction by relaxation/rewriting. Empirical experiments show that our spatially explicit embedding model, TransGeo, can outperform all the other 4 baseline methods on both task. As for the link prediction task, in the filter setting, our model outperforms the other baselines by at least 3.2% at *MRR* and 11.4% at *HIT@10*. In

Table 2.3: Query relaxation/rewriting results of different KG embedding models for Query 2.1

	Relaxation 1	Relaxation 2	Relaxation 3

TransE	**Query:**
	SELECT ?v WHERE {
	?v dbo:locationCity **dbr:Fountain_Valley,_California** .
	?v dbo:industry **dbr:Data_storage_device** .
	dbr:John_Tu dbo:knownFor ?v .
	}
	Answer: dbr:Kingston_Technology
		Query:
		SELECT ?v WHERE {
		?v dbo:locationCity **dbr:Agoura_Hills,_California** .
		?v dbo:industry **dbr:Interactive_entertainment** .
		dbr:Heavy_Iron_Studios dbo:owningCompany ?v .
		}
		Answer: dbr:THQ
			Query:
			SELECT ?v WHERE {
			?v dbo:locationCity **dbr:Pasadena,_California** .
			?v dbo:industry **dbr:Entertainment** .
			}
			Answer: dbr:Landmark_Entertainment_Group

Wang et al. [82]	**Query:**
	SELECT ?v
	WHERE {
	?v dbo:locationCity **dbr:Irvine,_California** .
	?v dbo:industry **dbr:Video_game_industry** .
	dbr:Activision_Blizzard dbo:division ?v .
	}
	Answer: dbr:Blizzard_Entertainment
		Query:
		SELECT ?v
		WHERE {
		?v dbo:locationCity **dbr:Mountain_View,_California** .
		?v dbo:industry **dbr:Interactive_entertainment** .
		}
		Answer: dbr:Paragon_Studios
			Query:
			SELECT ?v
			WHERE {
			?v dbo:locationCity **dbr:San_Mateo,_California** .
			?v dbo:industry **dbr:Video_game** .
			}
			Answer: dbr:Digital_Pictures

$TransGeo_{unweighted}$	**Query:**
	SELECT ?v
	WHERE {
	?v dbo:locationCity **dbr:Palo_Alto,_California** .
	?v dbo:industry dbr:Computer_hardware .
	dbr:William_Redington_Hewlett dbo:knownFor ?v .
	}
	Answer: dbr:Hewlett-Packard
		Query:
		SELECT ?v
		WHERE {
		?v dbo:locationCity **dbr:California** .
		?v **dbo:keyPerson dbr:Elon_Musk** .
		dbr:Elon_Musk dbo:knownFor ?v .
		}
		Answer: dbr:SolarCity
			Query:
			SELECT ?v
			WHERE {
			?v dbo:locationCity **dbr:Santa_Clara,_California** .
			?v dbo:industry dbr:Computer_hardware .
			}
			Answer: dbr:Console_Inc

$TransGeo_{regular}$	**Query:**
	SELECT ?v
	WHERE {
	?v **dbo:foundationPlace dbr:Santa_Clara,_California** .
	?v dbo:industry dbr:Computer_hardware .
	}
	Answer: dbr:Jasomi_Networks
		Query:
		SELECT ?v
		WHERE {
		?v dbo:location **dbr:San_Carlos,_California** .
		?v dbo:industry dbr:Computer_hardware .
		}
		Answer: dbr:Check_Point
			Query:
			SELECT ?v
			WHERE {
			?v dbo:locationCity **dbr:Lake_Forest,_California** .
			?v dbo:industry dbr:Computer_hardware .
			}
			Answer: dbr:PSSC_Labs

TransGeo	**Query:**
	SELECT ?v
	WHERE {
	?v dbo:locationCity **dbr:Redwood_City,_California** .
	?v dbo:industry dbr:Computer_hardware .
	dbr:Steve_Jobs **dbo:occupation** ?v .
	}
	Answer: dbr:NeXT
		Query:
		SELECT ?v
		WHERE {
		?v dbo:locationCity **dbr:California** .
		?v dbo:industry dbr:Computer_hardware .
		dbr:Steve_Jobs **dbo:board** ?v .
		}
		Answer: dbr:Apple_Inc
			Query:
			SELECT ?v
			WHERE {
			?v **dbo:foundationPlace dbr:Sioux_City,_Iowa** .
			?v dbo:industry dbr:Computer_hardware .
			dbr:EMachines dbo:owningCompany ?v .
			}
			Answer: dbr:Gateway,_Inc

terms of the unanswerable geographic question approximate answer prediction task, our model outperform the other 4 baselines by at least 11.2% at *MRR* and 20% at *HIT@10*.

In terms of future work, firstly, the distance decay information is explicitly encoded into our KG embedding model which gives up on flexibility, e. g., to model modes of transportation. In the future, we want to explore ways to only consider distance decay during query relaxation rather than the model training step. Secondly, as for the method to compute the edge weights of the knowledge graph, we used point geometries which may yield misleading results for larger geographic areas such as states. This limitation is due to the availability of existing knowledge graphs. Work to support more complex geometries and topology is under way.

Chapter 3

Multi-Scale Representation Learning for Spatial Feature Distributions using Grid Cells

This chapter focuses on a more fundamental question that goes beyond the context of geographic question answering - *how to design a general-purpose representation learning model for (geographic) locations such that it can be utilized to design spatially-explicit machine learning models for multiple downstream geospatial tasks*. Inspired by the grid cell research in Neuroscience, we propose a multi-scale representation learning model called $Space2Vec$ to encode point locations into the embedding space. We utilize $Space2Vec$ to do POI type prediction based on 1) POI locations (location modeling) and 2) nearby POIs' types and locations (spatial context modeling). We also use $Space2Vec$ to do geo-aware image classification. Experiment results show that $Space2Vec$ can outperform multiple well-established baselines such as RBF kernels, tile embeddings, and multi-layer feed-forward networks on both tasks. Further performance analysis shows that the superiority of $Space2Vec$ is attributed to $Space2Vec$'s ability to handle distributions at different scales. In contrast, other baselines can at most well handle distributions at one scale but fail in other scales.

Peer Reviewed Publication	
Title	Multi-Scale Representation Learning for Spatial Feature Distributions using Grid Cells
Authors	Gengchen Mai, Krzysztof Janowicz, Bo Yan, Rui Zhu, Ling Cai, Ni Lao
Venue	The Eighth International Conference on Learning Representations (ICLR 2020)
Publisher	openreview.net
Pages	1-15
Submit Date	September 25, 2019
Accepted Date	December 19, 2019
Publication Date	December 19, 2019
Copyright	Reprinted with permission from openreview.net

Abstract: Unsupervised text encoding models have recently fueled substantial progress in Natural Language Processing (NLP). The key idea is to use neural networks to convert words in texts to vector space representations (embeddings) based on word positions in a sentence and their contexts, which are suitable for end-to-end training of downstream tasks. We see a strikingly similar situation in spatial analysis, which focuses on incorporating both absolute positions and spatial contexts of geographic objects such as Points of Interest (POIs) into models. A general-purpose representation model for space is valuable for a multitude of tasks. However, no such general model exists to date beyond simply applying discretization or feed-forward nets to coordinates, and little effort has been put into jointly modeling distributions with vastly different characteristics, which commonly

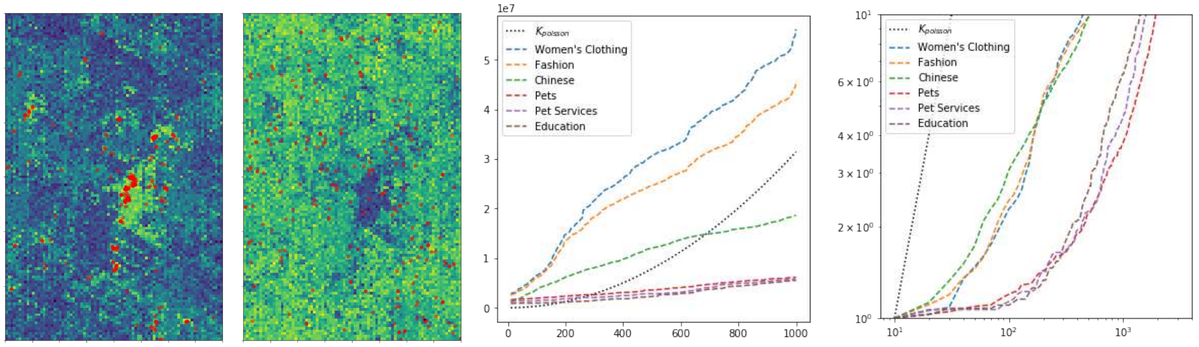
emerges from GIS data. Meanwhile, Nobel Prize-winning Neuroscience research shows that grid cells in mammals provide a multi-scale periodic representation that functions as a metric for location encoding and is critical for recognizing places and for path-integration. Therefore, we propose a representation learning model called Space2Vec to encode the absolute positions and spatial relationships of places. We conduct experiments on two real-world geographic data for two different tasks: 1) predicting types of POIs given their positions and context, 2) image classification leveraging their geo-locations. Results show that because of its multi-scale representations, Space2Vec outperforms well-established ML approaches such as RBF kernels, multi-layer feed-forward nets, and tile embedding approaches for location modeling and image classification tasks. Detailed analysis shows that all baselines can at most well handle distribution at one scale but show poor performances in other scales. In contrast, Space2Vec s multi-scale representation can handle distributions at different scales. [1]

3.1 Introduction

Unsupervised text encoding models such as Word2Vec [91], Glove [92], ELMo [93], and BERT [94] have been effectively utilized in many Natural Language Processing (NLP) tasks. At their core they train models which encode words into vector space representations based on their positions in the text and their context. A similar situation can be encountered in the field of Geographic Information Science (GIScience). For example, spatial interpolation aims at predicting an attribute value, e.g., elevation, at an unsampled location based on the known attribute values of nearby samples. Geographic information has become an important component to many tasks such as fine-grained image classification [64], point cloud classification and semantic segmentation [95], rea-

[1] Link to project repository: `https://github.com/gengchenmai/space2vec`

soning about Point of Interest (POI) type similarity [70], land cover classification [96], and geographic question answering [47]. Developing a *general* model for vector space representation of any point in space would pave the way for many future applications.



(a) Women's Cloth (b) Education (c) Ripley's K (d) Renormalized Ripley's K

Figure 3.1: The challenge of joint modeling distributions with very different characteristics. (a)(b) The POI locations (red dots) in Las Vegas and Space2Vec predicted conditional likelihood of Women's Clothing (with a clustered distribution) and Education (with an even distribution). The dark area in (b) indicates that the downtown area has more POIs of other types than education. (c) Ripley's K curves of POI types for which Space2Vec has the largest and smallest improvement over *wrap* [64]. Each curve represents the number of POIs of a certain type inside certain radios centered at every POI of that type; (d) Ripley's K curves renormalized by POI densities and shown in log-scale. To efficiently achieve multi-scale representation Space2Vec concatenates the grid cell encoding of 64 scales (with wave lengths ranging from 50 meters to 40k meters) as the first layer of a deep model, and trains with POI data in an unsupervised fashion.

However, existing models often utilize *specific* methods to deal with geographic information and often disregards geographic coordinates. For example, Place2Vec [70] converts the coordinates of POIs into spatially collocated POI pairs within certain distance bins, and does not preserve information about the (cardinal) direction between points. Li et al. [97] propose DCRNN for traffic forecasting in which the traffic sensor network is converted to a distance weighted graph which necessarily forfeits information about the spatial layout of sensors. There is, however, no general representation model beyond simply applying discretization [98, 99] or feed-forward nets [100, 64] to coordinates.

A key challenge in developing a general-purpose representation model for space is how to deal with mixtures of distributions with very different characteristics (see an example in Figure 3.1), which often emerges in spatial datasets [101]. For example, there are POI types with clustered distributions such as women's clothing, while there are other POI types with regular distributions such as education. These feature distributions co-exist in the same space, and yet we want a single representation to accommodate all of them in a task such as location-aware image classification [64]. Ripley's K is a spatial analysis method used to describe point patterns over a given area of interest. Figure 3.1c shows the K plot of several POI types in Las Vegas. One can see that as the radius grows the numbers of POIs increase at different rates for different POI types. In order to see the relative change of density at different scales, we renormalize the curves by each POI type's density and show it in log scale in Figure 3.1d. One can see two distinct POI type groups with different distribution patterns with clustered and even distributions. If we want to model the distribution of these POIs by discretizing the study area into tiles, we have to use small grid sizes for women's clothing while using larger grid sizes for educations because smaller grid sizes lead to over- parameterization of the model and overfitting. In order to jointly describe these distributions and their patterns, we need an encoding method which supports *multi-scale representations*.

Nobel Prize winning Neuroscience research [102] has demonstrated that grid cells in mammals provide a multi-scale periodic representation that functions as a metric for location encoding, which is critical for integrating self-motion. Moreover, Blair et al. [103] show that the multi-scale periodic representation of grid cells can be simulated by summing three cosine grating functions oriented 60° apart, which may be regarded as a simple Fourier model of the hexagonal lattice. This research inspired us to encode locations with multi-scale periodic representations. Our assumption is that decomposed geographic coordinates helps machine learning models, such as deep neural nets, and multi-scale representations deal with the inefficiency of intrinsically single-scale methods such as RFB kernels or discretization (tile embeddings). To validate this intuition, we propose an encoder-decoder framework to encode the distribution of point-features[2] in space and train such a model in an unsupervised manner. This idea of using sinusoid functions with different frequencies to encode positions is similar to the position encoding proposed in the Transformer model [104]. However, the position encoding model of Transformer deals with a discrete 1D space – the positions of words in a sentence – while our model works on higher dimensional continuous spaces such as the surface of earth.

In summary, the contributions of our work are as follows:

1. We propose an encoder-decoder encoding framework called Space2Vec using sinusoid functions with different frequencies to model absolute positions and spatial contexts. We also propose a multi-head attention mechanism based on context points. To the best of our knowledge, this is the first attention model that explicitly considers the spatial relationships between the query point and context points.

2. We conduct experiments on two real world geographic data for two different tasks: 1) predicting types of POIs given their positions and context, 2) image classification leveraging their geo-locations. Space2Vec outperforms well-established encoding methods such as RBF kernels, multi-layer feed-forward nets, and tile embedding approaches for location modeling and image classification.

3. To understand the advantages of Space2Vec we visualize the firing patterns (response maps) of location models' encoding layer neurons and show how they handle spatial structures at different scales by integrating multi-scale representations. Furthermore the firing patterns for the spatial context models neurons give insight into how the grid-like cells capture the decreasing distance effect with multi-scale

[2] In GIS and spatial analysis, 'features' are representations of real-world entities. A tree can, for instance, be modeled by a point-feature, while a street would be represented as a line string feature.

representations.

3.2 Problem Formulation

Distributed representation of point-features in space can be formulated as follows. Given a set of points $\mathcal{P} = \{p_i\}$, i.e., Points of Interests (POIs), in L-D space ($L = 2, 3$) define a function $f_{\mathcal{P},\theta}(\mathbf{x}) : \mathbb{R}^L \to \mathbb{R}^d$ ($L \ll d$), which is parameterized by θ and maps any coordinate $\mathbf{x}$ in space to a vector representation of d dimension. Each point (e.g., a restaurant) $p_i = (\mathbf{x}_i, \mathbf{v}_i)$ is associated with a location $\mathbf{x}_i$ and attributes $\mathbf{v}_i$ (i.e., POI features such as type, name, capacity, etc.). The function $f_{\mathcal{P},\theta}(\mathbf{x})$ encodes the probability distribution of point features over space and can give a representation of any point in the space. Attributes (e.g. place types such as *Museum*) and coordinate of point can be seen as analogies to words and word positions in commonly used word embedding models.

3.3 Related Work

There has been theoretical research on neural network based path integration/spatial localization models and their relationships with grid cells. Both Cueva et al. [105] and Banino et al. [106] showed that grid-like spatial response patterns emerge in trained networks for navigation tasks which demonstrate that grid cells are critical for vector-based navigation. Moreover, Gao et al. [107] propose a representational model for grid cells in navigation tasks which has good quality such as magnified local isometry. All these research is focusing on understanding the relationship between the grid-like spatial response patterns and navigation tasks from a theoretical perspective. In contrast, our goal focuses on utilizing these theoretical results on real world data in geoinformatics.

Radial Basis Function (RBF) kernel is a well-established approach to generating learning friendly representation from points in space for machine learning algorithms such as

SVM classification [108] and regression [109]. However, the representation is example based – i.e., the resultant model uses the positions of training examples as the centers of Gaussian kernel functions [110]. In comparison, the grid cell based location encoding relies on sine and cosine functions, and the resultant model is inductive and does not store training examples.

Recently the computer vision community shows increasing interests in incorporating geographic information (e.g. coordinate encoding) into neural network architectures for multiple tasks such as image classification [99] and fine grained recognition [98, 100, 64]. Both Berg et al. [98] and Tang et al. [99] proposed to discretize the study area into regular grids. To model the geographical prior distribution of the image categories, the grid id is used for GPS encoding instead of the raw coordinates. However, choosing the correct discretization is challenging [111, 112], and incorrect choices can significantly affect the final performance [113, 114]. In addition, discretization does not scale well in terms of memory use. To overcome these difficulties, both Chu et al. [100] and Mac Aodha et al. [64] advocated the idea of inductive location encoders which directly encode coordinates into a location embedding. However, both of them directly feed the coordinates into a feed-forward neural network [100] or residual blocks [64] without any feature decomposition strategy. Our experiments show that this direct encoding approach is insufficient to capture the spatial feature distribution and Space2Vec significantly outperforms them by integrating spatial representations of different scales.

3.4 Method

We solve *distributed representation of point-features in space* (defined in Section 3.2) with an encoder-decoder architecture:

1. Given a point $p_i = (\mathbf{x}_i, \mathbf{v}_i)$ a **point space encoder** $Enc^{(x)}()$ encodes location $\mathbf{x}_i$

into a location embedding $\mathbf{e}[\mathbf{x}_i] \in \mathbb{R}^{d^{(x)}}$ and a **point feature encoder** $Enc^{(v)}()$ encodes its feature into a feature embedding $\mathbf{e}[\mathbf{v}_i] \in \mathbb{R}^{d^{(v)}}$. $\mathbf{e} = [\mathbf{e}[\mathbf{x}_i]; \mathbf{e}[\mathbf{v}_i]] \in \mathbb{R}^d$ is the full representation of point $p_i \in \mathcal{P}$, where $d = d^{(x)} + d^{(v)}$. $[;]$ represents vector concatenation. In contrast, geographic entities not in $\mathcal{P}$ within the studied space can be represented by their location embedding $\mathbf{e}[\mathbf{x}_j]$ since its $\mathbf{v}_i$ is unknown.

2. We developed two types of decoders which can be used independently or jointly. A **location decoder** $Dec_s()$ reconstructs point feature embedding $\mathbf{e}[\mathbf{v}_i]$ given location embedding $\mathbf{e}[\mathbf{x}_i]$, and a **spatial context decoder** $Dec_c()$ reconstructs the feature embedding $\mathbf{e}[\mathbf{v}_i]$ of point p_i based on the space and feature embeddings $\{\mathbf{e}_{i1}, ..., \mathbf{e}_{ij}, ..., \mathbf{e}_{in}\}$ of nearest neighboring points $\{p_{i1}, ..., p_{ij}, ..., p_{in}\}$, where n is a hyper-parameter.

3.4.1 Encoder

Point Feature Encoder Each point $p_i = (\mathbf{x}_i, \mathbf{v}_i)$ in a point set $\mathcal{P}$ is often associated with features such as the air pollution station data associate with some air quality measures, a set of POIs with POI types and names, a set of points from survey and mapping with elevation values, a set of points from geological survey with mineral content measure, and so on. The point feature encoder $Enc^{(v)}()$ encodes such features $\mathbf{v}_i$ into a feature embedding $\mathbf{e}[\mathbf{v}_i] \in \mathbb{R}^{d^{(v)}}$. The implementation of $Enc^{(v)}()$ depends on the nature of these features. For example, if each point represents a POI with multiple POI types (as in this study), the feature embedding $\mathbf{e}[\mathbf{v}_i]$ can simply be the mean of each POI types' embeddings $\mathbf{e}[\mathbf{v}_i] = \frac{1}{H}\sum_{h=1}^{H} \mathbf{t}_h^{(\gamma)}$, where $\mathbf{t}_h^{(\gamma)}$ indicates the hth POI type embedding of a POI p_i with H POI types. We apply L_2 normalization to the POI type embedding matrix.

Point Space Encoder A part of the novelty of this paper is from the point space encoder $Enc^{(x)}()$. We first introduce Theorem 1 which provide an analytical solution $\phi(\mathbf{x})$ as the base of encoding any location $\mathbf{x} \in \mathbb{R}^2$ in 2D space to a distributed representation:

Theorem 1 *Let* $\mathbf{\Psi}(\mathbf{x}) = (e^{i\langle \mathbf{a}_j, \mathbf{x}\rangle}, j = 1, 2, 3)^T \in \mathbb{C}^3$ *where* $e^{i\theta} = \cos\theta + i \sin\theta$ *is the Euler notation of complex values;* $\langle \mathbf{a}_j, \mathbf{x}\rangle$ *is the inner product of* $\mathbf{a}_j$ *and* $\mathbf{x}$*.* $\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3 \in \mathbb{R}^2$ *are 2D vectors such that the angle between* $\mathbf{a}_k$ *and* $\mathbf{a}_l$ *is* $2\pi/3$*,* $\forall j, \|\mathbf{a}_j\| = 2\sqrt{\alpha}$*. Let* $\mathbf{C} \in \mathbb{C}^{3\times 3}$ *be a random complex matrix such as* $\mathbf{C}^*\mathbf{C} = \mathbf{I}$*. Then* $\phi(\mathbf{x}) = \mathbf{C}\mathbf{\Psi}(\mathbf{x})$*,* $M(\Delta\mathbf{x}) = \mathbf{C}diag(\mathbf{\Psi}(\Delta\mathbf{x}))\mathbf{C}^*$ *satisfies*

$$\phi(\mathbf{x} + \Delta\mathbf{x}) = M(\Delta\mathbf{x})\phi(\mathbf{x}) \tag{3.1}$$

and

$$\langle \phi(\mathbf{x} + \Delta\mathbf{x}), \phi(\mathbf{x})\rangle = d(1 - \alpha\|\Delta\mathbf{x}\|^2) \tag{3.2}$$

where $d = 3$ *is the dimension of* $\phi(\mathbf{x})$ *and* $\Delta\mathbf{x}$ *is a small displacement from* $\mathbf{x}$*.*

The proof of Theorem 1 can be seen in Gao et al. [107]. $\phi(\mathbf{x}) = \mathbf{C}\mathbf{\Psi}(\mathbf{x}) \in \mathbb{C}^3$ amounts to a 6-dimension real value vector and each dimension shows a *hexagon* firing pattern which models the grid cell behavior. Because of the periodicity of $sin()$ and $cos()$, this single scale representation $\phi(\mathbf{x})$ does not form a global codebook of 2D positions, i.e. there can be $\mathbf{x} \neq \mathbf{y}$, but $\phi(\mathbf{x}) = \phi(\mathbf{y})$.

Inspired by Theorem 1 and the multi-scale periodic representation of grid cells in mammals [102] we set up our point space encoder $\mathbf{e}[\mathbf{x}] = Enc^{(x)}_{theory}(\mathbf{x})$ to use sine and cosine functions of different frequencies to encode positions in space. Given any point $\mathbf{x}$ in the studied 2D space, the space encoder $Enc^{(x)}_{theory}(\mathbf{x}) = \mathbf{NN}(PE^{(t)}(\mathbf{x}))$ where $PE^{(t)}(\mathbf{x}) = [PE^{(t)}_0(\mathbf{x}); ...; PE^{(t)}_s(\mathbf{x}); ...; PE^{(t)}_{S-1}(\mathbf{x})]$ is a concatenation of multi-scale representations of $d^{(x)} = 6S$ dimensions. Here S is the total number of grid scales and $s = 0, 1, 2, ..., S - 1$. $\mathbf{NN}()$ represents fully connected ReLU layers. Let $\mathbf{a}_1 = [1, 0]^T, \mathbf{a}_2 = [-1/2, \sqrt{3}/2]^T, \mathbf{a}_3 =$

$[-1/2, -\sqrt{3}/2]^T \in \mathbb{R}^2$ be three unit vectors and the angle between any of them is $2\pi/3$. $\lambda_{min}, \lambda_{max}$ are the minimum and maximum grid scale and $g = \frac{\lambda_{max}}{\lambda_{min}}$. At each scale s, $PE^{(t)}_s(\mathbf{x}) = [PE^{(t)}_{s,1}(\mathbf{x}); PE^{(t)}_{s,2}(\mathbf{x}); PE^{(t)}_{s,3}(\mathbf{x})]$ is a concatenation of three components, where

$$PE^{(t)}_{s,j}(\mathbf{x}) = [\cos(\frac{\langle \mathbf{x}, \mathbf{a}_j \rangle}{\lambda_{min} \cdot g^{s/(S-1)}}); \sin(\frac{\langle \mathbf{x}, \mathbf{a}_j \rangle}{\lambda_{min} \cdot g^{s/(S-1)}})] \forall j = 1, 2, 3; \tag{3.3}$$

$\mathbf{NN}()$ and $PE^{(t)}(\mathbf{x})$ are analogies of $\mathbf{C}$ and $\mathbf{\Psi}(\mathbf{x})$ in Theorem 1.

Similarly we can define another space encoder $Enc^{(x)}_{grid}(\mathbf{x}) = \mathbf{NN}(PE^{(g)}(\mathbf{x}))$ inspired by the position encoding model of Transformer [104], where $PE^{(g)}(\mathbf{x}) = [PE^{(g)}_0(\mathbf{x}); ...; PE^{(g)}_s(\mathbf{x}); ...; PE^{(g)}_{S-1}(\mathbf{x})]$ is still a concatenation of its multi-scale representations, while $PE^{(g)}_s(\mathbf{x}) = [PE^{(g)}_{s,1}(\mathbf{x}); PE^{(g)}_{s,2}(\mathbf{x})]$ handles each component l of $\mathbf{x}$ separately:

$$PE^{(g)}_{s,l}(\mathbf{x}) = [\cos(\frac{\mathbf{x}^{[l]}}{\lambda_{min} \cdot g^{s/(S-1)}}); \sin(\frac{\mathbf{x}^{[l]}}{\lambda_{min} \cdot g^{s/(S-1)}})] \forall l = 1, 2 \tag{3.4}$$

3.4.2 Decoder

Two types of decoders are designed for two major types of GIS problems: location modeling and spatial context modeling (See Section 3.5.1).

Location Decoder $Dec_s()$ directly reconstructs point feature embedding $\mathbf{e}[\mathbf{v}_i]$ given its space embedding $\mathbf{e}[\mathbf{x}_i]$. We use one layer feed-forward neural network $\mathbf{NN}_{\text{dec}}()$

$$\mathbf{e}[\mathbf{v}_i]' = Dec_s(\mathbf{x}_i; \theta_{\text{dec}_s}) = \mathbf{NN}_{\text{dec}}(\mathbf{e}[\mathbf{x}_i]) \tag{3.5}$$

For training we use inner product to compare the reconstructed feature embedding $\mathbf{e}[\mathbf{v}_i]'$ against the real feature embeddings of $\mathbf{e}[\mathbf{v}_i]$ and other negative points (see training detail in Section 3.4.3).

Spatial Context Decoder $Dec_c()$ reconstructs the feature embedding $\mathbf{e}[\mathbf{v}_i]$ of the center point p_i based on the space and feature embeddings $\{\mathbf{e}_{i1}, ..., \mathbf{e}_{ij}, ..., \mathbf{e}_{in}\}$ of n nearby points $\{p_{i1}, ..., p_{ij}, ..., p_{in}\}$. Note that the feed-in order of context points should not affect the prediction results, which can be achieved by permutation invariant neural network architectures [115] like PointNet [95].

$$\mathbf{e}[\mathbf{v}_i]' = Dec_c(\mathbf{x}_i, \{\mathbf{e}_{i1}, ..., \mathbf{e}_{ij}, ..., \mathbf{e}_{in}\}; \theta_{\text{dec}_c}) = g(\frac{1}{K}\sum_{k=1}^{K}\sum_{j=1}^{n}\alpha_{ijk}\mathbf{e}[\mathbf{v}_{ij}]) \tag{3.6}$$

Here g is an activation function such as sigmoid. $\alpha_{ijk} = \frac{exp(\sigma_{ijk})}{\sum_{o=1}^{n} exp(\sigma_{iok})}$ is the attention of p_i with its jth neighbor through the kth attention head, and

$$\sigma_{ijk} = LeakyReLU(\mathbf{a}_k^T[\mathbf{e}[\mathbf{v}_i]_{init}; \mathbf{e}[\mathbf{v}_{ij}]; \mathbf{e}[\mathbf{x}_i - \mathbf{x}_{ij}]]) \tag{3.7}$$

where $\mathbf{a}_k \in \mathbb{R}^{2d^{(v)}+d^{(x)}}$ is the attention parameter in the kth attention head. The multi-head attention mechanism is inspired by Graph Attention Network [116] and Mai et al. [117].

To represent the spatial relationship (distance and direction) between each context point $p_{ij} = (\mathbf{x}_{ij}, \mathbf{v}_{ij})$ and the center point $p_i = (\mathbf{x}_i, \mathbf{v}_i)$, we use the space encoder $Enc^{(x)}()$ to encode the displacement between them $\Delta\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_{ij}$. Note that we are modeling the spatial interactions between the center point and n context points simultaneously.

In Equation 3.7, $\mathbf{e}[\mathbf{v}_i]_{init}$ indicates the initial guess of the feature embedding $\mathbf{e}[\mathbf{v}_i]$ of point p_i which is computed by using another multi-head attention layer as Equation 3.6 where the weight $\alpha'_{ijk} = \frac{exp(\sigma'_{ijk})}{\sum_{o=1}^{n} exp(\sigma'_{iok})}$. Here, σ'_{ijk} is computed as Equation 3.8 where the query embedding $\mathbf{e}[\mathbf{v}_i]$ is excluded.

$$\sigma'_{ijk} = LeakyReLU(\mathbf{a}_k'^T[\mathbf{e}[\mathbf{v}_{ij}]; \mathbf{e}[\mathbf{x}_i - \mathbf{x}_{ij}]]) \tag{3.8}$$

3.4.3 Unsupervised Training

The unsupervised learning task can simply be maximizing the log likelihood of observing the true point p_i at position $\mathbf{x}_i$ among all the points in $\mathcal{P}$

$$\mathcal{L}_{\mathcal{P}}(\theta) = -\sum_{p_i \in \mathcal{P}} \log P(p_i | p_{i1}, ..., p_{ij}, ..., p_{in}) = -\sum_{p_i \in \mathcal{P}} \log \frac{\exp(\mathbf{e}[\mathbf{v}_i]^T \mathbf{e}[\mathbf{v}_i]')}{\sum_{p_o \in \mathcal{P}} \exp(\mathbf{e}[\mathbf{v}_o]^T \mathbf{e}[\mathbf{v}_i]')} \tag{3.9}$$

Here only the feature embedding of p_i is used (without location embedding) to prevent revealing the identities of the point candidates, and $\theta = [\theta_{\text{enc}}; \theta_{\text{dec}}]$

Negative sampling by Mikolov et al. [91] can be used to improve the efficiency of training

$$\mathcal{L}'_{\mathcal{P}}(\theta) = -\sum_{p_i \in \mathcal{P}} \Big(\log \sigma(\mathbf{e}[\mathbf{v}_i]^T \mathbf{e}[\mathbf{v}_i]') + \frac{1}{|\mathcal{N}_i|} \sum_{p_o \in \mathcal{N}_i} \log \sigma(-\mathbf{e}[\mathbf{v}_o]^T \mathbf{e}[\mathbf{v}_i]') \Big) \tag{3.10}$$

Here $\mathcal{N}_i \subseteq \mathcal{P}$ is a set of sampled negative points for p_i ($p_i \notin \mathcal{N}_i$) and $\sigma(x) = 1/(1+e^{-x})$.

3.5 Experiment

In this section we compare Space2Vec with commonly used position encoding methods, and analyze them both quantitatively and qualitatively.

Baselines Our baselines include 1) *direct* directly applying feed-forward nets [100]; 2) *tile* discretization [98, 118, 99]; 3) *wrap* feed-forward nets with coordinate wrapping [64]; and 4) *rbf* Radial Basis Function (RBF) kernels [108, 109]. See Appendix 3.7.1 for details of the baselines.

3.5.1 POI Type Classification Tasks

Dataset and Tasks To test the proposed model, we conduct experiments on geographic datasets with POI position and type information. We utilize the open-source dataset published by Yelp Data Challenge and select all POIs within the Las Vegas downtown area[3]. There are 21,830 POIs with 1,191 different POI types in this dataset. Note that each POI may be associated with one or more types, and we do not use any other meta-data such as business names, reviews for this study. We project geographic

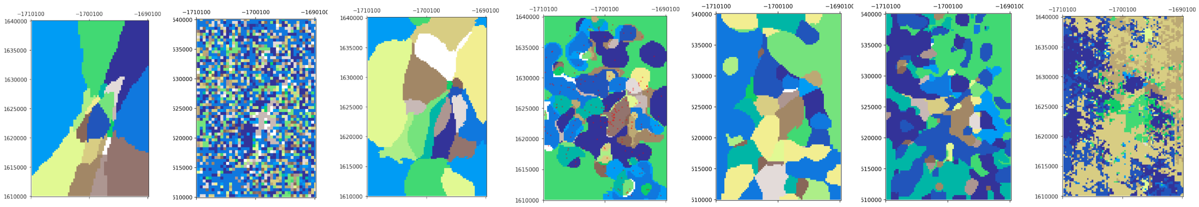
[3] The geographic range is (35.989438, 36.270897) for latitude and (-115.047977, -115.3290609) for longitude.

coordinates into projection coordinates using the NAD83/Conus Albers projection coordinate system[4]. The POIs are split into training, validation, and test dataset with ratios 80%:10%:10%. We create two tasks setups which represent different types of modeling need in Geographic Information Science:

- **Location Modeling** predicts the feature information associated with a POI based on its location $\mathbf{x}_i$ represented by the *location decoder* $Dec_s()$. This represents a large number of location prediction problems such as image fine grained recognition with geographic prior [100], and species potential distribution prediction [119].

- **Spatial Context Modeling** predicts the feature information associated with a POI based on its context $\{\mathbf{e}_{i1}, ..., \mathbf{e}_{ij}, ..., \mathbf{e}_{in}\}$ represented by the *spatial context decoder* $Dec_c()$. This represents a collections of spatial context prediction problem such as spatial context based facade image classification [71], and all spatial interpolation problems.

We use POI prediction metrics to evaluate these models. Given the real point feature embedding $\mathbf{e}[\mathbf{v}_i]$ and N negative feature embeddings $\mathcal{N}_i = \{\mathbf{e}[\mathbf{v}_i]^-\}$, we compare the predicted $\mathbf{e}[\mathbf{v}_i]'$ with them by cosine distance. The cosine scores are used to rank $\mathbf{e}[\mathbf{v}_i]$ and N negative samples. The negative feature embeddings are the feature embeddings of points p_j randomly sampled from $\mathcal{P}$ and $p_i \neq p_j$. We evaluate each model using Negative Log-Likelihood (NLL), Mean Reciprocal Rank (MRR) and HIT@5 (the chance of the true POI being ranked to top 5. We train and test each model 10 times to estimate standard deviations. See Appendix 3.7.2 for hyper-parameter selection details.



(a) *direct* (b) *tile* (c) *wrap* (d) *rbf* (σ=1k) (e) λ_{min}=1k (f) λ_{min}=500 (g) λ_{min}=50

Figure 3.2: Embedding clustering of (a) *direct*; (b) *tile* with the best cell size $c = 500$; (c) *wrap* ($h = 3, o = 512$); (d) *rbf* with the best σ (1k) and 200 anchor points (red) and (e)(f)(h) *theory* models with different λ_{min}, but fixed $\lambda_{max} = 40k$ and $S = 64$. All models use 1 hidden ReLU layers of 512 neurons except *wrap*.

Location Modeling Evaluation

We first study location modeling with the **location decoder** $Dec_s()$ in Section 3.4.2. We use a negative sample size of $N = 100$. Table 3.1 shows the average metrics of different models with their best hyper-parameter setting on the validation set. We can see that *direct* and *theorydiag* are less competitive, only beating the *random* selection baseline. Other methods with single scale representations – including *tile*, *wrap*, and *rbf* – perform better. The best results come from various version of the grid cell models, which are capable of dealing with multi-scale representations.

In order to understand the reason for the superiority of grid cell models we provide qualitative analysis of their representations. We apply hierarchical clustering to the location embeddings produced by studied models using cosine distance as the distance metric (See Figure 3.2). we can see that when restricted to large grid sizes ($\lambda_{min} = 1k$), *theory* has similar representation (Figure 3.2d, 3.2e, and Figure 3.4d, 3.4e) and performance compared to *rbf* ($\sigma = 1k$). However it is able to significantly outperform *rbf* ($\sigma = 1k$) (and *tile* and *wrap*) when small grid sizes ($\lambda_{min} = 500, 50$) are available. The relative improvements over *rbf* ($\sigma = 1k$) are -0.2%, +0.6%, +2.1% MRR for λ_{min}=1k, 500, 50 respectively.

[4] `https://epsg.io/5070-1252`

Table 3.1: The evaluation results of different location models on the validation and test dataset.

	Train	Validation			Testing	
	NLL	NLL	MRR	HIT@5	MRR	HIT@5
random		-	0.052 (0.002)	4.8 (0.5)	0.051 (0.002)	5.0 (0.5)
direct	1.285	1.332	0.089 (0.001)	10.6 (0.2)	0.090 (0.001)	11.3 (0.2)
tile (c=500)	1.118	1.261	0.123 (0.001)	16.8 (0.2)	0.120 (0.001)	17.1 (0.3)
wrap(h=3,o=512)	1.222	1.288	0.112 (0.001)	14.6 (0.1)	0.119 (0.001)	15.8 (0.2)
rbf (σ=1k)	1.209	1.279	0.115 (0.001)	15.2 (0.2)	0.123 (0.001)	16.8 (0.3)
grid (λ_{min}=50)	1.156	1.258	0.128 (0.001)	18.1 (0.3)	0.139 (0.001)	**20.0** (0.2)
hexa (λ_{min}=50)	1.230	1.297	0.107 (0.001)	14.0 (0.2)	0.105 (0.001)	14.5 (0.2)
theorydiag (λ_{min}=50)	1.277	1.324	0.094 (0.001)	12.3 (0.3)	0.094 (0.002)	11.2 (0.3)
theory (λ_{min}=1k)	1.207	1.281	0.123 (0.002)	16.3 (0.5)	0.121 (0.001)	16.2 (0.1)
theory (λ_{min}=500)	1.188	1.269	0.132 (0.001)	17.6 (0.3)	0.129 (0.001)	17.7 (0.2)
theory (λ_{min}=50)	1.098	1.249	**0.137** (0.002)	**19.4** (0.1)	**0.144** (0.001)	**20.0** (0.2)

Multi-Scale Analysis of Location Modeling

In order to show how our multi-scale location representation model will affect the prediction of POI types with different distribution patterns, we classify all 1,191 POI types into three groups based on radius r, which is derived from each POI types' renormalized Ripley's K curve (See Figure 3.1d for examples). It indicates the x axis value of the intersection between the curve and the line of $y = 3.0$. A lower r indicates a more clustered distribution patterns. These three groups are listed below:

1. Clustered ($r \leq 100m$): POI types with clustered distribution patterns;
2. Middle ($100m < r < 200m$): POI types with less extreme scales;
3. Even ($r \geq 200m$): POI types with even distribution patterns.

Table 3.2 shows the performance (MRR) of *direct*, *tile*, *wrap*, *rbf*, and our *theory* model on the test dataset of the location modeling task with respect to these three different POI distribution groups. The numbers in () indicate the MRR difference betweeb a baseline and *theory*. *# POI* refers to total number of POI belong to each group[5]. We can see that 1) The two neural net approaches (*direct* and *wrap*) have no scale related parameter and are not performing ideally across all scales, with *direct* performs worse because of its simple single layer network. 2) The two approaches with built-in scale parameter (*tile* and *rbf*) have to trade off the performance of different scales. Their best parameter settings lead to close performances to that of Space2Vec at the middle scale, while performing poorly in both clustered and regular groups. These observation clearly shows that **all baselines can at most well handle distribution at one scale but show poor performances in other scales. In contrast, Space2Vecs multi-scale representation can handle distributions at different scales.**

Spatial Context Modeling Evaluation

Next, we evaluate the **spatial context decoder** $Dec_c()$ in Section 3.4.2. We use the same evaluation set up as location modeling. The context points are obtained by querying the n-th nearest points using PostGIS ($n = 10$). As for validation and test datasets, we make sure the center points are all unknown during the training phase. Table 3.3 shows the evaluation results of different models for spatial context modeling. The baseline approaches (*direct*, *tile*, *wrap*, *rbf*) generally perform poorly in context modeling. We designed specialized version of these approaches (*polar*, *polar_tile*, *scaled_rbf*) with polar coordinates, which lead to significantly improvements. Note that these are models proposed by us specialized for context modeling and therefore are less general than the grid cell approaches. Nevertheless the grid cell approaches are able to perform better

[5] The reason why the sum of *# POI* of these three groups does not equal to the total number of POI is because one POI can have multiple types and they may belonging to different groups.

Table 3.2: Comparing performances in different POI groups. We classify all 1,191 POI types into three groups based on the radius r of their root types, where their renormalized Ripley's K curve (See Figure 3.1d) reach 3.0: 1) Clustered ($r \leq 100m$): POI types with clustered distribution patterns; 2) Middle ($100m < r < 200m$): POI types with unclear distribution patterns; 3) Even ($r \geq 200m$): POI types with even distribution patterns. The MRR of *wrap* and *theory* on those three groups are shown. The numbers in () indicate the difference between the MRR of a baseline model and the MRR of *theory* with respect to a specific group. *#POI* refers to the total number of POIs belonging to each group. *Root Types* indicates the root categories of those POI types belong to each group.

POI Groups	**Clustered** ($r \leq 100m$)	**Middle** ($100m < r < 200m$)	**Even** ($r \geq 200m$)
direct	0.080 (-0.047)	0.108 (-0.030)	0.084 (-0.047)
wrap	0.106 (-0.021)	0.126 (-0.012)	0.122 (-0.009)
tile	0.108 (-0.019)	0.135 (-0.003)	0.111 (-0.020)
rbf	0.112 (-0.015)	0.136 (-0.002)	0.119 (-0.012)
theory	0.127 (-)	0.138 (-)	0.131 (-)
# POI	16,016	7,443	3,915
Root Types	Restaurants; Shopping; Food; Nightlife; Automotive; Active Life; Arts & Entertainment; Financial Services	Beauty & Spas; Health & Medical; Local Services; Hotels & Travel; Professional Services; Public Services & Government	Home Services; Event Planning & Services; Pets; Education

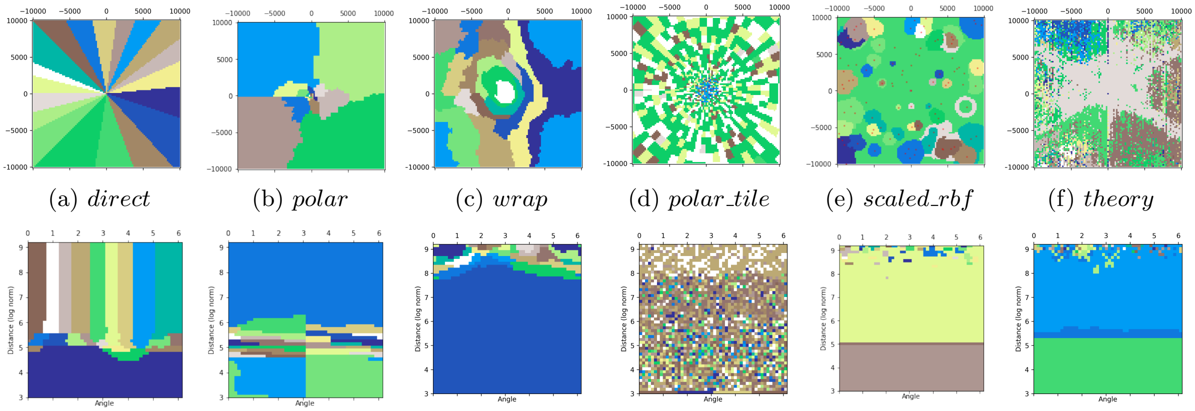
than the specialized approaches on the test dataset while have competitive performance on validation dataset. See Figure 3.3 for the visualization of context models. Actually the gains are small for all baseline approaches also. The reason is that we expect location encoding to be less important when context information is accessible. Similarly as discussed in [107], it is when there is a lack of visual clues that the grid cells of animals are the most helpful for their navigation.

Figure 3.3 shows the location embedding clustering results in both Cartesian and polar coordinate systems. We can see that *direct* (Figure 3.3a, 3.3g) only captures the distance information when the context POI is very close ($log(\parallel \Delta \mathbf{x}_{ij} \parallel +1) \leq 5$) while in the farther spatial context it purely models the direction information. *polar* (Figure 3.3b, 3.3h) has the similar behaviors but captures the distance information in a more fine-grained manner. *wrap* (Figure 3.3c, 3.3i) mainly focuses on differentiating relative positions in farther spatial context *cont* which might explain its lower performance[6]. *polar_tile* (Figure 3.3d) mostly responds to distance information. Interestingly, *scaled_rbf* and *theory* have similar representations in the polar coordinate system (Figure 3.3k, 3.3l) and similar performance (Table 3.3). While *scaled_rbf* captures the gradually decreased distance

[6] Note that *wrap* is original proposed by Mac Aodha et al. [64] for location modelling, not spatial context modelling. This results indicates *wrap* is not good at this task.

Table 3.3: The evaluation results of different spatial context models on the validation and test dataset. All encoders contains a 1 hidden layer FFN. All grid cell encoders set λ_{min}=10, λ_{max}=10k.

	Train	Validation			Testing	
Space2Vec	NLL	NLL	MRR	HIT@5	MRR	HIT@5
none	1.163	1.297	0.159 (0.002)	22.4 (0.5)	0.167 (0.006)	23.4 (0.7)
direct	1.151	1.282	0.170 (0.002)	24.6 (0.4)	0.175 (0.003)	24.7 (0.5)
polar	1.157	1.283	0.176 (0.004)	25.4 (0.4)	0.178 (0.006)	24.9 (0.1)
tile ($c = 50$)	1.163	1.298	0.173 (0.004)	24.0 (0.6)	0.173 (0.001)	23.4 (0.1)
polar_tile($S = 64$)	1.161	1.282	0.173 (0.003)	25.0 (0.1)	0.177 (0.001)	24.5 (0.3)
wrap (h=2,o=512)	1.167	1.291	0.159 (0.001)	23.0 (0.1)	0.170 (0.001)	23.9 (0.2)
rbf ($\sigma = 50$)	1.160	1.281	**0.179** (0.002)	25.2 (0.6)	0.172 (0.001)	25.0 (0.1)
scaled_rbf (σ=40,β=0.1)	1.150	1.272	0.177 (0.002)	**25.7** (0.1)	0.181 (0.001)	25.3 (0.1)
grid(λ_{min}=10)	1.172	1.285	0.178 (0.004)	24.9 (0.5)	0.181 (0.001)	25.1 (0.3)
hexa (λ_{min}=10)	1.156	1.289	0.173 (0.002)	24.0 (0.2)	0.183 (0.002)	25.3 (0.2)
theorydiag ($\lambda_{min} = 10$)	1.156	1.287	0.168 (0.001)	24.1 (0.4)	0.174 (0.005)	24.9 (0.1)
theory(λ_{min}=200)	1.168	1.295	0.159 (0.001)	23.1 (0.2)	0.170 (0.001)	23.2 (0.2)
theory(λ_{min}=50)	1.157	1.275	0.171 (0.001)	24.2 (0.3)	0.173 (0.001)	24.8 (0.4)
theory(λ_{min}=10)	1.158	1.280	0.177 (0.003)	25.2 (0.3)	**0.185** (0.002)	**25.7** (0.3)



(a) *direct* (b) *polar* (c) *wrap* (d) *polar_tile* (e) *scaled_rbf* (f) *theory*

(g) *direct* (h) *polar* (i) *wrap* (j) *polar_tile* (k) *scaled_rbf* (l) *theory*

Figure 3.3: Embedding clustering in the original space of (a) *direct*; (b) *polar*; (c) *wrap*, h=2,o=512; (d) *polar_tile*, $S = 64$, (e) *scaled_rbf*, $\sigma = 40$, β=0.1; and (f) *theory*, $\lambda_{min} = 10$, $\lambda_{max} = 10k$, $S = 64$. (g)(h)(i)(j)(k)(l) are the clustering results of the same models in the polar-distance space using $\log(\| \Delta\mathbf{x}_{ij} \| + 1)$. All models use 1 hidden ReLU (except *wrap*) layers of 512 neurons. Most models except *wrap* can capture a shift when distance is around $e^5 - 1 \approx 150$ meters.

effect with a scaled kernel size which becomes larger in farther distance, *theory* achieves this by integrating representations of different scales.

3.5.2 Fine-Grained Image Classification Tasks

To demonstrate the generalizability of Space2Vec for space representation we utilized the proposed point space encoder $Enc^{(x)}()$ model in a well-known computer vision task: *fine-grained image classification*. As we discussed in Section 3.3, many studies [98, 100, 64] have shown that geographic prior information - where (and when) the image is taken - is very important additional information for the fine-grained image classification task and can substantially improve the model performance. For example, the appearance information is usually not sufficient to differentiate two visually similar species. In this case, the geographic prior becomes much more important because these two species may have very different spatial prior distributions such as the example of European Toads and Spiny Toads in Figure 1 of Mac Aodha et al. [64].

We adopt the task setup of Mac Aodha et al. [64]. During training we have a set of tuples $D = \{(I_i, \mathbf{x}_i, y_i, p_i) \mid i = 1, ..., N\}$ where I_i indicates an image, $y_i \in \{1, 2, ..., C\}$ is the corresponding class label (species category), $\mathbf{x}_i = [longitude_i, latitude_i]$ is the geographic coordinates where the image was taken, and p_i is the id of the photographer who took this image. At training time, a location encoder is trained to capture the spatial prior information $P(y \mid \mathbf{x})$. At inference time, p_i information is not available and the final image classification prediction is calculated based on the combination of two models: 1) the trained location encoder which captures the spatial priors $P(y \mid \mathbf{x})$ and 2) the pretrained image classification model, InceptionV3 network [120] which captures $P(y \mid I)$. Bayesian theory has been used to derive the joint distribution $P(y \mid I, \mathbf{x})$. See Mac Aodha et al. Mac Aodha et al. [64] for detail explanation as well as the loss function. Note that while Space2Vec outperforms specialized density estimation methods such as Adaptive Kernel [98], it would be interesting to explore early fusion Space2Vec 's representations with the image module.

We use two versions of our point space encoder $Enc^{(x)}()$ model (*grid*, *theory*) as the location encoder to capture the spatial prior information $P(y \mid \mathbf{x})$. The evaluation results of our models as well as multiple baselines are shown in Table 3.4. We can see that **both *grid*, *theory* outperform previous models as well as that of Mac Aodha et al. [64] on two fine-grained image classification datasets with significant sizes: BirdSnap†, NABirds†**. *theory* shows superiority over *grid* on NABirds† while fail to outperform *grid* on BirdSnap†. Note that we only pick baseline models which capture spatial-only prior and drop models which additionally consider time information. Both *grid* and *theory* use 1 hidden ReLU layers of 512 neurons for $\mathbf{NN}()$ and they have the same hyperparameters: λ_{min}=0.0001, λ_{max}=360, $S = 64$. Like Mac Aodha et al. [64], the location embedding size $d^{(x)}$ is 1024 and we train the location encoder for 30 epochs.

Table 3.4: Fine-grained image classification results on two datasets: BirdSnap† and NABirds†. The classification accuracy is calculated by combining image classification predictions $P(y \mid I)$ with different spatial priors $P(y \mid \mathbf{x})$. The *grid* and *theory* model use 1 hidden ReLU layers of 512 neurons. The evaluation results of the baseline models are from Table 1 of Mac Aodha et al. [64].

	BirdSnap†	NABirds†
No Prior (i.e. uniform)	70.07	76.08
Nearest Neighbor (num)	77.76	79.99
Nearest Neighbor (spatial)	77.98	80.79
Adaptive Kernel [98]	78.65	81.11
tile [99] (location only)	77.19	79.58
wrap [64] (location only)	78.65	81.15
rbf (σ=1k)	78.56	81.13
grid (λ_{min}=0.0001, λ_{max}=360, S = 64)	**79.44**	81.28
theory (λ_{min}=0.0001, λ_{max}=360, S = 64)	79.35	**81.59**

Our implementation is based on the original code[7] of Mac Aodha et al. [64] for both model training and evaluation phase.

3.6 Conclusion

We introduced an encoder-decoder framework as a general-purpose representation model for space inspired by biological grid cells' multi-scale periodic representations. The model is an inductive learning model and can be trained in an unsupervised manner. We conduct two experiments on POI type prediction based on 1) POI locations and 2) nearby POIs. The evaluation results demonstrate the effectiveness of our model. Our analysis reveals that it is the ability to integrate representations of different scales that makes the grid cell models outperform other baselines on these two tasks. In the future, we hope to

[7] `https://github.com/macaodha/geo_prior/`

incorporate the presented framework to more complex GIS tasks such as social network analysis, and sea surface temperature prediction.

Acknowledgments

The presented work is partially funded by the NSF award 1936677 C-Accel Pilot - Track A1 (Open Knowledge Network): *Spatially-Explicit Models, Methods, And Services For Open Knowledge Networks*, Esri Inc., and Microsoft AI for Earth Grant: *Deep Species Spatio-temporal Distribution Modeling for Biodiversity Hotspot Prediction.* We thank Dr. Ruiqi Gao for discussions about grid cells, Dr. Wenyun Zuo for discussion about species potential distribution prediction and Dr. Yingjie Hu for his suggestions about the introduction section.

3.7 Appendix

3.7.1 Baselines

To help understand the mechanism of distributed space representation we compare multiple ways of encoding spatial information. Different models use different point space encoder $Enc^{(x)}()$ to encode either location $\mathbf{x}_i$ (for location modeling *loc*) or the displacement between the center point and one context point $\Delta\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_{ij}$ (for spatial context modeling *cont*)[8].

- *random* shuffles the order of the correct POI and N negative samples randomly as the predicted ranking. This shows the lower bound of each metrics.

- *direct* directly encode location $\mathbf{x}_i$ (or $\Delta\mathbf{x}_{ij}$ for *cont*) into a location embedding $\mathbf{e}[\mathbf{x}_i]$ (or $\mathbf{e}[\Delta\mathbf{x}_{ij}]$) using a feed-forward neural networks (FFNs)[9], denoted as $Enc^{(x)}_{direct}(\mathbf{x})$ without decomposing coordinates into a multi-scale periodic representation. This is essentially the GPS encoding method used by Chu et al. [100]. Note that Chu et al. [100] is not open sourced and we end up implementing the model architecture ourselves.

- *tile* divides the study area A_{loc} (for *loc*) or the range of spatial context defined by λ_{max}, A_{cont}, (for *cont*) into grids with equal grid sizes c. Each grid has an embedding to be used as the encoding for every location $\mathbf{x}_i$ or displacement $\Delta\mathbf{x}_{ij}$ fall into this grid. This is a common practice by many previous work when dealing with coordinate data [98, 118, 99].

- *wrap* is a location encoder model recently introduced by Mac Aodha et al. [64]. It first normalizes $\mathbf{x}$ (or $\Delta\mathbf{x}$) into the range $[-1, 1]$ and uses a coordinate wrap

[8] We will use meter as the unit of $\lambda_{min}, \lambda_{max}, \sigma, c$.

[9] we first normalizes $\mathbf{x}$ (or $\Delta\mathbf{x}$) into the range $[-1, 1]$

mechanism $[\sin(\pi \mathbf{x}^{[l]}); \cos(\pi \mathbf{x}^{[l]})]$ to convert each dimension of $\mathbf{x}$ into 2 numbers. This is then passed through an initial fully connected layer, followed by a series of h residual blocks, each consisting of two fully connected layers (o hidden neurons) with a dropout layer in between. We adopt the official code of Mac Aodha et al. [64][10] for this implementation.

- *rbf* randomly samples M points from the training dataset as RBF anchor points $\{\mathbf{x}_m^{anchor}, m = 1...M\}$ (or samples M $\Delta\mathbf{x}_m^{anchor}$ from A_{cont} for *cont*) [11], and use gaussian kernels $\exp\big(-\dfrac{\parallel \mathbf{x}_i - \mathbf{x}_m^{anchor} \parallel^2}{2\sigma^2}\big)$ (or $\exp\big(-\dfrac{\parallel \Delta\mathbf{x}_{ij} - \Delta\mathbf{x}_m^{anchor} \parallel^2}{2\sigma^2}\big)$ for *cont*) on each anchor points, where σ is the kernel size. Each point p_i has a M-dimension RBF feature vector which is fed into a FNN to obtain the spatial embedding. This is a strong baseline for representing floating number features in machine learning models.

- *grid* as described in Section 3.4.1 inspired by the position encoding in Transformer [104].

- *hexa* Same as *grid* but use $sin(\theta)$, $sin(\theta + 2\pi/3)$, and $sin(\theta + 4\pi/3)$ in $PE_{s,l}^{(g)}(\mathbf{x})$.

- *theory* as described in Section 3.4.1, uses the theoretical models [107] as the first layer of $Enc_{theory}^{(x)}(\mathbf{x})$ or $Enc_{theory}^{(x)}(\Delta\mathbf{x}_{ij})$.

- *theorydiag* further constrains $\mathbf{NN}()$ as a block diagonal matrix, with each scale as a block.

We also have the following baselines which are specific to the spatial context modeling task.

[10] `http://www.vision.caltech.edu/~macaodha/projects/geopriors/`

[11] these anchor points are fixed in both *loc* and *cont*.

- *none* the decoder $Dec_c()$ does not consider the spatial relationship between the center point and context points but only the co-locate patterns such as Place2Vec [70]. That means we drop the $\mathbf{e}[\Delta\mathbf{x}_{ij}]$ from the attention mechanism in Equation 3.7 and 3.8.

- *polar* first converts the displacement $\Delta\mathbf{x}_{ij}$ into polar coordinates (r, θ) centered at the center point where $r = log(\parallel \Delta\mathbf{x}_{ij} \parallel +1)$. Then it uses $[r, \theta]$ as the input for a FFN to obtain the spatial relationship embedding in Equation 3.7. We find out that it has a significant performance improvement over the variation with $r =\parallel \Delta\mathbf{x}_{ij} \parallel$.

- *polar_tile* is a modified version of *tile* but the grids are extracted from polar coordinates (r, θ) centered at the center point where $r = log(\parallel \Delta\mathbf{x}_{ij} \parallel +1)$. Instead of using grid size c, we use the number of grids along θ (or r) axis, F, as the only hyperparameter. Similarly, We find that $r = log(\parallel \Delta\mathbf{x}_{ij} \parallel +1)$ outperform $r =\parallel \Delta\mathbf{x}_{ij} \parallel$ significantly.

- *scaled_rbf* is a modified version of rbf for *cont* whose kernel size is proportional to the distance between the current anchor point and the origin, $\parallel \Delta\mathbf{x}_m^{anchor} \parallel$. That is $\exp\big(-\frac{\parallel \Delta\mathbf{x}_{ij} - \Delta\mathbf{x}_m^{anchor} \parallel^2}{2\sigma_{scaled}^2}\big)$. Here $\sigma_{scaled} = \sigma + \beta \parallel \Delta\mathbf{x}_m^{anchor} \parallel$ where σ is the basic kernel size and β is kernel rescale factor, a constant. We developed this mechanism to help RFB to deal with relations at different scale, and we observe that it produces significantly better result than vanilla RBFs.

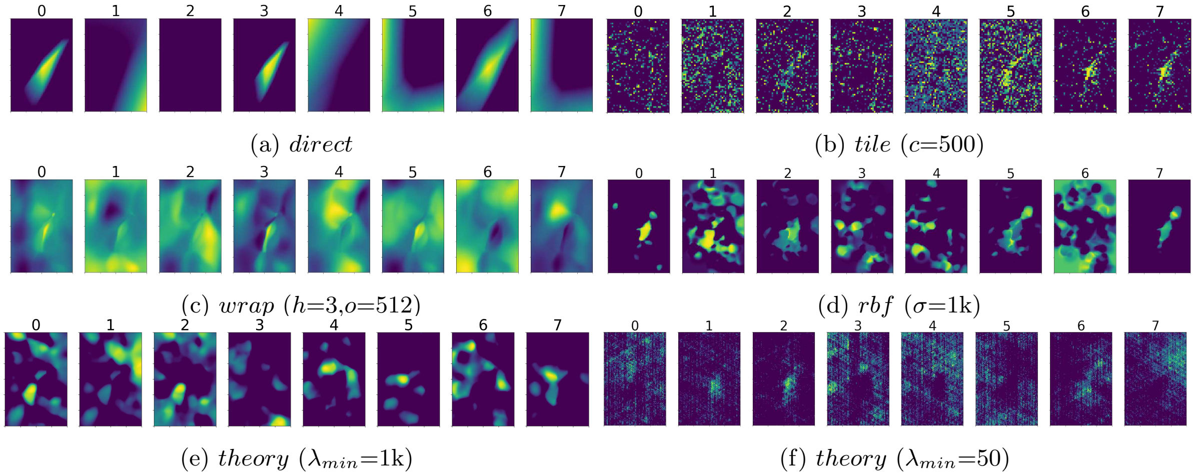
3.7.2 Hyper-Parameter Selection

We perform grid search for all methods based on their performance on the validation sets.

Location Modeling The hyper-parameters of *theory* models are based on grid search with $d^{(v)} = (32, 64, 128, 256)$, $d^{(x)} = (32, 64, 128, 256)$, $S = (4, 8, 16, 32, 64, 128)$, and $\lambda_{min} = (1, 5, 10, 50, 100, 200, 500, 1k)$ while $\lambda_{max} = 40k$ is decided based on the total size of the study area. We find out the best performances of different grid cell based models are obtained when $d^{(v)} = 64$, $d^{(x)} = 64$, $S = 64$, and $\lambda_{min} = 50$. In terms of *tile*, the hyper-parameters are selected from $c = (10, 50, 100, 200, 500, 1000)$ while $c = 500$ gives us the best performance.As for rbf, we do grid search on the hyper-parameters: $M = (10, 50, 100, 200, 400, 800)$ and $\sigma = (10^2, 10^3, 10^4, 10^5, 10^6, 10^7)$. The best performance of rbf is obtain when $M = 200$ and $\sigma = 10^3$. As for *wrap*, grid search is performed on: $h = (1, 2, 3, 4)$ and $o = (64, 128, 256, 512)$ while $h = 3$ and $o = 512$ gives us the best result. All models use FFNs in their $Enc^{(x)}()$ except *wrap*. The number of layers f and the number of hidden state neurons u of the FFN are selected from $f = (1, 2, 3)$ and $u = (128, 256, 512)$. We find out $f = 1$ and $u = 512$ give the best performance for *direct*, *tile*, rbf, and *theory*. So we use them for every model for a fair comparison.

Spatial Context Modeling Grid search is used for hyperparameter tuning and the best performance of different grid cell models is obtain when $d^{(v)} = 64$, $d^{(x)} = 64$, $S = 64$, and $\lambda_{min} = 10$. We set $\lambda_{max} = 10k$ based on the maximum displacement between context points and center points to make the location encoding unique. As for multiple baseline models, grid search is used again to obtain the best model. The best model hyperparameters are shown in () besides the model names in Table 3.3. Note that both rbf and *scaled_rbf* achieve the best performance with $M = 100$.

3.7.3 Firing Pattern for the Neurons



(a) *direct*

(b) *tile* (c=500)

(c) *wrap* (h=3,o=512)

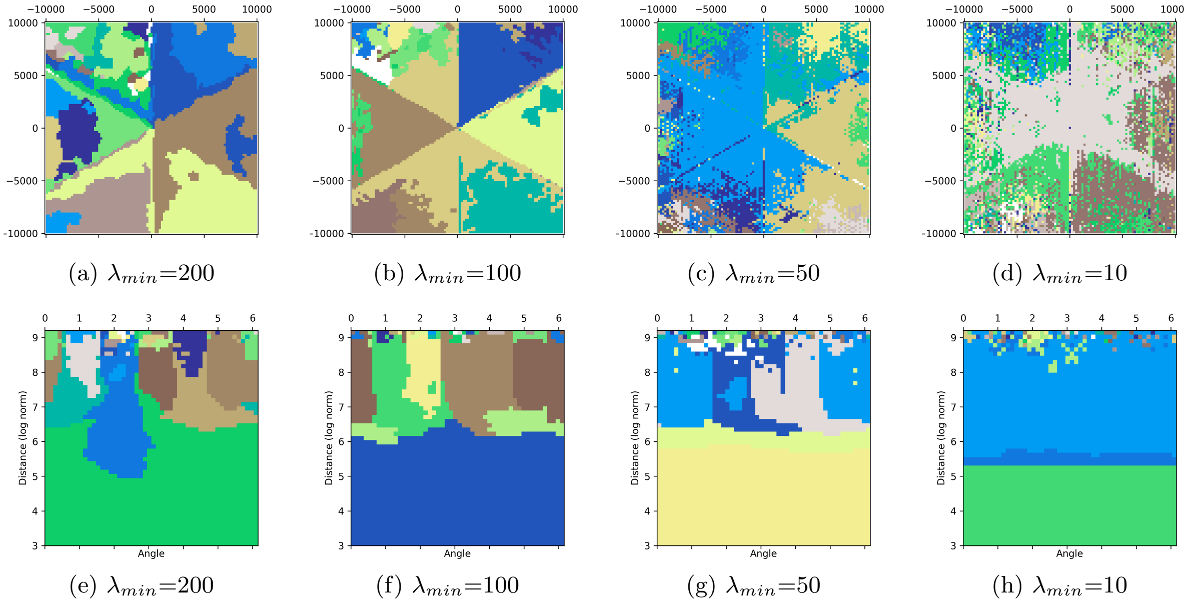
(d) *rbf* (σ=1k)

(e) *theory* (λ_{min}=1k)

(f) *theory* (λ_{min}=50)

Figure 3.4: The firing pattern for the first 8 neurons (out of 64) given different encoders in location modeling.

3.7.4 Embedding clustering of RBF and theory models



(a) λ_{min}=200

(b) λ_{min}=100

(c) λ_{min}=50

(d) λ_{min}=10

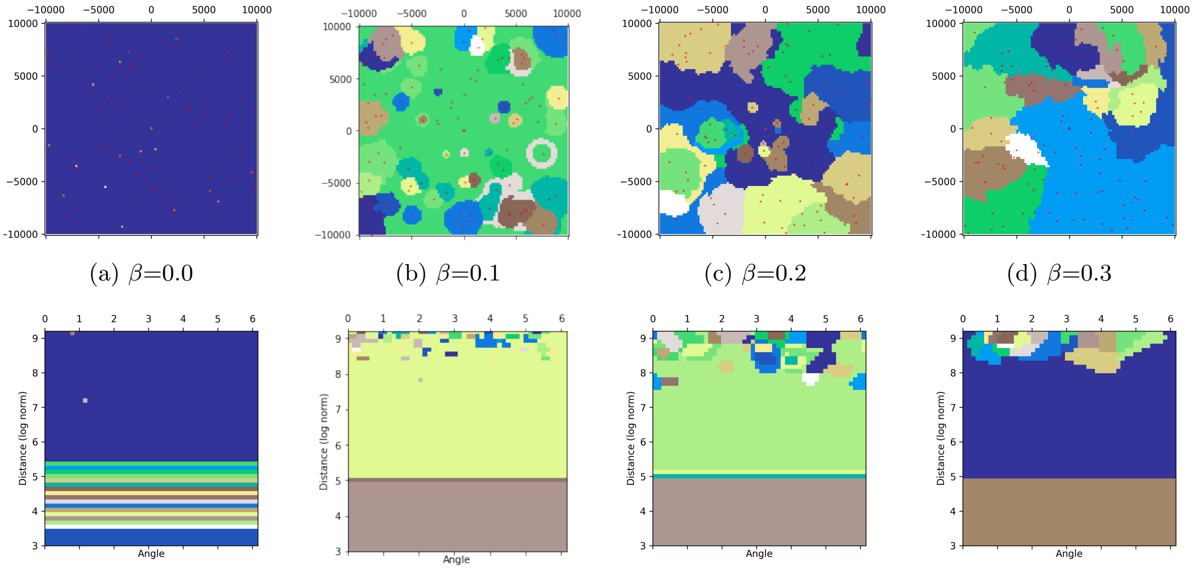
(e) λ_{min}=200

(f) λ_{min}=100

(g) λ_{min}=50

(h) λ_{min}=10

Figure 3.5: Embedding clustering in the original space of (a)(b)(c)(d) *theory* with different λ_{min}, but the same $\lambda_{max} = 10k$ and $S = 64$. (e)(f)(g)(h) are the embedding clustering results of the same models in the polar-distance space. All models use 1 hidden ReLU layers of 512 neurons.



(a) β=0.0

(b) β=0.1

(c) β=0.2

(d) β=0.3

(e) β=0.0

(f) β=0.1

(g) β=0.2

(h) β=0.3

Figure 3.6: Embedding clustering of RBF models with different kernel rescalar factor β (a)(b)(c)(d) in the original space; (e)(f)(g)(h) in the polar-distance space. Here β=0.0 indicates the original RBF model. All models use σ=10m as the basic kernel size and 1 hidden ReLU layers of 512 neurons.

Chapter 4

SE-KGE: A location-aware Knowledge Graph Embedding Model for Geographic Question Answering and Spatial Semantic Lifting

This chapter focuses on utilizing the location encoder - Space2Vec - proposed in Chapter 3 to develop a knowledge graph-based geographic query/question answering system. The core idea is to integrate this location encoder into a knowledge graph embedding (KGE) based logic query answering framework such that the whole GeoQA model is location-aware. There is no existing knowledge graph embedding model that can encode geographic information into the embedding space. This work fills this gap by developing a location-aware KGE model - *SE-KGE* - which directly encodes point coordinates of small-scale geographic entities (or bounding boxes of large-scale geographic entities) into the KG embedding space. The effectiveness of *SE-KGE* has been demonstrated through two KG-based geospatial tasks - geographic logic query answering and spatial semantic lifting. Evaluation results on our newly constructed geographic knowledge graph and GeoQA dataset *DBGeo* show that our *SE-KGE* can outperform multiple baseline models on both tasks.

Peer Reviewed Publication	
Title	SE−KGE: A location-aware Knowledge Graph Embedding model for Geographic Question Answering and Spatial Semantic Lifting
Authors	Gengchen Mai, Krzysztof Janowicz, Ling Cai, Rui Zhu, Blake Regalia, Bo Yan, Meilin Shi, Ni Lao
Venue	Transactions in GIS
Editors	John P. Wilson
Publisher	Wiley
Pages	623-655
Submit Date	February 15, 2020
Accepted Date	April 1, 2020
Publication Date	May 30, 2020
Copyright	Reprinted with permission from Wiley
DOI	https://doi.org/10.1111/tgis.12629

Abstract: Learning knowledge graph (KG) embeddings is an emerging technique for a variety of downstream tasks such as summarization, link prediction, information retrieval, and question answering. However, most existing KG embedding models neglect space and, therefore, do not perform well when applied to (geo)spatial data and tasks. For those models that consider space, most of them primarily rely on some notions of distance. These models suffer from higher computational complexity during training

while still losing information beyond the relative distance between entities. In this work, we propose a location-aware KG embedding model called *SE-KGE*. It directly encodes spatial information such as point coordinates or bounding boxes of geographic entities into the KG embedding space. The resulting model is capable of handling different types of spatial reasoning. We also construct a geographic knowledge graph as well as a set of geographic query-answer pairs called *DBGeo* to evaluate the performance of *SE-KGE* in comparison to multiple baselines. Evaluation results show that *SE-KGE* outperforms these baselines on the *DBGeo* dataset for geographic logic query answering task. This demonstrates the effectiveness of our spatially-explicit model and the importance of considering the scale of different geographic entities. Finally, we introduce a novel downstream task called *spatial semantic lifting* which links an arbitrary location in the study area to entities in the KG via some relations. Evaluation on *DBGeo* shows that our model outperforms the baseline by a substantial margin.

4.1 Introduction and Motivation

The term *Knowledge Graph* typically refers to a labeled and directed multi-graph of statements (called triples) about the world. These triples often originate from heterogeneous sources across domains. According to Nickel et al. [121], most of the widely used knowledge graphs are constructed in a curated (e.g., WordNet), collaborative (e.g., Wikidata, Freebase), or auto semi-structured (e.g., YAGO [122], DBpedia, Freebase) fashion rather than an automated unstructured approach (e.g., Knowledge Vault [123]). Despite containing billions of statements, these knowledge graphs suffer from *incompleteness and sparsity* [124, 123, 117]. To address these problems, many relational machine learning models [121] have been developed for knowledge graph completion tasks including several embedding-based techniques such as RESCAL [125], TransE [85], TransH [86], HOLE [126], R-GCN [127], and TransGCN [128]. The key idea of the embedding-based technique [85, 86, 126, 128, 89] is to project entities and relations in a knowledge graph onto a continuous vector space such that entities and relations can be quantitatively represented as vectors/embeddings. In such a way, we can evaluate the plausibility of a statement/triple in a KG.

The aforementioned incompleteness and sparsity problems also affect the performance of downstream tasks such as question answering [82] since missing triples or links result in certain questions becoming unanswerable [80]. Consequently, researchers have recently focused on relaxing these unanswerable queries or predicting the most probable answers based on knowledge graph embedding models [82, 83, 47].

Most research on knowledge graph embeddings has neglected spatial aspects such as the location of geographic entities despite the important role such entities play within knowledge graphs [129]. In fact, most of the current knowledge graph embedding models (e.g. TransE, TransH, TransGCN, R-GCN, and HOLE) ignore triples that contain *datatype properties*, and, hence, literals for dates, texts, numbers, geometries, and so forth. Put differently, properties such as `dbo:elevation`, `dbo:populationTotal`, and `dbo:areaWater` to name but a few are not considered during the training phase. Instead, these models strictly focus on triples with *object type properties*, leading to substantial information loss in practice. A few models do consider a limited set of datatypes. LiteralE [130] is one example, which encodes numeric and date information into its embedding space, while MKBE [131] encodes images and unstructured texts. Therefore, in this work, we propose a novel technique which directly encodes spatial footprints, namely point coordinates and bounding boxes, thereby making them available while learning knowledge graph embeddings.

Geographic information forms the basis for many KG downstream tasks such as geographic knowledge graph completion [73], geographic ontology alignment [132], geographic entity alignment [133], geographic question answering [47], and geographic knowledge graph summarization [69]. In the following, we will focus on geographic logic query answering as an example and more concretely on conjunctive graph queries (CGQ) or logic queries [83]. Due to the sparsity of information in knowledge graphs, many (geographic) queries are unanswerable *without spatial or non-spatial reasoning.* Knowledge graph embedding techniques have, therefore, been developed to handle unanswerable questions [83, 82, 47, 117] by inferring new triples in the KG embedding space based on existing ones. However, since most KG embedding models cannot handle *datatype properties* thus cannot encode geographic information into the KG embedding space, they perform spatial reasoning tasks poorly in the KG embedding space, which in turn leads to a poor performance of handling unanswerable geographic questions.

```
SELECT ?State WHERE {
 ?RiverMouth dbo:state ?State.                           (a)
 ?River dbo:mouthPosition ?RiverMouth.                   (b)
 ?River dbp:etymology dbr:Alexander_von_Humboldt.        (c)
}
```

Listing 4.1: Query q_A: An unanswerable SPARQL query over DBpedia which includes a partonomy relation

One example of unanswerable geographic questions that can be represented as a logic query is *which states contain the mouth of a river which is named after Alexander von Humboldt?* (Query q_A). The corresponding SPARQL query is shown in Listing 4.1. Running this query against the DBpedia SPARQL endpoint yields no results. In fact, two rivers are named after von Humboldt - `dbr:Humboldt_River` and `dbr:North_Fork_Humboldt_River` - and both have mouth positions as entities in DBpedia (`dbr:Humboldt_River__mouthPosition__1` and `dbr:North_Fork_Humboldt_River` __sourcePosition__1). However, the dbo:state (or dbo:isPartOf) relation between these river mouths and other geographic features such as states is missing. This makes Query q_A unanswerable (graph query pattern (a) in Listing 4.1). If we use the locations of the river mouths to perform a simple point-in-polygon test against the borders of all states in the US, we can deduce that dbr:Nevada contains both river mouths.

```
SELECT ?place WHERE {
 dbr:Yosemite_National_Park dbo:nearestCity ?place.
}
```

Listing 4.2: Query q_B: A SPARQL query over DBpedia which indicates a simple point-wise distance relation

Another example is the query in Listing 4.2, which asks for *the nearest city to Yosemite National Park* (Query q_B). If the triple dbr:Yosemite_National_Park dbo:nearestCity dbo:Mariposa,_California is missing from the current knowledge graph, Query q_B becomes unanswerable while it could simply be inferred by a distance-based query commonly used in GIS. Similar cases can include cardinal directions such as dbp:north. All these observations lead to the following research question: how could we enable spatial reasoning via *partonomic relations*, *point-wise metric relations*, and *directional relations* in the KG embedding-based systems?

One may argue that classical spatial reasoning can be used instead of direct location encoding to obtain answers to aforementioned questions. This is partially true for data and query endpoints that support GeoSPARQL and for datasets that are clean and complete. However, in some cases even GeoSPARQL-enabled query endpoints cannot accommodate spatial reasoning due to inherent challenges of representing spatial data in knowledge graphs. These challenges stem from principles of conceptual vagueness and uncertainty [39], and are further complicated by technical limitations. In this study we aim at enabling the model to perform *implicit spatial reasoning* in the hidden embedding space, which means instead of doing classical spatial reasoning by explicitly performing spatial operations during query time, the spatial information (points or bounding boxes) of geographic entities (e.g., $Indianapolis$) are directly encoded into the entity embeddings which are jointly optimized with relation embeddings (e.g, $isPartOf$). The trained embeddings of geographic entities encode their spatial information and the embeddings of spatial relations capture the logic of spatial reasoning. At query time, a normal link prediction process can be used to answer geographic questions and no explicit spatial reasoning is needed. Find more detail of this example in Section 4.7.

Existing approaches are only able to incorporate spatial information into the KG embedding space in a very limited fashion, e.g., through their training procedures. Furthermore, they estimate entity similarities based on some form of distance measures among entities, and ignore their absolute positions or relative directions. For example, Trisedya et al. [133] treated geographic coordinates as strings (a sequence of characters) and used a compositional function to encode these coordinate strings for geographic entities alignment. In order to incorporate distance relations between geographic entities, both Mai et al. [47] and Qiu et al. [73] borrowed the translation assumption from TransE [85]. For each geographic triple $s = (h, r, t)$ in the KG, where h and t are geographic entities, the geospatial distance between h and t determines the frequency of resampling this triple such that triples containing two closer geographic entities are sampled more frequently, and thus these two geographic entities are closer in the embedding space. Similarly, Yan et al. [69] used distance information to construct virtual spatial relations between geographic entities during the knowledge graph summarization process. This data conversion process (coordinates to pairwise distances) is unnecessarily expensive and causes information loss, e.g., absolute positions and relative directional information. In this work, we explore to directly encode entity locations into a high dimensional vector space, which preserves richer spatial information than distance measures. These location embeddings can be trained jointly with knowledge graph embedding.

Location encoders [64, 100, 65] refer to the neural network models which encode a pair of coordinates into a high dimensional embedding which can be used in downstream tasks such as geo-aware fine-grained image classification [64, 100, 65] and Point of Interest (POI) type classification [65]. Mai et al. [65] showed that multi-scale grid cell representation outperforms commonly used kernel based methods (e.g., RBF) as well as the single scale location encoding approaches. Given the success of location encoding in other machine learning tasks, the question is whether we can incorporate the location encoder architecture into a knowledge graph embedding model to make it spatially explicit [47]. One initial idea is directly using a location encoder as the entity encoder which encodes the spatial footprint (e.g., coordinates) of a geographic entity into a high dimensional vector. Such entity embeddings can be used in different decoder architectures for different tasks. However, several challenges remain to be solved for this initial approach.

First, point location encoding can handle point-wise metric relations such as distance (e.g., `dbo:nearestCity`) as well as directional relations (e.g., `dbp:north`, `dbp:south`) in knowledge graphs, but it is not easy to encode regions which are critical for relations such as containment (e.g., `dbo:isPartOf`, `dbo:location`, `dbo:city`, `dbo:state`, and `dbo:country`). For example, in Query q_A, the location encoder can encode `dbr:Yosemite_National_Park` and `dbo:Mariposa,_California` as two high dimensional embeddings based on which distance relations can be computed since the location embeddings preserve the relative distance information between locations [65]. However, point locations and location embeddings are insufficient to capture more complex relations between geographic entities such as containment as these require more complex spatial footprints (e.g., polygons). This indicates that we need to find a way to represent geographic entities as *regions* instead of points in the embedding space based on location encoders, especially for large scale geographic entities such as `dbr:California`, which is represented as a single pair of coordinates (a point) in many widely used KGs. We call this *scale effect* to emphasize the necessity of encoding the spatial extents of geographic entities instead of points, especially for large scale geographic entities.

The second challenge is how to seamlessly handle geographic and non-geographic entities together in the same entity encoder framework. Since location encoder is an essential *component* of the entity encoder, how should we deal with non-geographic entities that do not have spatial footprints? This is a non-trivial problem. For example, in order to weight triples using distance during KG embedding training, Qiu et al. [73] constructed a geographic knowledge graph which only contains geographic entities. Mai et al. [47] partially solved the problem by using a lower bound l as the lowest triple weight to handle non-geographic triples. However, this mechanism cannot distinguish triples involving both geographic and non-geographic entities from triples that only contain non-geographic entities.

The third challenge is how to capture the spatial and other semantic aspects at the same time when designing spatially explicit KG embedding model based on location encoders. The embedding of a geographic entity is expected to capture both its spatial (e.g., spatial extent) and other semantic information (e.g., type information) since both of them are necessary to answer geographic questions. Take Query q_A in Listing 4.1 as an example. We need both spatial and type information encoded in the entity embeddings to answer this question. The spatial information is necessary to perform partonomical reasoning in the embedding space to select geographic entities which *contain* a given river mouth, while type information is required to filter the answers and get entities with type *state*. The traditional KG embedding models fail to capture the spatial information which leads to a lower performance in geographic question answering.

Finally, thanks to the inductive learning nature of the location encoder, another interesting question is how to design a spatially-explicit KG embedding model so that it can be used to infer new relations between entities in a KG and any arbitrary location in the study area. We call this task **spatial semantic lifting** as an analogy to traditional *semantic lifting* which refers to the process of associating unstructured content to semantic knowledge resources [134]. For example, given any location $\vec{x}_i$, we may want to ask *which radio station broadcasts at* $\vec{x}_i$ i.e., to infer `dbo:broadcastArea`. None of the existing KG embedding models can solve this task.

In this work, we develop a spatially-explicit knowledge graph embedding model, *SE-KGE*, which directly solves those challenges. **The contributions of our work are as follow**:

1. We develop a spatially-explicit knowledge graph embedding model (*SE-KGE*), which applies a location encoder to incorporate spatial information (coordinates and spatial extents) of geographic entities. To the best of our knowledge, this is the first KG embedding model that can incorporate spatial information, especially spatial extents, of geographic entities into the model architecture.

2. *SE-KGE* is extended to an end-to-end geographic logic query answering model which predicts the most probable answers to unanswerable geographic logic queries over KG.

3. We apply *SE-KGE* on a novel task called *spatially semantic lifting*. Evaluations show that our model can substantially outperform the baseline by 9.86% on AUC and 9.59% on APR for the *DBGeo* dataset. Furthermore, our analysis shows that this model can achieve implicit spatial reasoning for different types of spatial relations.

The rest of this paper is structured as follow. We briefly summarize related work in Section 4.2. Then basic concepts are discussed in Section 4.3. In Section 4.4, we formalize the query answering and spatial semantic lifting task. Then, in Section 4.5, we give an overview of the logic query answering task before introducing our method. Section 4.6 describes the *SE-KGE* architecture. Experiments and evaluations are summarized in Section 4.7. Finally, we conclude our work in Section 4.8.

4.2 Related Work

In this section, we briefly review related work on knowledge graph embeddings, query answering, and location encoding.

4.2.1 Knowledge Graph Embedding

Learning knowledge graph embeddings (KGE) is an emerging topic in both the Semantic Web and machine learning fields. The idea is to represent entities and relations as vectors or matrices within an embedding spaces such that these distributed representations can be easily used in downstream tasks such as KG completion and question answering. Many KG embedding models have been proposed such as RESCAL [125], TransE [85], and TransH [86]. Most of these approaches cannot handle triples with data type properties nor triples involving spatial footprints.

The only KG embedding methods considering distance decay between geographic entities are Qiu et al. [73] and Mai et al. [47]. Mai et al. [47] computed the weight of each geographic triple $s = (h, r, t)$ as $\max(\ln \frac{D}{dis(h,t)+\varepsilon}, l)$ where h and t are geographic entities, and D is the longest (simplified) earth surface distance. ε is a hyperparameter to avoid zero denominator and l is the lowest edge weight we allow for each triple. As for non-geographic triples, l is used as the triple weight. Then this knowledge graph is treated as an undirected, unlabeled, edge-weighted multigraph. An edge-weighted PageRank is applied on this multigraph. The PageRank score for each node/entity captures the the structure information of the original KG as well as the distance decay effect among geographic entities. These scores are used in turn as weights to sample the entity context from the 1-degree neighborhood of each entity which is used in the KG embedding training process. As for Qiu et al. [73], the distance decay effect was deployed in a triple negative sampling process. Given a triple $s = (h, r, t)$ in the KG, each negative triple $s' = (h', r, t')$ of it was assigned a weight based on $w_{geo} = \dfrac{1}{1 + \left|log_{10}\dfrac{dis(h,t)+\theta}{dis(h',t')+\theta}\right|}$ where θ is a hyperparameter to avoid a zero denominator. w_{geo} is used in the max-margin loss function for the embedding model training. Note that non-geographic triples are not considered in Qiu et al. [73]. We can see that, instead of directly encoding an entity's location, they rely on some form of distance measures as weights for triple resampling. This process is computationally expensive and does not preserve other spatial properties such as direction. In contrast, our work introduces a direct encoding approach to handle spatial information.

4.2.2 Query Answering

Compared to link prediction [85], query answering [82, 83, 117] focuses on a more complex problem since answering a query requires a system to consider multiple triple patterns together. Wang et al. [82] designed an algorithm to answer a subset of SPARQL queries based on a pretrained KG embedding model. However, this is not an end-to-end model since the KG embedding training and query answering process are separated. Hamilton et al. [83] proposed an end-to-end logic query answering model, GQE, which can answer conjunctive graph queries. CGA [117] further improved GQE by using a self-attention based intersection operator. In our work, we will utilize GQE and CGA [117] as the underlying logic query answering baseline. We provide an overview about logic query answering in Section 4.5.

4.2.3 Location Encoding

Generating representations of points/locations that can benefit representation learning is a longstanding problem in machine learning. There are many well-established methods such as the *Kernel trick* [135] widely used in SVM classification and regression. However, these location representation methods use the positions of training examples as the centers of Gaussian kernels and thus need to memorize the training examples.

Kejriwal et al. [72] proposed a graph embedding approach to representing GeoNames locations as high dimensional embeddings. They converted the locations in GeoNames into a weighted graph where locations are nodes and the weight of each edge is computed based on the distance between two locations. Then a Glove [92] word embedding model is applied on this generated graph to obtain the embedding for each location. Despite its novelty, this model is a transductive learning based model which means if new locations are added, the weighted graph has to be regenerated and the whole model needs to be retrained. In other words, this embedding approach can not be easily generalized to unseen locations. This calls for inductive learning [136] based models.

Recently, location encoding technique [64, 100, 65] has been proposed to directly encode a location (a pair of coordinates) $\vec{x}$ as a high-dimension vector which can be incorporated into multiple downstream tasks. As shown by Mai et al. [65], the advantages of location encoding is that **1)** it can preserve absolute position information as well as relative distance and direction information between locations; **2)** it does not need to memorize the positions of training examples as all kernel based methods do [137]; **3)** In contrast to many transductive learning models, it is an inductive learning model [138] which can encode any location/point no matter it appears in the training dataset or not.

In theory, we can adopt any location encoder [100, 64, 65] to capture the spatial information of each geographic entity e_i in a knowledge graph $\mathcal{G}$. In this work, we utilize the $Space2Vec$ [65] location encoder, which is inspired by Nobel Prize-winning neuroscience research about grid cells [102] as well as the position encoding module of the Transformer model [104]. $Space2Vec$ first encodes a location $\vec{x}$ as a multi-scale periodic representation $PE(\vec{x})$ by using sinusoidal functions with different frequencies and then feeds the resulting embedding into a N layer feed forward neural network $\mathbf{NN}()$.

$$LocEnc^{(x)}(\vec{x}) = \mathbf{NN}(PE(\vec{x})) \tag{4.1}$$

The advantages of such location encoder compared to previous work [100, 64] are that **1)** it can be shown that location embeddings from $Space2Vec$ are able to preserve global position information as well as relative distance and direction, and that **2)** multi-scale representation learning approach outperforms traditional kernel-based methods (e.g., RBF) as well as single-scale location encoding approaches [100, 64] for several machine learning tasks. In the following, we will use $LocEnc^{(x)}()$ to denote the $Space2Vec$ model.

4.3 Basic Concepts

Definition 5 (Geographic Knowledge Graph) *A geographic knowledge graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ *is a directed edge and node labeled multigraph where* $\mathcal{V}$ *is a set of entities/nodes and* $\mathcal{E}$ *is the set of directed edges. Any directed and labeled edge will be called a triple* $s = (h, r, t)$ *where the nodes become heads* $h \in \mathcal{V}$ *and tails* $t \in \mathcal{V}$*, and the role label* $r \in \mathcal{R}$ *will be called the relationship between them. The set of triples/statements contained by* $\mathcal{G}$ *is denoted as* $\mathcal{T}$ *and* $\mathcal{R}$ *denotes as the set of relations (predicates, edge labels) in* $\mathcal{G}$*. Each triple can also be represented as* $r(h, t)$*, or* $r^{-1}(t, h)$ *where* r^{-1} *indicates the inverse relation of* r*.* $Domain(r)$ *and* $Range(r)$ *indicate the domain and range of relation* r*.*

$\Gamma() : \mathcal{V} \to \mathcal{C}$ *is a function which maps an entity* $e \in \mathcal{V}$ *to a unique type* $c \in \mathcal{C}$*, where* $\mathcal{C}$ *is the set of all entity types in* $\mathcal{G}$*.*[1]

[1] Note that, in many knowledge graphs (e.g., DBpedia, Wikidata), an entity can belong to multiple

The geographic entity set $\mathcal{V}_{pt}$ is a subset of $\mathcal{V}$ ($\mathcal{V}_{pt} \subseteq \mathcal{V}$). $\mathcal{PT}(\cdot)$ is a mapping function that maps any geographic entity $e \in \mathcal{V}_{pt}$ to its geographic location (coordinates) $\mathcal{PT}(e) = \vec{x}$ where $\vec{x} \in \mathcal{A} \subseteq \mathbb{R}^2$. Here $\mathcal{A}$ denotes the bounding box containing all geographic entities in the studied knowledge graph $\mathcal{G}$. We call it study area.

$\mathcal{V}_{pn}$ is a subset of $\mathcal{V}_{pt}$ ($\mathcal{V}_{pn} \subseteq \mathcal{V}_{pt}$) which represents the set of large-scale geographic entities whose spatial extent cannot be ignored. In this work, we use a bounding box to represent a geographic entity's spatial footprint. $\mathcal{PN}(\cdot)$ is a mapping function defined on $\mathcal{V}_{pn}$ that maps a geographic entity $e \in \mathcal{V}_{pn}$ to its spatial extent $\mathcal{PN}(e)$ and $\mathcal{PN}(e) = [\vec{x}^{min}; \vec{x}^{max}] \in \mathbb{R}^4$. In the vector concatenation above, $\vec{x}^{min}$, $\vec{x}^{max} \in \mathcal{A} \subseteq \mathbb{R}^2$ indicate the southwest and northeast point of the entity's bounding box.

Note that in many existing knowledge graphs, a triple can include a datatype property (e.g., `dbo:abstract`) implying that the tail is a literal. In line with related work [85, 126, 121, 89], we do not consider this kind of triples here in general. However, we do consider datatype properties about the spatial footprints of geographic entities implicitly by using $\mathcal{PT}(\cdot)$ *or* $\mathcal{PN}(\cdot)$.[2] While we do not model them directly as triples, we use the spatial footprints of geographic entities as input features for the entity encoder.

Definition 6 (Conjunctive Graph Query (CGQ)) *A query $q \in Q(\mathcal{G})$ that can be written as follows:*

$$\begin{aligned} & q = V_?.\exists V_1, V_2, .., V_m : b_1 \wedge b_2 \wedge ... \wedge b_n \\ \textit{where} \quad & b_i = r_i(e_k, V_l), V_l \in \{V_?, V_1, V_2, .., V_m\}, e_k \in \mathcal{V}, r \in \mathcal{R} \\ \textit{or} \quad & b_i = r_i(V_k, V_l), V_k, V_l \in \{V_?, V_1, V_2, .., V_m\}, k \neq l, r \in \mathcal{R} \end{aligned}$$

types. We use this definition to be in line with many existing work [83, 117] so that we can compare our results. It is easy to relax this requirement which we will discuss in Section 4.6.1.

[2] It is worth mentioning that most KG to date merely store point geometries even for features such as the United States.

Conjunctive graph queries are also called logic queries. Here $Q(\mathcal{G})$ is a set of all conjunctive graph queries that can be asked over $\mathcal{G}$. $V_?$ denotes the target variable of query q (*target node*) which will be replaced with the answer entity a, while $V_1, V_2, .., V_m$ are existentially quantified bound variables (*bound nodes*). $\{e_k | e_k \ in \ q\}$ is a set of anchor nodes and b_i is a basic graph pattern in this CGQ. We define the dependency graph of q as the graph with basic graph pattern $\{b_1, ..., b_n\}$ formed between the anchor nodes $\{e_k | e_k \ in \ q\}$ and the variable nodes $V_?, V_1, V_2, .., V_m$ (Figure 4.1). Each conjunctive graph query can be written as a SPARQL query.[3]

Note that the dependency graph of q represents computations on the KG and is commonly assumed to be a *directed acyclic graph (DAG)* [83] where the entities (*anchor nodes*) e_k in q are the source nodes and the target variable $V_?$ is the unique sink node. This restriction makes the logic query answering task in line with the usual question answering set up (e.g., semantic parsing [13, 12]).

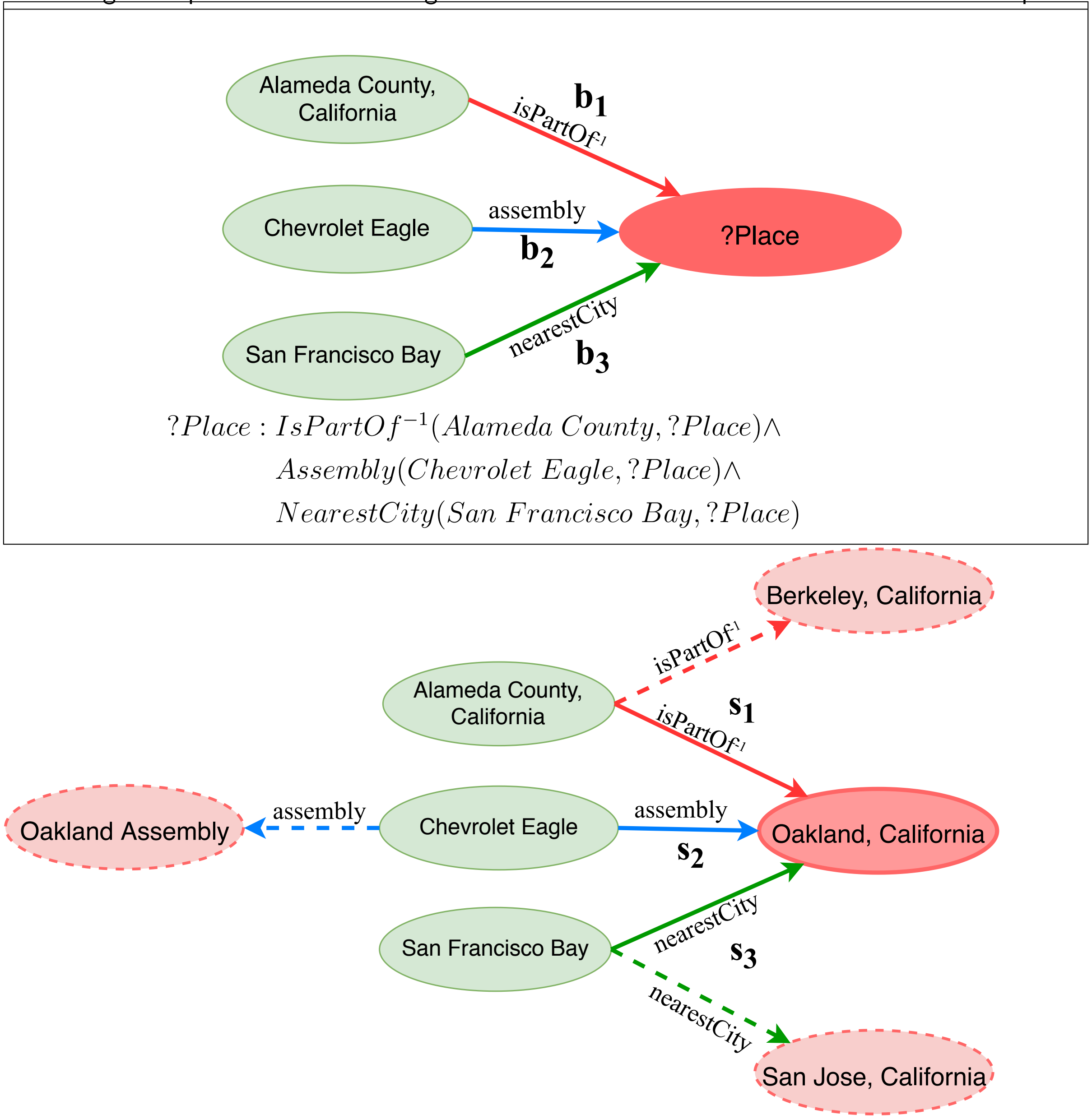
Definition 7 (Geographic Conjunctive Graph Query (GCGQ)) *A conjunctive graph query $q \in Q(\mathcal{G})$ is said to be a geographic conjunctive graph query if the answer entity a corresponding to the target variable $V_?$ is a geographic entity, i.e., $a = \varphi(\mathcal{G}, q) \wedge a \in \mathcal{V}_{pt}$ where $\varphi(\mathcal{G}, q)$ indicates the answer when executing query q on $\mathcal{G}$. We denote all possible geographic CGQ on $\mathcal{G}$ as $Q_{geo}(\mathcal{G}) \subseteq Q(\mathcal{G})$.*

An example geographic conjunctive query q_C is shown in Figure 4.1 whose corresponding SPARQL query is shown in Listing 4.3. The corresponding natural language question is *[which city in Alameda County, California is the assembly place of Chevrolet Eagle and the nearest city to San Francisco Bay]*. This query is especially interesting since it includes a non-spatial relation (`dbo:assembly`), a point-wise metric spatial relation (`dbo:nearestCity`) and a partonomy relation (`dbo:isPartOf`). Note that

[3] For the detail comparison between CQG and SPARQL 1.1 query, please refer to Section 2.1 of Mai et al. [117].

executing each basic graph pattern in Query q_C over DBpedia will yield multiple answers. For example, b_1 will return all subdivisions of Alameda County, California. b_2 matches multiple assembly places of Chevrolet Eagle such as `dbr:Oakland,_California`, `dbr:Oakland_Assembly`, and `dbr:Flint,_Michigan`. Interestingly, `dbr:Oakland_Assembly` should be located in `dbr:Oakland,_California` while there are no relationship between them in DBpedia except for their spatial footprints which can be inferred that they are closed to each other. b_3 will return three entities[4] - `dbr:San_Francisco`, `dbr:San_Jose,_California` and `dbr:Oakland,_California`. Combining these three basic graph patterns will yield one answer `dbr:Oakland,_California`. In our knowledge graph, both triple s_1 (See Figure 4.1) and triple s_2 are missing which makes Query q_C an unanswerable geographic query.

[4] `dbo:nearestCity` triples in DBpedia are triplified from the "Nearest major city" row of the info box in each entity's corresponding Wikipedia page which may contain several cities. See `http://dbpedia.org/resource/San_Francisco_Bay`.



$$
\begin{aligned}
?Place : IsPartOf^{-1}(Alameda\ County, ?Place) \wedge \\
Assembly(Chevrolet\ Eagle, ?Place) \wedge \\
NearestCity(San\ Francisco\ Bay, ?Place)
\end{aligned}
$$

Figure 4.1: **Query q_C: Top box:** Conjunctive Graph Query and Directed Acyclic Graph of the query structure corresponding to the SPARQL query in Listing 4.3. b_1, b_2, and b_3 indicates three basic graph patterns in query q_C. `?Place` is the target variable indicated as the red node while three green nodes are anchor nodes. There is no bound variable in this query. **Below:** The matched underlining KG patterns represented by solid arrows. s_1, s_2, and s_3 indicates the matched triples for b_1, b_2, and b_3 respectively for query q_C.

```
SELECT ?place  WHERE{
 ?place dbo:isPartOf dbr:Alameda_County,_California.     (1)
 dbr:Chevrolet_Eagle dbo:assembly ?place.                (2)
 dbr:San_Francisco_Bay dbo:nearestCity ?place.           (3)
}
```

Listing 4.3: Query q_C: A geographic conjunctive query which is rewritten as a SPARQL query over DBpedia including both non-spatial relations and different types of spatial relation.

4.4 Problem Statement

In this work, we focus on two geospatial tasks - *geographic logic query answering* and *spatial semantic lifting*.

Task 1 (Logic Query Answering) *Given a geographic knowledge graph $\mathcal{G}$ and an unanswerable conjunctive graph query $q \in Q(\mathcal{G})$ (i.e., $\varphi(\mathcal{G}, q) = \emptyset$), a query embedding function $\mathbf{\Phi}_{\mathcal{G},\theta}(q) : Q(\mathcal{G}) \rightarrow \mathbb{R}^d$, which is parameterized by θ, is defined to map q to a vector representation of d dimension. The most probable answer a' to q is the entity nearest to $\mathbf{q} = \mathbf{\Phi}_{\mathcal{G},\theta}(q)$ in the embedding space:*

$$a' = \arg\max_{e_i \in \mathcal{V}} \ \mathbf{\Omega}(\mathbf{\Phi}_{\mathcal{G},\theta}(q), Enc(e_i)) = \arg\max_{e_i \in \mathcal{V}} \ \mathbf{\Omega}(\mathbf{q}, \mathbf{e}_i) \tag{4.2}$$

Here $\mathbf{e}_i = Enc(e_i) \in \mathbb{R}^d$ is the entity embedding of e_i produced by an embedding encoder $Enc()$. $\mathbf{\Omega}(\cdot)$ denotes the cosine similarity function:

$$\mathbf{\Omega}(\mathbf{q}, \mathbf{e}_i) = \frac{\mathbf{q} \cdot \mathbf{e}_i}{\parallel \mathbf{q} \parallel \parallel \mathbf{e}_i \parallel} \tag{4.3}$$

Note that q can be a geographic query or non-geographic query, i.e., $q \in (Q(\mathcal{G}) \setminus Q_{geo}(\mathcal{G})) \vee Q_{geo}(\mathcal{G})$. *Geographic logic query answering* indicates a logic query answering process over $Q_{geo}(\mathcal{G})$. The query embedding function $\mathbf{\Phi}_{\mathcal{G},\theta}(q)$ is constructed based on all three components of *SE-KGE* without any extra parameters: $Enc()$, $\mathcal{P}()$, and $\mathcal{I}()$, i.e., $\theta = \{\theta_{Enc}, \theta_{\mathcal{P}}, \theta_{\mathcal{I}}\}$.

Task 2 (Spatial Semantic Lifting) *Given a geographic knowledge graph $\mathcal{G}$ and an arbitrary location $\vec{x} \in \mathcal{A} \subseteq \mathbb{R}^2$ from the current study area $\mathcal{A}$, and a relation $r \in \mathcal{R}$ such that $Domain(r) \subseteq \mathcal{V}_{pt}$, we define a spatial semantic lifting function $\mathbf{\Psi}_{\mathcal{G},\theta_{ssl}}(\vec{x}, r) : \mathcal{A} \times \mathcal{R} \to \mathbb{R}^d$, which is parameterized by θ_{ssl}, to map $\vec{x}$ and r to a vector representation of d dimension, i.e., $\mathbf{s} = \mathbf{\Psi}_{\mathcal{G},\theta_{ssl}}(\vec{x}, r) \in \mathbb{R}^d$. A nearest neighbor search is utilized to search for the most probable entity $e' \in \mathcal{V}_{pt}$ so that a virtual triple can be constructed between location $\vec{x}$ and e', i.e., $r(\vec{x}, e')$, where*

$$e' = \underset{e_i \in \mathcal{V}}{\arg\max}\ \mathbf{\Omega}(\mathbf{\Psi}_{\mathcal{G},\theta_{ssl}}(\vec{x}, r), Enc(e_i)) = \underset{e_i \in \mathcal{V}}{\arg\max}\ \mathbf{\Omega}(\mathbf{s}, \mathbf{e}_i) \tag{4.4}$$

The spatial semantic lifting function $\mathbf{\Psi}_{\mathcal{G},\theta_{ssl}}(\vec{x}, r)$ consists of two components of *SE-KGE* without any extra parameter: $Enc()$ and $\mathcal{P}()$, i.e., $\theta_{ssl} = \{\theta_{Enc}, \theta_{\mathcal{P}}\}$. This spatial semantic lifting task is related to the link prediction task [124] which is commonly used in the knowledge graph embedding literature [85, 126, 128]. The main difference is that instead of predicting links between entities in the original knowledge graph $\mathcal{G}$ as link prediction does, spatial semantic lifting links an arbitrary location $\vec{x}$ to $\mathcal{G}$. *Since none of the existing KG embedding models can directly encode locations, they cannot be used for spatial semantic lifting.*

4.5 Logic Query Answering Backgrounds

Before introducing our *SE-KGE* model, we will first give an overview of how previous work [83, 117] tackled the logic query answering task with KG embedding models. Generally speaking, a logic query answering model is composed of three major components: entity encoder $Enc()$, projection operator $\mathcal{P}()$, and intersection operator $\mathcal{I}()$.

1. **Entity encoder** $Enc()$: represents each entity as a high dimension vector (embedding);

2. **Projection operator** $\mathcal{P}()$: given a basic graph pattern $b = r(e_i, V_j)$ (or $b = r(V_i, V_j)$) in a CGQ q while the subject embedding $\mathbf{e}_i$ (or $\mathbf{v}_i$) of entity e_i (or Variable V_i) is known beforehand, $\mathcal{P}()$ *projects* the subject embedding through a relation specific matrix to predict the embedding of V_j.

3. **Intersection operator** $\mathcal{I}()$: integrates different predicted embeddings of the same Variable (e.g., V_j) from different basic graph patterns into one single embedding to represent this variable.

Given these three neural network modules, any CGQ q can be encoded according by following their DAG query structures such that the embedding of the unique target variable $V_?$ for each query can be obtained - $\mathbf{v}_?$. We call it *query embedding* $\mathbf{q} = \mathbf{\Phi}_{\mathcal{G},\theta}(q) = \mathbf{v}_?$ for CGQ q. Then the most probable answer is obtained by a nearest neighbor search for $\mathbf{q}$ in the entity embedding space (See Equation 4.2). Our work will follow the same model component setup and query embedding computing process. However, neither Hamilton et al. [83] nor Mai et al. [117] has considered encoding spatial information of geographic entities into the entity embedding space which is the core contribution of our work. Moreover, we extend the current model architecture such that it can also be applied to the spatial semantic lifting task, a new task we proposed which is impossible for previous works [83, 117]. In the following, we will use $\cdot^{(GQE)}$ and $\cdot^{(CQA)}$ to indicate that these are model components used by Hamilton et al. [83] nor Mai et al. [117]:

4.5.1 Entity Encoder

In general, an entity encoder aims at representing any entity in a KG as a high dimension embedding so that it can be fed into following neural network modules. The normal practice shared by most KG embedding models [85, 126, 86, 127, 87, 47, 128, 73] is to initialize an embedding matrix randomly where each column indicates an embedding for a specific entity. The entity encoding becomes an *embedding lookup* process and these embeddings will be updated during the neural network backpropagation in the training time.

Previous works have demonstrated that most of the information captured by entity embeddings is type information [139, 83]. So Hamilton et al. [83] and Mai et al. [117] took a step further and use a type-specific embedding lookup approach. We call the resulting module entity feature encoder $Enc^{(c)}()$.

Definition 8 (Entity Feature Encoder: $Enc^{(c)}()$**)** *Given any entity* $e_i \in \mathcal{V}$ *with type* $c_i = \Gamma(e_i) \in \mathcal{C}$ *from* $\mathcal{G}$*, entity feature encoder* $Enc^{(c)}()$ *computes the feature embedding* $\mathbf{e}_i^{(c)} \in \mathbb{R}^{d^{(c)}}$ *which captures the type information of entity* e_i *by using an embedding lookup approach:*

$$\mathbf{e}_i^{(c)} = Enc^{(c)}(e_i) = \frac{\mathbf{Z}_{c_i}\mathbf{h}_i^{(c)}}{\| \mathbf{Z}_{c_i}\mathbf{h}_i^{(c)} \|_{L2}} \tag{4.5}$$

Here $\mathbf{Z}_{c_i} \in \mathbb{R}^{d^{(c)} \times |\mathcal{C}|}$ is the type-specific embedding matrix for all entities with type $c_i = \Gamma(e_i) \in \mathcal{C}$. $\mathbf{h}_i^{(c)}$ is a one-hot vector such that $\mathbf{Z}_{c_i}\mathbf{h}_i^{(c)}$ will perform an embedding lookup operation which selects an entity feature embedding from the corresponding column. $\| \cdot \|_{L2}$ indicates the $L2$-norm.

Both Hamilton et al. [83] and Mai et al. [117] use $Enc^{(c)}()$ as their entity encoder (See Equation 4.6). Figure 4.2 is an illustration of their approach. Note that this encoder does not consider the spatial information (e.g., coordinates and spatial extents) of geographic entities which causes a lower performance for answering geographic logic queries. As for our *SE-KGE* model, we add an additional entity space encoder $Enc^{(x)}()$ to handle this (See Definition 11).

$$Enc^{(GQE)}(e_i) = Enc^{(CGA)}(e_i) = Enc^{(c)}(e_i) \tag{4.6}$$

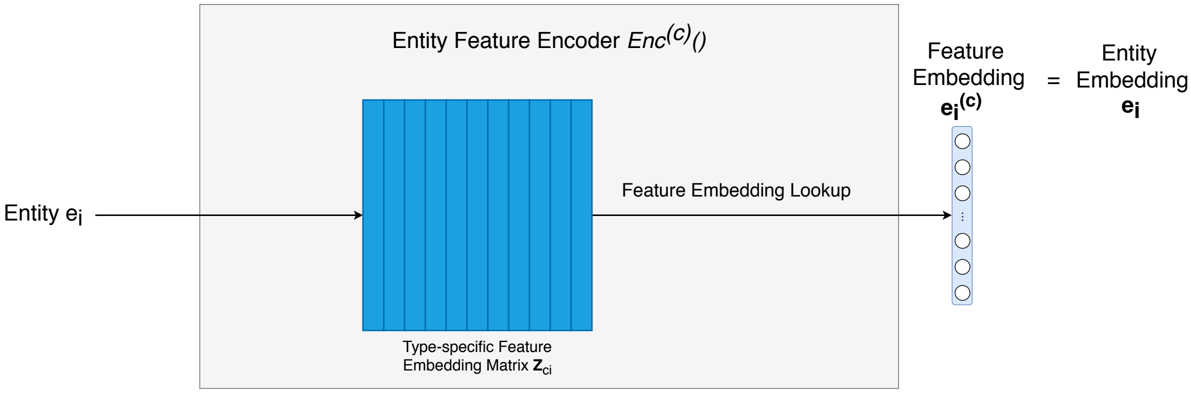

Entity Feature Encoder Enc^(c)()
Entity e_i
Feature Embedding Lookup
Type-specific Feature Embedding Matrix Z_ci
Feature Embedding e_i^(c)
=
Entity Embedding e_i

Figure 4.2: The entity encoder used by Hamilton et al. [83] and Mai et al. [117].

4.5.2 Projection Operator

The projection operator is utilized to do link prediction: given a basic graph pattern $b = r(h_i, V_j)$ in a conjunctive graph query q with relation r in which h_i is either an entity e_i (an anchor node in q) or an existentially quantified bound variable V_i, the projection operator $\mathcal{P}()$ predicts the embedding $\mathbf{e}'_i \in \mathbb{R}^{d^{(c)}}$ for Variable V_j. Here, the embedding of h_i can be either the entity embedding $\mathbf{e}_i = Enc^{(c)}(e_i)$ or the computed embedding $\mathbf{v}_i$ for V_i which is known beforehand. Both Hamilton et al. [83] and Mai et al. [117] share the same projection operator $\mathcal{P}^{(GQE)} = \mathcal{P}^{(CGA)}$ (See Equation 4.7) by using a bilinear matrix $\mathbf{R}_r \in \mathbb{R}^{d^{(c)} \times d^{(c)}}$. $\mathbf{R}_r$ can also be a bilinear diagonal matrix as DisMult [140] whose corresponding projection operator is indicated as $\mathcal{P}^{(GQE_{diag})}$.

$$\mathbf{e}_i' = \begin{cases} \mathcal{P}^{(GQE)}(e_i, r) = \mathcal{P}^{(CGA)}(e_i, r) = \mathbf{R}_r Enc^{(c)}(e_i) = \mathbf{R}_r \mathbf{e}_i & if\ input\ = (e_i, r) \\ \mathcal{P}^{(GQE)}(V_i, r) = \mathcal{P}^{(CGA)}(V_i, r) = \mathbf{R}_r \mathbf{v}_i & if\ input\ = (V_i, r) \end{cases} \tag{4.7}$$

In *SE-KGE*, we extend projection operator $\mathcal{P}()$ so that it can be used in the spatial semantic lifting task (See Definition 12).

Figure 4.3 uses the basic graph pattern $b_2 = Assembly(Chevrolet\ Eagle, ?Place)$ in Figure 4.1 as an example to demonstrate how to do link prediction with $\mathcal{P}^{(GQE)}() = \mathcal{P}^{(CGA)}()$. The result embedding $\mathbf{e}_{?2}$ can be treated as the prediction of the embedding of Variable `?Place`. By following the same process, we can predict the embedding of the variable `?Place` from the other two basic graph patterns b_1 and b_3 - $\mathbf{e}_{?1}$ and $\mathbf{e}_{?3}$.

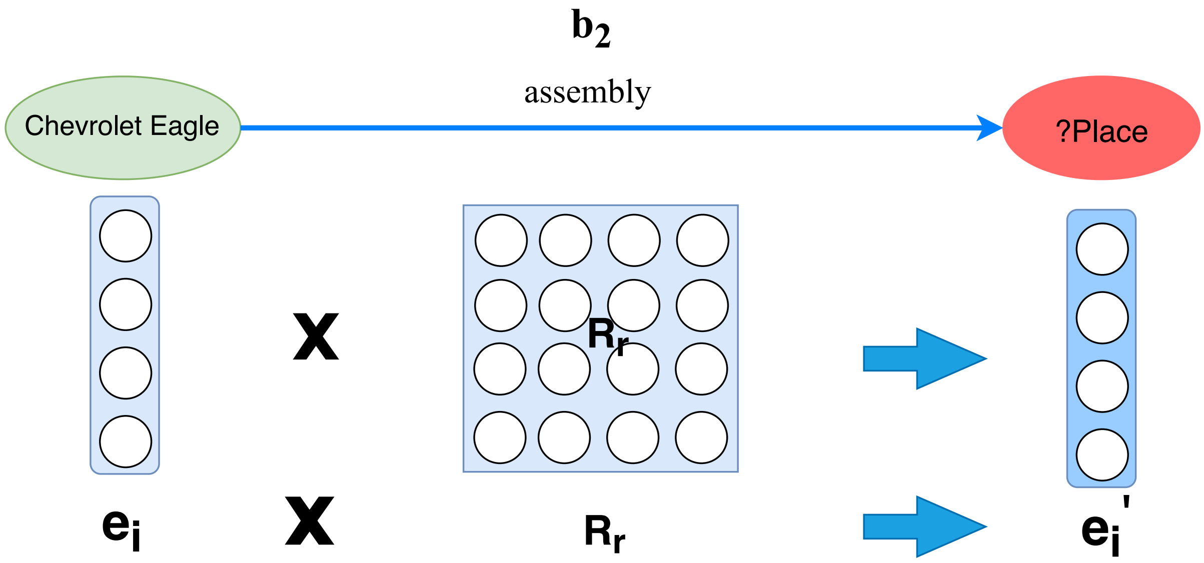


Figure 4.3: An illustration of projection operator $\mathcal{P}^{(GQE)}() = \mathcal{P}^{(CGA)}()$ used by Hamilton et al. [83] and Mai et al. [117].

4.5.3 Intersection Operator

The intersection operator $\mathcal{I}()$ is used to *integrate* multiple embeddings $\mathbf{e}_{1?}$, $\mathbf{e}_{2?}$, ..., $\mathbf{e}_{i?}$,..., $\mathbf{e}_{n?}$ which represent the same (bound or target) variable $V_?$ in a CGQ q to produce one single embedding $e_?$ to represent this variable. Figure 4.4 illustrates this idea by using CGQ q_C in Figure 4.1 as an example where $\mathbf{e}_{?1}$, $\mathbf{e}_{?2}$ and $\mathbf{e}_{?3}$ indicates the predicted embedding of `?Place` from three different basic graph pattern b_1, b_2, and b_3. The intersection operator *integrates* them into one single embedding $\mathbf{e}_?$ to represent `?Place`. Since `?Place` is the target variable of q, $\mathbf{e}_?$ is the final *query embedding* we use to do nearest neighbor search to obtain the most probable answer (See Task 1). More formally,

Definition 9 (Intersection Operator $\mathcal{I}()$) *Given a set of n different input embeddings $\mathbf{e}_{1?}$, $\mathbf{e}_{2?}$, ..., $\mathbf{e}_{j?}$,..., $\mathbf{e}_{n?}$, intersection operator $\mathcal{I}()$ produces one single embedding $\mathbf{e}_?$:*

$$\mathbf{e}_? = \mathcal{I}(\{\mathbf{e}_{1?}, \mathbf{e}_{2?}, ..., \mathbf{e}_{j?}, ..., \mathbf{e}_{n?}\}) \tag{4.8}$$

Intersection operator $\mathcal{I}()$ represents the logical conjunction in the embedding space. Any permutation invariant function can be used here as a conjunction such as elementwise mean, maximum, and minimum. We can also use any permutation invariant neural network architecture [115] such as Deep Sets [115]. *GQE* [83] used an elementwise minimum plus a feed forward network as the intersection operator which we indicate as $\mathcal{I}^{(GQE)}()$. Mai et al. [117] showed that their *CGA* model with a self-attention based intersection operator $\mathcal{I}^{(CGA)}()$ can outperform *GQE*. So in this work, we use $\mathcal{I}^{(CGA)}()$ as the intersection operator $\mathcal{I}()$. Readers that are interested in this technique are suggested to check Mai et al. [117] for more details.

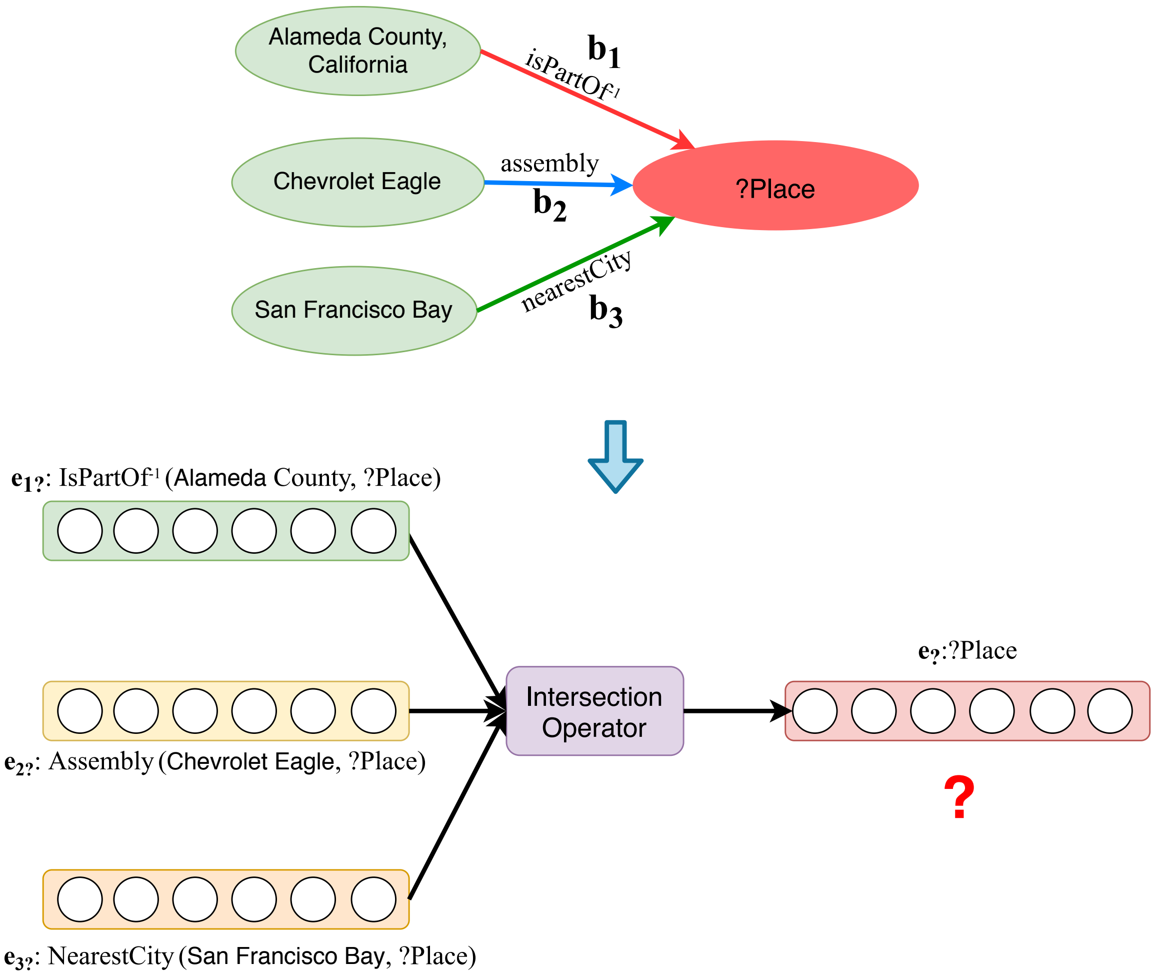

Alameda County, California
$\mathbf{b}_1$
isPartOf^{-1}
assembly
Chevrolet Eagle
$\mathbf{b}_2$
?Place
San Francisco Bay
nearestCity
$\mathbf{b}_3$
$\mathbf{e}_{1?}$: IsPartOf^{-1} (Alameda County, ?Place)
$\mathbf{e}_{2?}$: Assembly (Chevrolet Eagle, ?Place)
$\mathbf{e}_{3?}$: NearestCity (San Francisco Bay, ?Place)
Intersection Operator
$\mathbf{e}_{?}$:?Place
?

Figure 4.4: An illustration of intersection operator $\mathcal{I}()$.

4.5.4 Query Embedding Computing

Hamilton et al. [83] proposed a way to compute the query embedding of a CGQ q based on these three components. Given a CGQ q, we can encode all its anchor nodes (entities) into entity embedding space using $Enc()$. Then we recursively apply the projection operator $\mathcal{P}()$ and intersection operator $\mathcal{I}()$ by following the DAG of q until we get an embedding for the target node (variable $V_?$), i.e., $\mathbf{q} = \mathbf{\Phi}_{\mathcal{G},\theta}(q) = \mathbf{v}_?$. Then we use the nearest neighbor search in the entity embedding space to find the *closest* embedding, whose corresponding entity will be the predicted answer to Query q. For details of the query embedding algorithm, please refer to Hamilton et al. [83].

Figure 4.5 gives an illustration of the query embedding computation process in the embedding space by using Query q_C as an example. We first use $Enc()$ to get the embeddings of three anchor nodes (see the dash green box in Figure 4.5.). Then $\mathcal{P}()$ (three green arrows) is applied to each basic graph pattern to get three embeddings $\mathbf{e}_{1?}$, $\mathbf{e}_{2?}$, and $\mathbf{e}_{3?}$. $\mathcal{I}()$ (red arrows) is used later on to integrate them into one single embedding $\mathbf{e}_{?}$ or $\mathbf{q}$ for the target variable `?Place`.

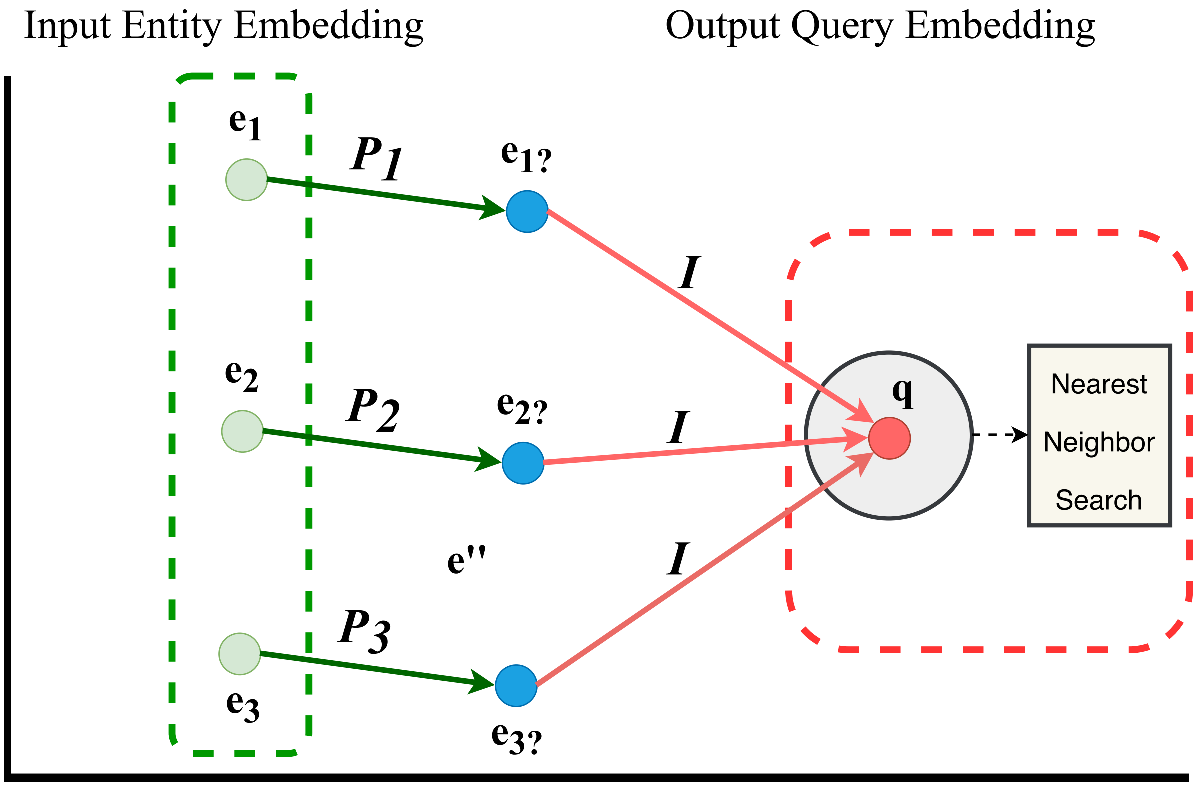


Figure 4.5: An illustration of (geographic) logic query answering in the embedding space

In this work, we follow the same query embedding computation process. Furthermore, we extent the current model architecture to do spatial semantic lifting.

4.6 *SE-KGE* Model

Since many geographic questions highly rely on spatial information (e.g., coordinates) and spatial reasoning, a spatially-explicit model is desired for the geographic logic query answering task. Moreover, the spatial semantic lifting task, Task 2, is only possible if we have an entity encoder which can encode the spatial information of geographic entities as well as a specially designed projection operator. To solve these problem, we propose a new entity encoder *Enc*() (See Section 4.6.1) and a new projection operator (See Section 4.6.2) for our *SE-KGE* model. Next, Task 1 and 2 require different training processes which will be discussed in Section 4.6.3 and 4.6.4. *SE-KGE* extends the general logic query answering framework of *GQE* [83] and *CGA* [117] with explicit spatial embedding representations.

4.6.1 Entity Encoder

Definition 10 (Entity Encoder: *Enc*()**)** *Given a geographic knowledge graph $\mathcal{G}$, entity encoder $Enc() : \mathcal{V} \rightarrow \mathbb{R}^d$ is defined as a function parameterized by θ_{Enc}, which maps any entity $e_i \in \mathcal{V}$ to a vector representation of d dimension, so called entity embedding $\mathbf{e}_i \in \mathbb{R}^d$. $Enc()$ consists of two parts – the entity feature encoder $Enc^{(c)}() : \mathcal{V} \rightarrow \mathbb{R}^{d^{(c)}}$ and the entity space encoder $Enc^{(x)}() : \mathcal{V} \rightarrow \mathbb{R}^{d^{(x)}}$. These two encoders map any entity $e_i \in \mathcal{V}$ to a feature embedding $\mathbf{e}_i^{(c)} \in \mathbb{R}^{d^{(c)}}$ and space embedding $\mathbf{e}_i^{(x)} \in \mathbb{R}^{d^{(x)}}$, respectively. The final entity embedding $\mathbf{e}_i$ is the concatenation of $\mathbf{e}_i^{(c)}$ and $\mathbf{e}_i^{(x)}$, i.e., :*

$$\mathbf{e}_i = Enc(e_i) = [Enc^{(c)}(e_i); Enc^{(x)}(e_i)] = [\mathbf{e}_i^{(c)}; \mathbf{e}_i^{(x)}] \tag{4.9}$$

Here $[;]$ denotes vector concatenation of two column vectors and $d = d^{(c)} + d^{(x)}$. $Enc^{(c)}()$ has been defined in Definition 8.

Entity Space Encoder

In our work, and instead of calling them *location encoder* and *location embedding* [64], we use the term *space encoder* to refer to the neural network model that encodes the spatial information of an entity and call the encoding results *space embeddings*. While location encoder focus on encoding one single point location, our space encoder $Enc^{(x)}()$ aims at handling spatial information of geographic entities at different scales:

1. For a small geographic entity $e_i \in \mathcal{V}_{pt} \setminus \mathcal{V}_{pn}$ such as radio stations or restaurants, we use its location $\vec{x}_i = \mathcal{PT}(e_i)$ as the input to $Enc^{(x)}()$.

2. For an geographic entity with a large extent $e_i \in \mathcal{V}_{pn}$ such as countries and states, at each encoding time, we randomly generate a point $\vec{x}_i^{(t)}$ as the input for $Enc^{(x)}()$ based on the 2D uniform distribution defined on its spatial extent (bounding box) $\mathcal{PN}(e_i) = [\vec{x}_i^{min}; \vec{x}_i^{max}]$, i.e., $\vec{x}_i^{(t)} \sim \mathcal{U}(\vec{x}_i^{min}, \vec{x}_i^{max})$. Since during training $Enc^{(x)}()$ will be called multiple times, it will at the end learn a uniform distribution over e_i's bounding box. In practice, one can sample using any process, such as stratified random sampling, or vary the sampling density by expected variation.

3. For non-geographic entity $e_i \in \mathcal{V} \setminus \mathcal{V}_{pt}$, we randomly initialize its space embedding. One benefit of this approach is that during the KG embedding training process, these embeddings will be updated based on back propagation in neural networks so that the spatial information of its connected entities in $\mathcal{G}$ will propagate to this embedding as its pseudo space footprint. For example, a person's spatial embedding will be close to the embedding of his/her birthplace or hometown.

The entity space encoder $Enc^{(x)}()$ is formally defined as follow:

Definition 11 (Entity Space Encoder: $Enc^{(x)}()$) *Given any entity $e_i \in \mathcal{V}$ from $\mathcal{G}$,*

$Enc^{(x)}()$ computes the space embedding $\mathbf{e}_i^{(x)} = Enc^{(x)}(e_i) \in \mathbb{R}^{d^{(x)}}$ by

$$\mathbf{e}_i^{(x)} = \begin{cases} LocEnc^{(x)}(\vec{x}_i)\ , where\ \vec{x}_i = \mathcal{PT}(e_i)\ , & if\ e_i \in \mathcal{V}_{pt} \setminus \mathcal{V}_{pn} \\ LocEnc^{(x)}(\vec{x}_i^{(t)})\ , where\ \vec{x}_i^{(t)} \sim \mathcal{U}(\vec{x}_i^{min}, \vec{x}_i^{max})\ ,\ \mathcal{PN}(e_i) = [\vec{x}_i^{min}; \vec{x}_i^{max}]\ , & if\ e_i \in \mathcal{V}_{pn} \\ \dfrac{\mathbf{Z}_x \mathbf{h}_i^{(x)}}{\parallel \mathbf{Z}_x \mathbf{h}_i^{(x)} \parallel_{L2}}, & if\ e_i \in \mathcal{V} \setminus \mathcal{V}_{pt} \end{cases} \tag{4.10}$$

Here $\mathbf{Z}_x$ and $\mathbf{h}_i^{(x)}$ are the embedding matrix and one-hot vector for non-geographic entities in entity space encoder $Enc^{(x)}()$ similar to Equation 4.5. $LocEnc^{(x)}()$ denotes a location encoder module (See Equation 4.1). Figure 4.6 illustrates the architecture of entity encoder $Enc()$. Compared with GQE's entity encoder $Enc^{(GQE)}()$ shown in Figure 4.2, the proposed entity encoder of *SE-KGE* adds the entity space encoder $Enc^{(x)}()$ which leverages a multi-scale grid cell representation to capture the spatial information of geographic entities.

As far as using a bounding box as approximation is concerned, one reason to use bounding boxes instead of the real geometries is that doing point-in-polygon operation in real time during ML model training is very expensive and not efficient. Many spatial databases use bounding boxes as approximations of the real geometries to avoid intensive computation. We adopt the same strategy here. Moreover, the detailed spatial footprint of e_i is expected to be captured through the training process of the entity embedding. For example, even if the model is only aware of the bounding box of California, by using the `dbo:isPartOf` relations between California and its subdivisions, the model will be informed of all the spatial extents of its subdivisions.

4.6.2 Projection Operator

Definition 12 (Projection Operator $\mathcal{P}()$) *Given a geographic knowledge graph $\mathcal{G}$, a projection operator $\mathcal{P}() : \mathcal{V} \cup \mathcal{A} \times \mathcal{R} \rightarrow \mathbb{R}^d$ maps a pair of (e_i, r), (V_i, r), or $(\vec{x}_i, r)$, to*

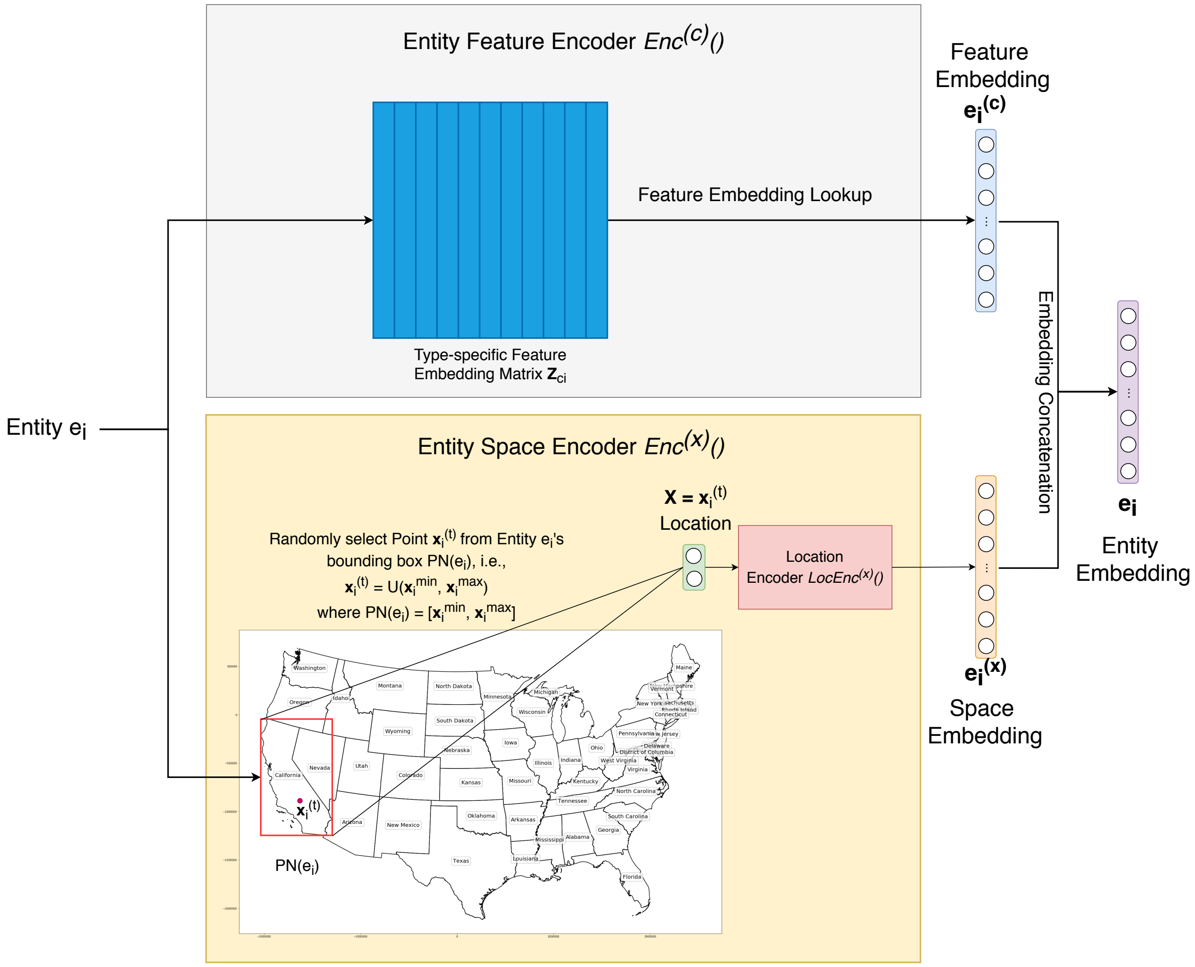

Entity Feature Encoder $Enc^{(c)}()$
Feature Embedding $\mathbf{e}_i^{(c)}$
Feature Embedding Lookup
Type-specific Feature Embedding Matrix $\mathbf{Z}_{c_i}$
Entity e_i
Embedding Concatenation
$\mathbf{e}_i$
Entity Embedding
Entity Space Encoder $Enc^{(x)}()$
$\mathbf{X} = \mathbf{x}_i^{(t)}$
Location
Randomly select Point $\mathbf{x}_i^{(t)}$ from Entity e_i's bounding box $PN(e_i)$, i.e.,
$\mathbf{x}_i^{(t)} = U(\mathbf{x}_i^{min}, \mathbf{x}_i^{max})$
where $PN(e_i) = [\mathbf{x}_i^{min}, \mathbf{x}_i^{max}]$
Location Encoder $LocEnc^{(x)}()$
$\mathbf{e}_i^{(x)}$
Space Embedding
$\mathbf{x}_i^{(t)}$
$PN(e_i)$

Figure 4.6: The entity encoder $Enc()$ of *SE-KGE*. Compared with previous work (Figure 4.2) an entity space encoder component $Enc^{(x)}()$ is added to capture the spatial information of geographic entities.

an embedding $\mathbf{e}'_i$. According to the input, $\mathcal{P}()$ can be treated as: ***(1) link prediction*** $\mathcal{P}^{(e)}(e_i, r)$***:*** *given a triple's head entity e_i and relation r, predicting the tail; (2)* ***link prediction*** $\mathcal{P}^{(e)}(V_i, r)$***:*** *given a basic graph pattern $b = r(V_i, V_j)$ and $\mathbf{v}_i$ which is the computed embedding for the existentially quantified bound variable V_i, predicting the embedding for Variable V_j;* ***(2) spatial semantic lifting*** $\mathcal{P}^{(x)}(\vec{x}_i, r)$***:*** *given an arbitrary location $\vec{x}_i$ and relation r, predicting the most probable linked entity. Formally, $\mathcal{P}()$ is* *defined as:*

$$\mathbf{e}'_i = \begin{cases} \mathcal{P}^{(e)}(e_i, r) = diag(\mathbf{R}_r^{(c)}, \mathbf{R}_r^{(x)})Enc(e_i) = diag(\mathbf{R}_r^{(c)}, \mathbf{R}_r^{(x)})\mathbf{e}_i & if\ input\ = (e_i, r) \\ \mathcal{P}^{(e)}(V_i, r) = diag(\mathbf{R}_r^{(c)}, \mathbf{R}_r^{(x)})\mathbf{v}_i & if\ input\ = (V_i, r) \\ \mathcal{P}^{(x)}(\vec{x}_i, r) = diag(\mathbf{R}_r^{(xc)}, \mathbf{R}_r^{(x)})[LocEnc^{(x)}(\vec{x}_i); LocEnc^{(x)}(\vec{x}_i)] & if\ input\ = (\vec{x}_i, r) \end{cases} \tag{4.11}$$

where $\mathbf{R}_r^{(c)} \in \mathbb{R}^{d^{(c)} \times d^{(c)}}$, $\mathbf{R}_r^{(x)} \in \mathbb{R}^{d^{(x)} \times d^{(x)}}$, and $\mathbf{R}_r^{(xc)} \in \mathbb{R}^{d^{(c)} \times d^{(x)}}$ are three trainable and relation-specific matrices. $\mathbf{R}_r^{(c)}$ and $\mathbf{R}_r^{(x)}$ focus on the feature embedding and space embedding. $\mathbf{R}_r^{(xc)}$ transforms the space embedding $\mathbf{e}_i^{(x)}$ to its correspondence in feature embedding space. $diag(\mathbf{R}_r^{(c)}, \mathbf{R}_r^{(x)}) \in \mathbb{R}^{d \times d}$ and $diag(\mathbf{R}_r^{(xc)}, \mathbf{R}_r^{(x)}) \in \mathbb{R}^{d \times 2d^{(x)}}$ indicate two block diagonal matrices based on $\mathbf{R}_r^{(c)}$, $\mathbf{R}_r^{(x)}$, and $\mathbf{R}_r^{(xc)}$. $[LocEnc^{(x)}(\vec{x}_i); LocEnc^{(x)}(\vec{x}_i)]$ indicates the concatenation of two identical space embedding $LocEnc^{(x)}(\vec{x}_i)$. Here, we use the same $\mathcal{P}^{(e)}()$ for the first two cases to indicate they share the same neural network architecture. This is because both of them are link prediction tasks with different inputs.

Link Prediction: Figure 4.7 illustrates the idea of projection operator $\mathcal{P}^{(e)}()$ by using the basic graph pattern b_2 in q_C (See Figure 4.1) as an example (the first case). Given the embedding of `dbr:Chevrolet_Eagle` and the relation-specific matrix $diag(\mathbf{R}_r^{(c)}, \mathbf{R}_r^{(x)})$ for relation `dbo:assembly`, we can predict the embedding of the variable `?Place` - $\mathbf{e}_{?2}$.

Spatial Semantic Lifting: Figure 4.8 shows how to use $\mathcal{P}^{(x)}()$ in the semantic lifting task. See Section 4.6.4 for detail description. Note that "×" in Figure 4.7 and 4.8 indicates $diag(\mathbf{R}_r^{(c)}, \mathbf{R}_r^{(x)})\mathbf{e}_i$ and $diag(\mathbf{R}_r^{(xc)}, \mathbf{R}_r^{(x)})[LocEnc^{(x)}(\vec{x}_i); LocEnc^{(x)}(\vec{x}_i)]$.

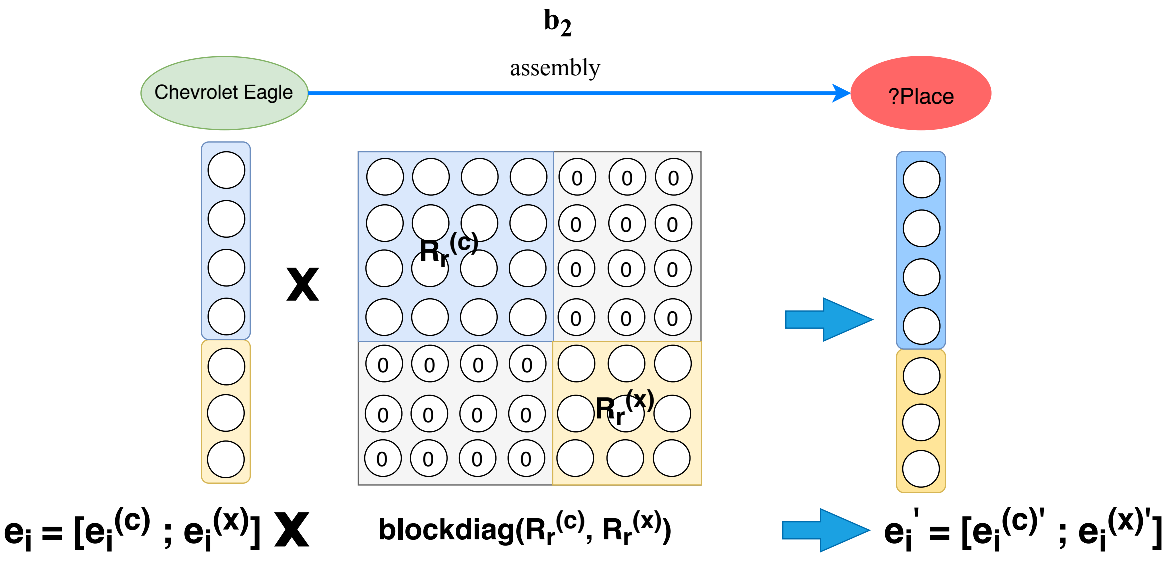

b₂
assembly
Chevrolet Eagle
?Place
R_r^(c)
R_r^(x)
0
e_i = [e_i^(c) ; e_i^(x)] X
blockdiag(R_r^(c), R_r^(x))
e_i' = [e_i^(c)' ; e_i^(x)']

Figure 4.7: An illustration of projection operator $\mathcal{P}^{(e)}()$ of *SE-KGE* with the input (e_i, r).

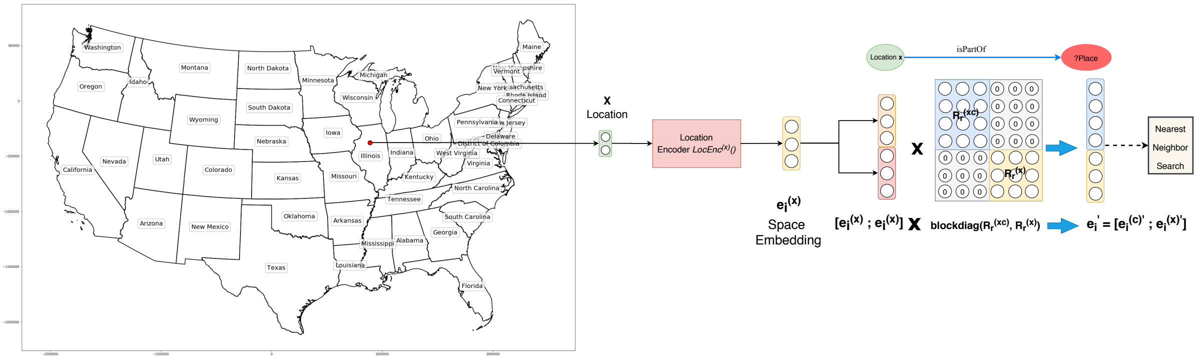


Figure 4.8: Spatial semantic lifting in the embedding space by using $Enc()$ and $\mathcal{P}^{(x)}()$

4.6.3 Geographic Logic Query Answering $\Phi_{\mathcal{G},\theta}(q)$ Model Training

We train the *SE-KGE* on both the original knowledge graph structure with an unsupervised objective $\mathcal{L}_{KG}$ and the query-answer pairs with a supervised objective $\mathcal{L}_{QA}$ (See

Equation 4.12):

$$\mathcal{L}^{(QA)} = \mathcal{L}_{KG} + \mathcal{L}_{QA} \tag{4.12}$$

Unsupervised KG Training Phase In this phase, we train *SE-KGE* components based on the local KG structure. In $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, for every entity $e_i \in \mathcal{V}$, we first obtain its 1-degree neighborhood $N(e_i) = \{(r_{ui}, e_{ui}) | r_{ui}(e_{ui}, e_i) \in \mathcal{G}\} \cup \{(r_{oi}^{-1}, e_{oi}) | r_{oi}(e_i, e_{oi}) \in \mathcal{G}\}$. We sample n tuples from $N(e_i)$ to form a sampled neighborhood $N_n(e_i) \subseteq N(e_i)$ and $|N_n(e_i)| = n$. We treat this subgraph as a conjunctive graph query with n basic graph patterns, in which entity e_i holds the target variable position. The model predicts the embedding of e_i such that the correct embedding $\mathbf{e}_i$ is the closest one to the predicted embedding $\mathbf{e}_i''$ against all embeddings $\mathbf{e}_i^-$ in negative sample set $Neg(e_i)$:

$$\mathcal{L}_{KG} = \sum_{e_i \in \mathcal{V}} \sum_{e_i^- \in Neg(e_i)} max(0, \Delta - \mathbf{\Omega}(\mathbf{H}_{KG}(e_i), \mathbf{e}_i) + \mathbf{\Omega}(\mathbf{H}_{KG}(e_i), \mathbf{e}_i^-)) \tag{4.13}$$

where

$$\mathbf{e}_i'' = \mathbf{H}_{KG}(e_i) = \mathcal{I}(\{\mathcal{P}^{(e)}(e_{ci}, r_{ci}) | (r_{ci}, e_{ci}) \in N_n(e_i)\}) \tag{4.14}$$

Here $\mathcal{L}_{KG}$ is a max-margin loss and Δ is the margin.

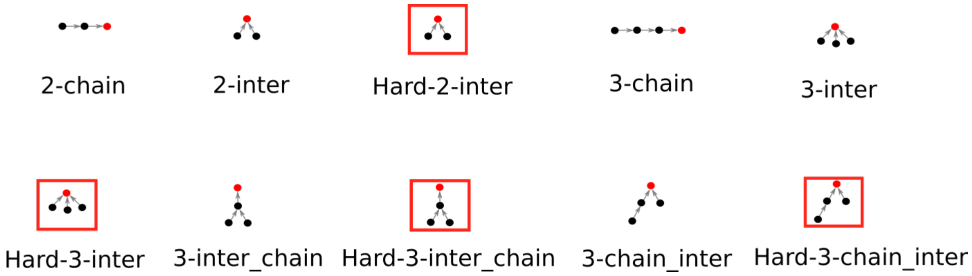


Figure 4.9: The DAG structures of the conjunctive graph queries we sampled from $\mathcal{G}$. Nodes indicates entities or variables and edges indicate basic graph patterns. The red node is the target variable of the corresponding query. the DAG structures surrounded by red boxes indicate queries sampled with hard negative sampling method.

Supervised Query-Answer Pair Training Phase We train *SE-KGE* by using conjunctive query-answer pairs. We first sample X different conjunctive graph query (logical query)-answer pairs $S = \{(q_i, a_i)\}$ from $\mathcal{G}$. We treat each entity as the target variable of a CQG and sample K queries for each DAG structure. All DAG structures we considered in this work are shown in Figure 4.9. The way to do query sampling is to sort the nodes in a DAG in a topological order and sample one basic graph pattern at one time by following this order and navigating on the $\mathcal{G}$ [83]. **In order to generate geographic conjunctive graph query, we have the restriction $e_i \in \mathcal{V}_{pt}$.**

The training objective is to make the correct answer entity embedding $\mathbf{a}_i$ be the closest one to the predicted query embedding $\mathbf{q}_i = \mathbf{\Phi}_{\mathcal{G},\theta}(q_i)$ against all the negative answers' embeddings $\mathbf{a}_i^-$ in negative answer set $Neg(q_i, a_i)$. We also use a max-margin loss:

$$\mathcal{L}_{QA} = \sum_{(q_i,a_i)\in S} \sum_{a_i^- \in Neg(q_i,a_i)} max(0, \Delta - \mathbf{\Omega}(\mathbf{q}_i, \mathbf{a}_i) + \mathbf{\Omega}(\mathbf{q}_i, \mathbf{a}_i^-)) \tag{4.15}$$

For $Neg(q_i, a_i)$ we compared two negative sampling strategies : 1) *negative sampling*: $Neg(q_i, a_i) \subseteq \mathcal{V}$ is a fixed-size set of entities such that $\forall e_i^- \in Neg(q_i, a_i),\ \Gamma(e_i^-) = \Gamma(e_i)\ and\ e_i^- \neq e_i$; 2) *hard negative sampling*: $Neg(q_i, a_i)$ is a fixed-size set of entities which satisfy some of the basic graph patterns b_{ij} (See Definition 6) in q_i but not all of them.

4.6.4 Spatial Semantic Lifting $\mathbf{\Psi}_{\mathcal{G},\theta_{ssl}}(\vec{x}, r)$ Model Training

We randomly select a point $\vec{x}_i \in \mathcal{A} \subseteq \mathbb{R}^2$ from the study area, and use location encoder $LocEnc^{(x)}$ to encode its location embedding $\mathbf{e}_i^{(x)} \in \mathbb{R}^{d^{(x)}}$. Since we do not have the feature embedding for this location, to make the whole model as an inductive learning one, we use $\mathcal{P}^{(x)}()$ to predict the tail embedding $\mathbf{e}' = \mathbf{\Psi}_{\mathcal{G},\theta_{ssl}}(\vec{x}_i, r)$ of this virtual triple $r(\vec{x}_i, e')$. This is equivalent to ask a query $r(\vec{x}_i, ?e)$ to $\mathcal{G}$. A nearest neighbor search in the entity embedding space will produce the predicted entity who can link to location $\vec{x}_i$ with relation r. Since given any location $\vec{x}_i$ from the study area, $\mathbf{\Psi}_{\mathcal{G},\theta_{ssl}}(\vec{x}_i, r)$ can predict the entity embedding that $\vec{x}_i$ can link to given relation r, this is a fully inductive learning based model. This model does not require location $\vec{x}_i$ to be selected from a predefined set of locations which is a requirement for transductive learning based models such as Kejriwal et al. [72]. Figure 4.8 shows the idea of spatial semantic lifting.

We train the spatial semantic lifting model *SE-KGE*$_{ssl}$ with $Enc()$, $\mathcal{P}^{(e)}()$, and $\mathcal{P}^{(x)}()$ by using two objectives: link prediction objective $\mathcal{L}_{LP}$ and spatial semantic lifting objective $\mathcal{L}_{SSL}$.

$$\mathcal{L}^{(SSL)} = \mathcal{L}_{LP} + \mathcal{L}_{SSL} \tag{4.16}$$

Link Prediction Training Phase The link prediction training phase aims at training the feature embeddings of each entity. For each triple $s_i = (h_i, r_i, t_i) \in \mathcal{T}$, we can use $Enc()$ and $\mathcal{P}^{(e)}()$ to predict the tail entity embedding given the head and relation - $\mathcal{P}^{(e)}(h_i, r_i)$ - or predict the head entity embedding given the tail and relation - $\mathcal{P}^{(e)}(t_i, r_i^{-1})$. Note that we have two separate $\mathcal{P}^{(e)}()$ for r_i and r_i^{-1}. Equation 4.17 shows the loss function where $Neg_t(e_i)$ is the set of negative entities who share the same type with entity e_i.

$$\begin{aligned}\mathcal{L}_{LP} = &\sum_{s_i=(h_i,r_i,t_i)\in\mathcal{T}} \sum_{t_i^-\in Neg_t(t_i)} max(0, \Delta - \mathbf{\Omega}(\mathcal{P}^{(e)}(h_i, r_i), \mathbf{t}_i) + \mathbf{\Omega}(\mathcal{P}^{(e)}(h_i, r_i), \mathbf{t}_i^-)) \\ &+ \sum_{s_i=(h_i,r_i,t_i)\in\mathcal{T}} \sum_{h_i^-\in Neg_t(h_i)} max(0, \Delta - \mathbf{\Omega}(\mathcal{P}^{(e)}(t_i, r_i^{-1}), \mathbf{h}_i) + \mathbf{\Omega}(\mathcal{P}^{(e)}(t_i, r_i^{-1}), \mathbf{h}_i^-))\end{aligned} \tag{4.17}$$

Spatial Semantic Lifting Training Phase We also directly optimize our model on the spatial semantic lifting objective. We denote $\mathcal{T}_s$ and $\mathcal{T}_o$ as sets of triples whose head (or tail) entities are geographic entities, i.e., $\mathcal{T}_s = \{s_i | s_i = (h_i, r_i, t_i) \in \mathcal{T} \wedge h_i \in \mathcal{V}_{pt}\}$ and $\mathcal{T}_o = \{s_i | s_i = (h_i, r_i, t_i) \in \mathcal{T} \wedge t_i \in \mathcal{V}_{pt}\}$. The training objective is to make the tail entity embedding $\mathbf{t}_i$ to be the closest one to the predicted embedding $\mathcal{P}^{(x)}(\mathcal{X}(h_i), r_i)$ against all negative entity embeddings $\mathbf{t}_i^-$. We do the same for the inverse triple (t_i, r_i^{-1}, h_i). The loss function is shown in Equation 4.18.

$$\begin{aligned}\mathcal{L}_{SSL} = & \sum_{s_i=(h_i,r_i,t_i)\in\mathcal{T}_s} \sum_{t_i^- \in Neg_t(t_i)} max(0, \Delta - \mathbf{\Omega}(\mathcal{P}^{(x)}(\mathcal{X}(h_i), r_i), \mathbf{t}_i) + \mathbf{\Omega}(\mathcal{P}^{(x)}(\mathcal{X}(h_i), r_i), \mathbf{t}_i^-)) \\ + & \sum_{s_i=(h_i,r_i,t_i)\in\mathcal{T}_o} \sum_{h_i^- \in Neg_t(h_i)} max(0, \Delta - \mathbf{\Omega}(\mathcal{P}^{(x)}(\mathcal{X}(t_i), r_i^{-1}), \mathbf{h}_i) + \mathbf{\Omega}(\mathcal{P}^{(x)}(\mathcal{X}(t_i), r_i^{-1}), \mathbf{h}_i^-))\end{aligned} \tag{4.18}$$

where

$$\mathcal{X}(e_i) = \begin{cases} \vec{x}_i = \mathcal{PT}(e_i) \ , & if\ e_i \in \mathcal{V}_{pt} \setminus \mathcal{V}_{pn} \\ \vec{x}_i^{(t)} \sim \mathcal{U}(\vec{x}_i^{min}, \vec{x}_i^{max}) \ , \ \mathcal{PN}(e_i) = [\vec{x}_i^{min}; \vec{x}_i^{max}] \ , & if\ e_i \in \mathcal{V}_{pn} \end{cases} \tag{4.19}$$

4.7 Experiment

To demonstrate how *SE-KGE* incorporates spatial information of geographic entities such as locations and spatial extents we experimented with two tasks – geographic logic query answering and spatial semantic lifting. To demonstrates the effectiveness of spatially explicit models and the importance to considering the scale effect in location encoding we select multiple baselines on the geographic logic query answering task. To show that *SE-KGE* is able to link a randomly selected location to entities in the existing KG with some relation, which none of the existing KG embedding models can solve, we proposed a new task - spatial semantic lifting.

4.7.1 *DBGeo* Dataset Generation

In order to evaluate our proposed location-aware knowledge graph embedding model *SE-KGE*, we first build a geographic knowledge graph which is a subgraph of DBpedia by following the common practice in KG embedding research [85, 86, 47]. We select the mainland of United States as the study area $\mathcal{A}$ since previous research [141] has shown that DBpedia has relatively richer geographic coverage in United States. The KG construction process is as follows:

1. We collect all the geographic entities within the mainland of United States as the seed entity set $\mathcal{V}_{seed}$ which accounts for 18,780 geographic entities[5]; We then collect their 1- and 2-degree object property triples with `dbo:` prefix predicates/relations[6];

2. We compute the degree of each entity in the collected KG and delete any entity, together with its corresponding triples, if its node degree is less than a threshold η. We use $\eta = 10$ for non-geographic entities and $\eta = 5$ for geographic entities, because many geographic entities, such as radio stations, have fewer object type property triples and a smaller threshold ensures that a relative large number of geographic entities can be extracted from the KG;

3. We further filter out those geographic entities that are newly added from Step 2 and are outside of the mainland of United States. The resulting triples form our KG, and we denote the geographic entity set as $\mathcal{V}_{pt}$.

4. We split $\mathcal{G}$ into training, validation, and testing triples with a radio of 90:1:9 so that every entity and relation appear in the training set. We denote the knowledge graph formed by the training triples as $\mathcal{G}_{train}$ while denoting the whole KG as $\mathcal{G}$.

[5] We treat an entity as a geographic entity if its has a `geo:geometry` triple in DBpedia

[6] `http://dbpedia.org/sparql?help=nsdecl`

5. We generate K conjunctive graph query-answer pairs from $\mathcal{G}$ for each DAG structure shown in Figure 4.9 based on the query-answer generation process we described in Section 4.6.3. $Q(\mathcal{G})$ and $Q(\mathcal{G})_{geo}$ indicate the resulting QA set while $Q_{geo}(\mathcal{G})$ indicates the geographic QA set. For each query q_i in training QA set, we make sure that each query is answerable based on $\mathcal{G}_{train}$, i.e., $\varphi(\mathcal{G}_{train}, q_i) \neq \emptyset$. As for query q_i in validation and testing QA set, we make sure each query q_i satisfies $\varphi(\mathcal{G}_{train}, q_i) = \emptyset$ and $\varphi(\mathcal{G}, q_i) \neq \emptyset$.

6. For each geographic entity $e \in \mathcal{V}_{pt}$, we obtain its location/coordinates by extracting its `geo:geometry` triple from DBpedia. We project the locations of geographic entities into US National Atlas Equal Area projection coordinate system (epsg:2163) $\mathcal{XY}$. $\mathcal{PT}(e) = \vec{x}$ indicates the location of e in the projection coordinate system $\mathcal{XY}$.

7. For each geographic entity $e \in \mathcal{V}_{pt}$, we get its spatial extent (bounding box) $\mathcal{PN}(e)$ in $\mathcal{XY}$ by using ArcGIS Geocoding API[7] and OpenStreetMap API. 80.6% of geographic entities are obtained. We denote them as $\mathcal{V}_{pn}$.

8. For each entity $e_i \in \mathcal{V}$, we obtain its types by using `rdf:type` triples. Note that there are entities having multiple types. We look up the DBpedia Ontology (class hierachy) to get their level-1 superclass. We find out that every entity in $\mathcal{G}$ has only one level-1 superclass type. Table 4.2 shows statistics of entities in different types.

9. To build the training/validation/testing datasets for spatial semantic lifting, we obtain $\mathcal{T}_s, \mathcal{T}_o \subseteq \mathcal{T}$ (See Section 4.6.4), each triple of which is composed of geographic entities as its head or tail. We denote $\mathcal{R}_{ssl} = \{r_i | s_i = (h_i, r_i, t_i) \in \mathcal{T}_s \cap \mathcal{T}_o\}$

[7] `https://geocode.arcgis.com/arcgis/rest/services/World/GeocodeServer/find`

Table 4.1: Statistics for our dataset in *DBGeo* (Section 4.7.1). "XXXX/QT" indicates the number of QA pairs per query type.

		DBGeo		
		Training	Validation	Testing
Knowledge Graph	$\vert\mathcal{T}\vert$	214,064	2,378	21,406
	$\vert\mathcal{R}\vert$	318	-	-
	$\vert\mathcal{V}\vert$	25,980	-	-
	$\vert\mathcal{V}_{pt}\vert$	18,323	-	-
	$\vert\mathcal{V}_{pn}\vert$	14,769	-	-
Geographic Question Answering	$\vert Q^{(2)}(\mathcal{G})\vert$	1,000,000	-	-
	$\vert Q^{(3)}(\mathcal{G})\vert$	1,000,000	-	-
	$\vert Q^{(2)}_{geo}(\mathcal{G})\vert$	1,000,000	1000/QT	10000/QT
	$\vert Q^{(3)}_{geo}(\mathcal{G})\vert$	1,000,000	1000/QT	10000/QT
Spatial Semantic Lifting	$\vert\mathcal{T}_s \cap \mathcal{T}_o\vert$	138,193	1,884	17,152
	$\vert\mathcal{R}_{ssl}\vert$	227	71	135

We denote $Q^{(2)}(\mathcal{G})$, $Q^{(3)}(\mathcal{G})$ as the general QA sets which contain 2 and 3 basic graph patterns, and similarly for $Q^{(2)}_{geo}(\mathcal{G})$, $Q^{(3)}_{geo}(\mathcal{G})$. Table 4.1 shows the statistics of the constructed $\mathcal{G}$, the generated QA sets, and the spatial semantic lifting dataset in *DBGeo*. Figure 4.10 shows the spatial distribution of all geographic entities $\mathcal{V}_{pt}$ in $\mathcal{G}$.

4.7.2 Evaluation on the Geographic Logic Query Answering Task

Baselines

In order to quantitatively evaluate *SE-KGE* on geographic QA task, we train *SE-KGE*$_{full}$ and multiple baselines on $\mathcal{G}$ in *DBGeo*. Compared to previous work [83, 117], the most

Table 4.2: Number of entities for each entity type in *DBGeo*

Entity Type	Number of Entities
dbo:Place	16,527
dbo:Agent	8,371
dbo:Work	594
dbo:Thing	179
dbo:TopicalConcept	134
dbo:MeanOfTransportation	104
dbo:Event	71

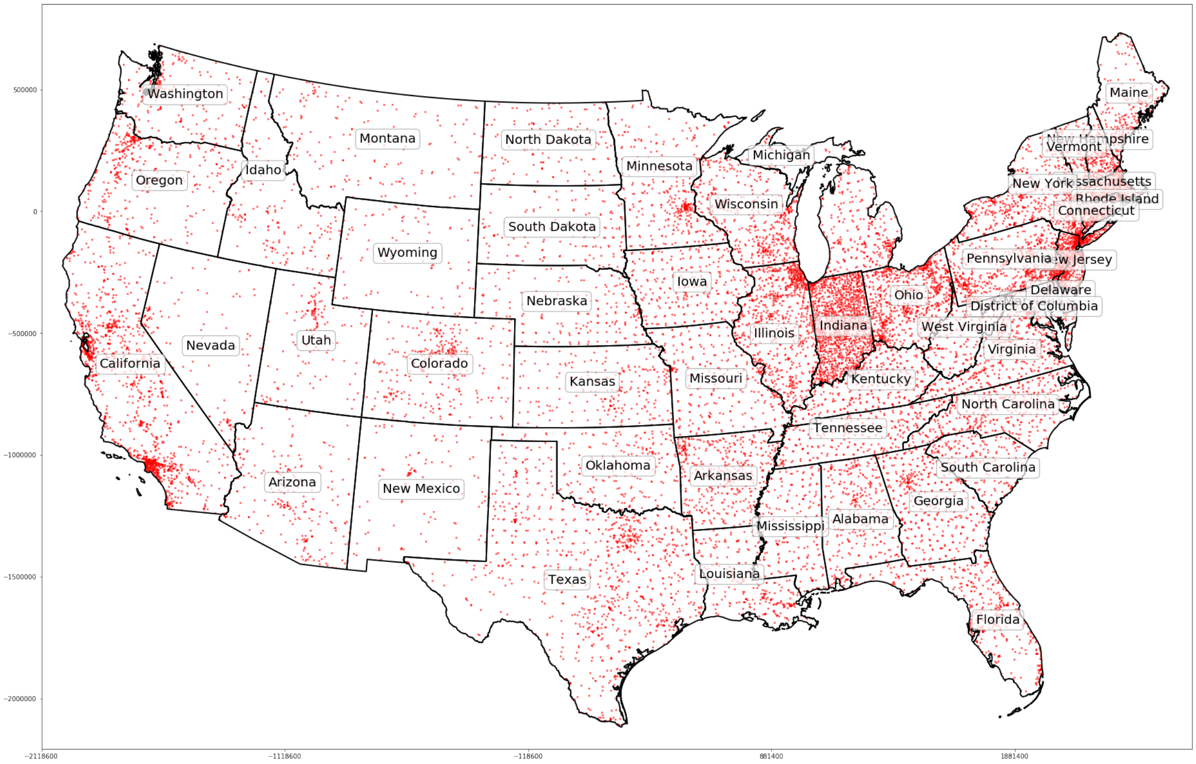


Figure 4.10: Spatial distribution of all geographic entities in $\mathcal{G}$

important contribution of this work is the entity space encoder $Enc^{(x)}()$ which makes our model spatially explicit. So we carefully select four baselines to test the contribution of $Enc^{(x)}()$ on the geographic logic QA task. We have selected four baselines:

1. GQE_{diag} and GQE: two versions of the logic query answering model proposed by Hamilton et al. [83] which have been discussed in detail in Section 4.5. The main different between GQE_{diag} and GQE is the projection operator they use: $\mathcal{P}^{(GQE_{diag})}$ and $\mathcal{P}^{(GQE)}$ accordingly. Compared with *SE-KGE*$_{full}$, both GQE_{diag} and GQE only use entity feature encoder $Enc^{(c)}()$ as the entity encoder and $\mathcal{I}^{(GQE)}$ as the intersection operator. Both methods only use $\mathcal{L}_{QA}$ in Equation 4.12 as the training objective. There two baselines are implemented based on the original code repository[8] of Hamilton et al. [83].

2. CGA: a logic query answering model proposed by Mai et al. [117] (See Section 4.5). Compared with *SE-KGE*$_{full}$, CGA uses different entity encoder ($Enc^{(CGA)}$) and projection operator ($\mathcal{P}^{(CGA)}$) such that the spatial information of each geographic entity is not considered. This baseline is used to test whether designing spatially explicit logic query answering model can outperform general models on the geographic query answering task.

3. *SE-KGE*$_{direct}$: a simpler version of *SE-KGE*$_{full}$ which uses a **single scale location encoder** in the entity encoder instead of the multi-scale periodic location encoder as shown in Equation 4.1 in Section 4.2.3. Instead of first decomposing input $\vec{x}$ into a multi-scale periodic representation by using sinusoidal functions with different frequencies [65], the location encoder of *SE-KGE*$_{direct}$ directly inputs $\vec{x}$ into a feed forward network. This single-scale location encoder is proposed in Mai et al. [65] as one baseline model - *direct*. Moreover, its entity space encoder does not consider

[8] `https://github.com/williamleif/graphqembed`

the spatial extent of each geographic entity either and just uses its coordinates to do location encoding. This baseline is used to test the effectiveness of using multi-scale periodical representation learning in our *SE-KGE* framework.

4. *SE-KGE*$_{pt}$: a simpler version of *SE-KGE*$_{full}$ whose entity space encoder does not consider the spatial extents of geographic entities. The only different between *SE-KGE*$_{pt}$ and *SE-KGE*$_{direct}$ is that *SE-KGE*$_{pt}$ uses $Space2Vec$ [65] as the location encoder while *SE-KGE*$_{direct}$ utilizes the single scale *direct* model as the location encoder. This baseline is used to test the necessity to consider the spatial extent of geographic entities in our *SE-KGE* framework. In other words, it uses Equation 4.20 for its space encoder:

$$\mathbf{e}_i^{(x)} = \begin{cases} LocEnc^{(x)}(\vec{x}_i) \ , where \ \vec{x}_i = \mathcal{PT}(e_i) \ , & if \ e_i \in \mathcal{V}_{pt} \\ \dfrac{\mathbf{Z}_x \mathbf{h}_i^{(x)}}{\parallel \mathbf{Z}_x \mathbf{h}_i^{(x)} \parallel_{L2}}, & if \ e_i \in \mathcal{V} \setminus \mathcal{V}_{pt} \end{cases} \tag{4.20}$$

5. *SE-KGE*$_{space}$: a simpler version of *SE-KGE*$_{full}$ whose entity encoder does not have the feature encoder component. This baseline is used to understand how the space encoder $Enc^{(x)}()$ captures the connectivity information of $\mathcal{G}$.

Training Details

We train our model *SE-KGE*$_{full}$ and six baselines on *DBGeo* dataset. GQE_{diag} and GQE are trained on the general QA pairs and geographic QA pairs as Hamilton et al. [83] did. The other models are additionally trained on the original KG structure. Gird search is used for hyperparameter tuning: $d = [32, 64, 128]$, $d^{(c)} = [16, 32, 64]$, $d^{(x)} = [16, 32, 64]$, $S = [8, 16, 32, 64]$, $\lambda_{min} = [10, 50, 200, 1000]$. The best performance is obtained when $d = 128$, $d^{(c)} = 64$, $d^{(x)} = 64$, $S = 16$, $\lambda_{min} = 50$. $\lambda_{max} = 5400000$ is determined by the study area $\mathcal{A}$. We also try different activation functions (i.e., Sigmoid, ReLU, LeakyReLU) for the full connected layers $\mathbf{NN}()$ of location encoder $LocEnc^{(x)}()$. We find out that *SE-KGE*$_{space}$ achieves the best performance with LeakyReLU as the activation function together with L2 normalization on the location embedding. *SE-KGE*$_{direct}$, *SE-KGE*$_{pt}$, and *SE-KGE*$_{full}$ obtain the best performance with Sigmoid activation function without L2 normalization on the location embedding. We implement all models in PyTorch and train/evaluate each model on a Ubuntu machine with 2 GeForce GTX Nvidia GPU cores, each of which has 10GB memory. The *DBGeo* dataset and related codes will be opensourced.

Evaluation Results

We evaluate *SE-KGE*$_{full}$ and six baselines on the validation and testing QA datasets of *DBGeo*. Each model produces a cosine similarity score between the predicted query embedding $\mathbf{q}$ and the correct answer embedding $\mathbf{a}$ (as well as the embedding of negative answers). The objective is to rank the correct answer top 1 among itself and all negative answers given their cosine similarity to $\mathbf{q}$. Two evaluation metrics are computed: Area Under ROC curve (AUC) and Average Percentile Rank (APR). AUC compares the correct answer with one random sampled negative answer for each query. An ROC curve is computed based on model performance on all queries and the area under this curve is obtained. As for APR, the percentile rank of the correct answer among all negative answers is obtained for each query based on the prediction of a QA model. Then APR is computed as the average of the percentile ranks of all queries. Since AUC only uses one negative sample per query while APR uses all negative samples for each query. We consider APR as a more robust evaluation metric.

Table 4.7.2 shows the evaluation results of *SE-KGE*$_{full}$ as well as six baselines on the validation and testing QA dataset of *DBGeo*. We split each dataset into different categories based on their DAG structures (See Figure 4.9). Note that logic query answering is a very challenging task. As for the two works which share a similar set up as ours, Hamilton et al. [83] show that their GQE model outperforms TransE baseline by 1.6% of APR on Bio dataset. Similarly, Mai et al. [117] demonstrate that their CGA model outperfroms GQE model by 1.39% and 1.65% of APR on DB18 and WikiGeo19 dataset. In this work, we show that our *SE-KGE*$_{full}$ outperforms the current state-of-the-art CGA model by 2.17% and 1.31% in terms of APR on the validation and testing dataset of *DBGeo* respectively. We regard it as a sufficient signal to show the effective of *SE-KGE*$_{full}$ on the geographic QA task. Some interesting conclusions can be drawn from Table 4.7.2:

1. CGA has a significant performance improvement over GQE_{diag} and GQE on *DBGeo*. This result is consistent with that of Mai et al. [117] which demonstrates the advantage of the self-attention mechanism in $\mathcal{I}^{(CGA)}$.

2. The performance of *SE-KGE*$_{direct}$ and CGA are similar, which shows that a simple single-scale location encoder (*SE-KGE*$_{direct}$) is not sufficient to capture the spatial information of geographic entities.

3. *SE-KGE*$_{full}$ performs better than *SE-KGE*$_{pt}$ which only considers the location information of geographic entities. This illustrates that scale effect is beneficial for the geographic logic QA task.

4. The performance of *SE-KGE*$_{space}$ is the worst among all models. This indicates that it is not enough to only consider spatial information as the input features for entity encoder $Enc()$. This makes sense because each entity in $\mathcal{G}$ has a lot of semantic information other than their spatial information, and only using spatial information for entity embedding learning is insufficient. However, *SE-KGE*$_{space}$ is a fully inductive learning model which enables us to do spatial semantic lifting.

5. Compared *SE-KGE*$_{full}$ with CGA, we can see that *SE-KGE*$_{full}$ outperforms CGA for almost all DAG structures on testing dataset except "Hard-3-chain_inter" (-0.58%) while top 2 DAG structures with the largest margin are "3-inter_chain" (2.15%) and "3-chain_inter" (2.08%). On the validation dataset, *SE-KGE*$_{full}$ gets higher ΔAPR compared to CGA on "Hard-3-inter_chain" (7.42%) and "3-inter_chain" (6.08%). GQE_{diag} shows the best performance on "Hard-3-chain_inter" query structure.

In order to demonstrate how the intersection operator $\mathcal{I}()$ helps to improve the model performance on the geographic QA task, we show *SE-KGE*$_{full}$'s predicted ranking list of entities on Query q_C as well as its three basic graph patterns in Table 4.7.2. These 12 entities in this table represent the hard negative sampling set of Query q_C. `dbr:Oakland,_California` is the correct answer for Query q_C. We can see that the top ranked four entities of b_1: $IsPartOf^{-1}(Alameda\ County, ?Place)$ are all subdivisions of Alameda County. The top ranked 5 entities of b_2: $Assembly(Chevrolet\ Eagle, ?Place)$ are all assembly places of Chevrolet Eagle. Similarly, the top ranked entities of b_3: $NearestCity(San\ Francisco\ Bay, ?Place)$ are close to San Francisco Bay. The full query q_C yield the best rank of the correct answer. This indicates that each basic graph pattern contributes to the query embedding prediction of *SE-KGE*$_{full}$. Moreover, to compare performances of different models on Query q_C, the percentile rank given by CGA, *SE-KGE*$_{pt}$, and *SE-KGE*$_{full}$ are 53.9%, 61.5%, and 77.0%, respectively.

We also test how well the location encoder $LocEnc^{(x)}()$ in *SE-KGE* can capture the global position information and how $LocEnc^{(x)}()$ interacts with other components of *SE-KGE*. We use *SE-KGE*$_{space}$ as an example. Since $LocEnc^{(x)}()$ is an inductive learning model, we divide the study area $\mathcal{A}$ into $20km \times 20km$ grids and take the location of each grid center as the input of $LocEnc^{(x)}()$. Each grid will get a $d^{(x)}$ dimension

Table 4.3: The evaluation of geographic logic query answering on *DBGeo* (using AUC (%) and APR (%) as evaluation metric)

	DAG Type	GQE_{diag}		GQE		CGA		$SE\text{-}KGE_{direct}$		$SE\text{-}KGE_{pt}$		$SE\text{-}KGE_{space}$		$SE\text{-}KGE_{full}$	
		AUC	APR	AUC	APR	AUC	APR	AUC	APR	AUC	APR	AUC	APR	AUC	APR
Valid	2-chain	63.37	64.89	84.23	**88.68**	84.56	86.8	83.12	84.79	**85.97**	84.9	76.81	67.07	85.26	87.25
	2-inter	97.23	97.86	96.00	97.02	98.87	98.58	98.98	98.28	98.95	98.52	85.51	87.13	**99.04**	**98.95**
	Hard-2-inter	70.99	73.55	66.04	73.83	73.43	79.98	73.27	76.36	**74.38**	82.16	63.15	62.91	73.42	**82.52**
	3-chain	61.42	67.94	79.65	79.45	79.11	80.93	77.92	79.26	79.38	83.97	70.09	60.8	**80.9**	**85.02**
	3-inter	98.01	99.21	96.24	98.17	99.18	**99.62**	**99.28**	99.41	99.1	99.56	87.62	89	99.27	99.59
	Hard-3-inter	78.29	85	68.26	77.55	79.59	86.06	79.5	84.28	**80.48**	**87.4**	63.37	67.17	78.86	85.2
	3-inter_chain	90.56	94.08	93.39	91.52	94.59	90.71	95.99	95.11	95.86	94.41	81.16	83.01	**96.7**	**96.79**
	Hard-3-inter_chain	74.19	**83.79**	70.64	74.54	73.97	76.28	74.81	78.9	**76.45**	75.95	65.54	68.21	76.33	83.7
	3-chain_inter	**98.01**	97.45	92.69	93.31	96.72	97.61	97.31	98.67	97.79	**98.76**	83.7	84.42	97.7	98.65
	Hard-3-chain_inter	**83.59**	**88.12**	66.86	74.06	72.12	77.53	73.23	79.24	74.74	80.47	65.13	69.29	74.72	78.11
	Full Valid	81.57	85.19	81.4	84.81	85.21	87.41	85.34	87.43	**86.31**	88.61	74.21	73.9	86.22	**89.58**
Test	2-chain	64.88	65.61	85	87.41	84.91	86.74	83.61	85.97	86.08	88.08	75.46	73.38	**86.35**	**88.12**
	2-inter	96.98	97.99	95.86	97.18	98.79	98.71	98.98	98.94	**98.98**	**99.08**	87.01	85.78	98.93	99.01
	Hard-2-inter	70.39	76.19	64.5	71.86	72.15	79.26	72.04	79.11	**73.72**	**81.78**	61.22	62.97	72.62	81.04
	3-chain	62.3	62.29	79.19	80.19	78.93	80.17	77.53	78.86	79.43	**81.28**	70.55	68.04	**80.49**	80.63
	3-inter	98.09	99.12	96.54	97.94	99.33	99.56	**99.45**	99.47	99.41	**99.63**	88.05	87.63	99.39	99.59
	Hard-3-inter	77.27	83.92	68.69	75.42	78.93	83.52	78.58	84.14	**80.11**	84.87	64.44	64.53	78.76	**84.89**
	3-inter_chain	90.39	91.96	92.54	93.13	93.46	94.36	95.23	95.92	95.02	95.78	81.52	79.61	**95.92**	**96.51**
	Hard-3-inter_chain	72.89	79.12	70.67	75.55	73.47	79.61	73.93	80.21	74.88	79.36	64.99	65.52	**75.36**	**80.72**
	3-chain_inter	97.35	98.27	92.22	94.08	96.55	96.67	97.29	98.39	**97.79**	98.68	85.28	84.08	97.64	**98.75**
	Hard-3-chain_inter	**83.33**	**86.24**	66.77	72.1	72.31	77.89	73.55	77.08	75.19	77.42	65.07	65.41	74.62	77.31
	Full Test	81.39	84.07	81.2	84.49	84.88	87.65	85.02	87.81	**86.06**	88.2	74.36	73.7	86.01	**88.96**

Table 4.4: The rank of entities in the hard negative sample set of Query q_C based on *SE-KGE*$_{full}$'s prediction for different queries: 1) b_1: $IsPartOf^{-1}(Alameda\ County, ?Place)$; 2) b_2: $Assembly(Chevrolet\ Eagle, ?Place)$; 3) b_3: $NearestCity(San\ Francisco\ Bay, ?Place)$; 4) The Full Query q_C. The correct answer is highlighted as bold.

	b_1	b_2	b_3	Query q_C
1	dbr:Emeryville,_California	dbr:Flint_Truck_Assembly	dbr:Alameda,_California	dbr:San_Jose,_California
2	dbr:Castro_Valley,_California	dbr:Norwood,_Ohio	dbr:San_Jose,_California	**dbr:Oakland,_California**
3	dbr:Alameda,_California	dbr:Flint,_Michigan	dbr:Berkeley,_California	dbr:Berkeley,_California
4	dbr:Berkeley,_California	dbr:Tarrytown,_New_York	**dbr:Oakland,_California**	dbr:Alameda,_California
5	dbr:San_Jose,_California	**dbr:Oakland,_California**	dbr:Emeryville,_California	dbr:San_Francisco
6	dbr:Fremont,_California	dbr:Alameda,_California	dbr:Fremont,_California	dbr:Fremont,_California
7	**dbr:Oakland,_California**	dbr:San_Jose,_California	dbr:Norwood,_Ohio	dbr:Emeryville,_California
8	dbr:San_Francisco	dbr:San_Francisco	dbr:San_Francisco	dbr:Castro_Valley,_California
9	dbr:Norwood,_Ohio	dbr:Berkeley,_California	dbr:Flint_Truck_Assembly	dbr:Flint,_Michigan
10	dbr:Flint_Truck_Assembly	dbr:Fremont,_California	dbr:Flint,_Michigan	dbr:Norwood,_Ohio
11	dbr:Tarrytown,_New_York	dbr:Emeryville,_California	dbr:Castro_Valley,_California	dbr:Flint_Truck_Assembly
12	dbr:Flint,_Michigan	dbr:Castro_Valley,_California	dbr:Tarrytown,_New_York	dbr:Tarrytown,_New_York

location embedding after location encoding. We apply hierarchical clustering on these embeddings. Figure 4.11a shows the clustering result. We compare it with the widely used USA Census Bureau-designated regions[9] (See Figure 4.11b). We can see that Figure 4.11a and 4.11b look very similar to each other. We use two clustering evaluation metrics - Normalized Mutual Information (NMI) and Rand Index - to measure the degree of similarity which yield 0.62 on NMI and 0.63 on Rand Index. To take a closer look at Figure 4.11a, we can also see that the clusters are divided on the state borders. We hypothesize that this is because $LocEnc^{(x)}()$ is informed of the connectivity of different geographic entities in $\mathcal{G}$ during model training, resulting in that locations which are connected in original $\mathcal{G}$ are also clustered after training.

To validate this hypothesis, we apply Louvain community detection algorithm with a shuffled node sequence[10] on the original $\mathcal{G}$ by treating $\mathcal{G}$ as an undirected and unlabeled graph. Figure 4.11c shows the community structure with the best modularity which contains 32 communities. Some interesting observations can be made by comparing these three figures:

1. Most communities in Figure 4.11c are separated at state borders, which is an evidence of our hypothesis;

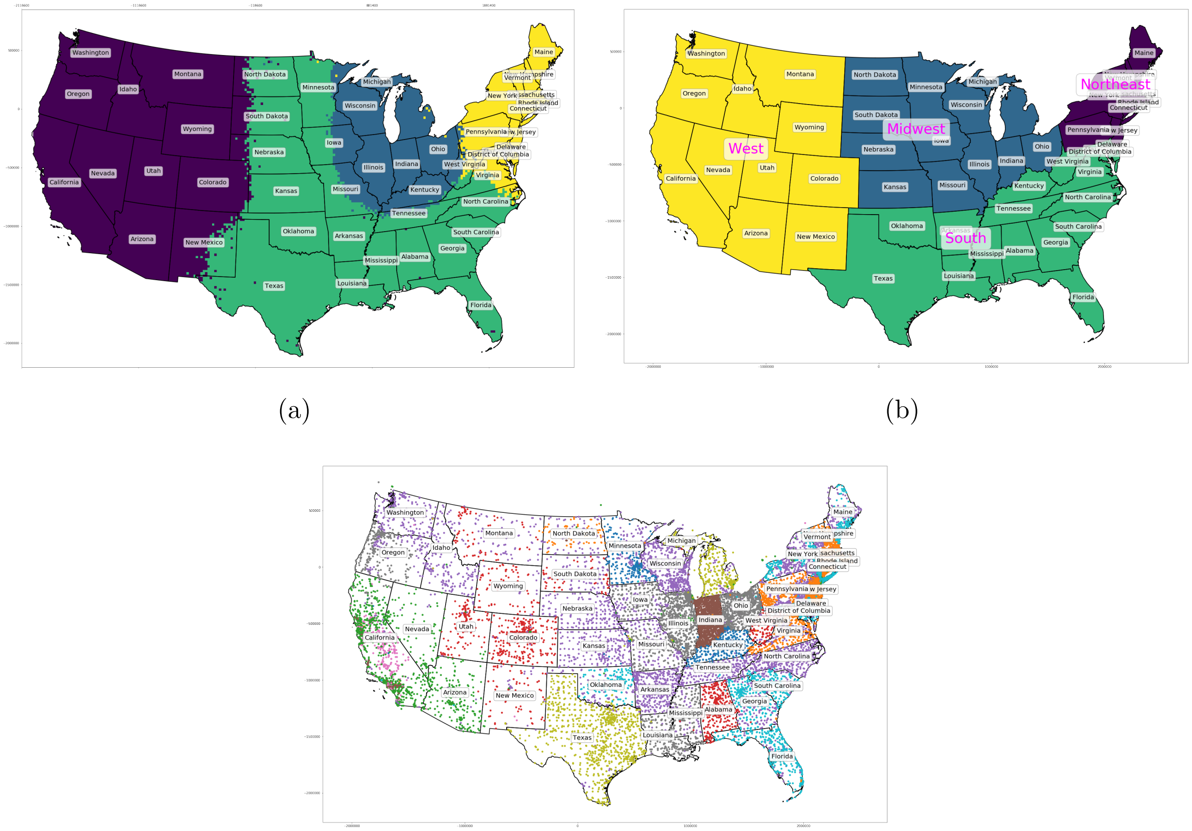
2. Some communities contain locations at different states, which are far away from each other. For example, the red community which contains locations from Utah, Colorado, and Alabama. This indicates that some locations are very similar purely based on the graph structure of $\mathcal{G}$. As $LocEnc^{(x)}()$ imposes spatial constraints on entities, spatially coherent clusters in Figure 4.11a are presented.

One hypothesis why Figure 4.11a and 4.11b look similar is that in the KG, the number of connections between entities within one Bureau-designated region is more than the

[9] `https://en.wikipedia.org/wiki/List_of_regions_of_the_United_States`

[10] `https://github.com/tsakim/Shuffled_Louvain`

number of connections among entities in different regions. This may be due to the fact that DBpedia uses census data as one of the data sources while census data is organized in a way which reflects Bureau-designated regions of the US. More research is needed to validate this hypothesis in the future.



(a)

(b)

(c)

Figure 4.11: (a) Clustering result of location embeddings produced by the location encoder $LocEnc^{(x)}()$ in $SE\text{-}KGE_{space}$. It illustrates spatial coherence and semantics (b) Census Bureau-designated regions of United States, and (c) the community detection (Shuffled Louvain) results of knowledge graph $\mathcal{G}$ by treating $\mathcal{G}$ as a undirected unlabeled multigraph. It lacks spatial coherence.

4.7.3 Evaluation on Spatial Semantic Lifting Task

Baselines

The spatial semantic lifting model is composed of $Enc()$, $\mathcal{P}^{(e)}()$, and $\mathcal{P}^{(x)}()$ which is indicated as *SE-KGE*$_{ssl}$. In order to study the contribution of feature encoder and location encoder, we create a baseline *SE-KGE*$'_{space}$ whose entity encoder does not have the feature encoder component, similar to *SE-KGE*$_{space}$. The difference is that they are trained on different objectives. These are the only two models that can do spatial semantic lifting task, since they are fully inductive learning models directly using locations as the only input features.

Training Detail

We train *SE-KGE*$_{ssl}$ and *SE-KGE*$'_{space}$ based on $\mathcal{L}^{(SSL)}$. To quantitatively evaluate them on spatial semantic lifting task, we use $\mathcal{T}_s \cap \mathcal{T}_o$ in the validation and testing dataset with different relations (See Table 4.1). For each triple $s_i = (h_i, r_i, t_i) \in \mathcal{T}_s$, given the head entity's location and r_i, we use $\mathcal{P}^{(x)}(\mathcal{X}(h_i), r_i)$ (See Equation 4.19) to predict the tail entity embedding. Similar process can be done for $s_j = (h_j, r_j, t_j) \in \mathcal{T}_o$ but from the reverse direction. We also use AUC and APR as the evaluation metrics. Note that since $\mathcal{X}(h_i) = \vec{x}_i^{(t)} \sim \mathcal{U}(\vec{x}_i^{min}, \vec{x}_i^{max})$, $\mathcal{PN}(h_i) = [\vec{x}_i^{min}; \vec{x}_i^{max}]\ if\ h_i \in \mathcal{V}_{pn}$, the location of head entity is randomly generated, which can be treated as unseen in the training process. We use the same hyperparameter configuration as *SE-KGE*$_{full}$.

Evaluation Results

Table 4.5 shows the overall evaluation results. We can see that *SE-KGE*$_{ssl}$ outperforms *SE-KGE*$'_{space}$ with a significant margin (ΔAUC = 9.86% and ΔAPR = 9.59% on the testing dataset) which clearly shows the strength of considering both feature embedding

Table 4.5: The evaluation of spatial semantic lifting on *DBGeo* over all validation/testing triples

	SE-KGE$_{space}$		*SE-KGE*$_{ssl}$		*SE-KGE*$_{ssl}$ - *SE-KGE*$_{space}$	
	AUC	APR	AUC	APR	ΔAUC	ΔAPR
Valid	72.85	75.49	**82.74**	**85.51**	9.89	10.02
Test	73.41	75.77	**83.27**	**85.36**	9.86	9.59

and space embedding in spatial semantic lifting task.

Next, among all validation and testing triples with different relations, we select a few relations and report APR of two models on these triples with specific relations. The results are shown in Table 4.6. These relations are selected since they are interesting from spatial reasoning perspective. We can see that *SE-KGE*$_{ssl}$ outperforms *SE-KGE*$'_{space}$ on all these triple sets with different relations.

In order to know how well *SE-KGE*$_{ssl}$ understands the semantics of different types of (spatial) relations, we visualize the spatial semantic lifting results in Figure 4.12 for four spatial relations: `dbo:state`, `dbo:nearestCity`, , `dbo:broadcastArea`$^{-1}$, and `dbo:isPartOf`. `dbo:state`, `dbo:isPartOf`, and `dbo:broadcastArea`$^{-1}$ are about partonomy relations while `dbo:nearestCity` represents an example of point-wise metric spatial relations. Some interesting observations can be made:

1. *SE-KGE*$_{ssl}$ is capable of capturing the spatial proximity such that the top 1 geographic entity (yellow point) in each case is the closest to location $\vec{x}$ (red triangle). We also treat this as an indicator for the capability of *SE-KGE*$_{ssl}$ to handle partonomy relations and point-wise metric spatial relations.

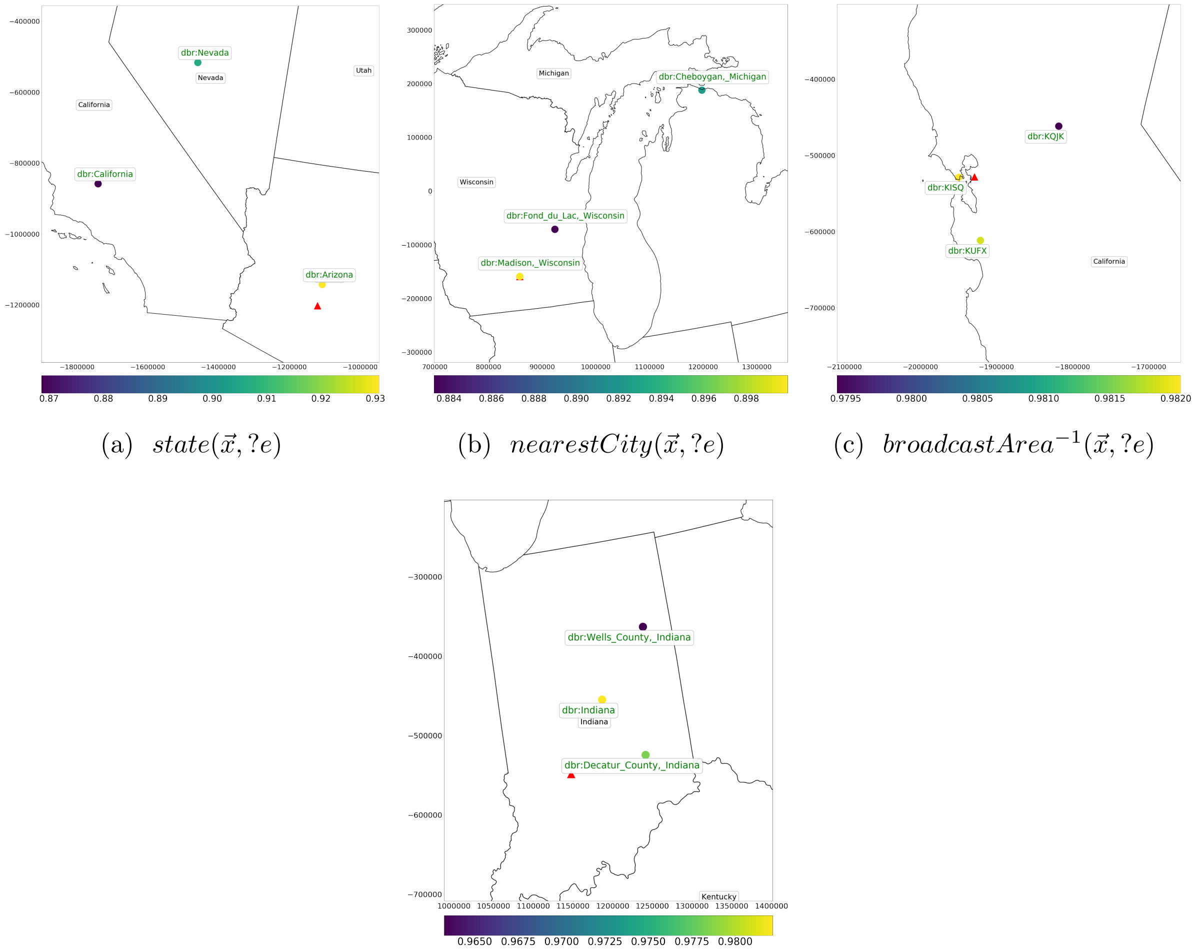
2. *SE-KGE*$_{ssl}$ can capture the semantics of relations, e.g., the domain and range of each relation/predicate. All top ranked entities are within the range of the cor-

Table 4.6: The evaluation of $\mathit{SE\text{-}KGE}_{ssl}$ and $\mathit{SE\text{-}KGE}'_{space}$ on *DBGeo* for a few selected relation r (using APR (%) as evaluation metric).

	Query Type	$\mathit{SE\text{-}KGE}'_{space}$	$\mathit{SE\text{-}KGE}_{ssl}$	ΔAPR
Valid	$state(\vec{x}, ?e)$	92.00	**99.94**	7.94
	$nearestCity(\vec{x}, ?e)$	84.00	**94.00**	10.00
	$broadcastArea^{-1}(\vec{x}, ?e)$	91.60	**95.60**	4.00
	$isPartOf(\vec{x}, ?e)$	88.56	**98.88**	10.32
	$locationCity(\vec{x}, ?e)$	83.50	**99.00**	15.50
	$residence^{-1}(\vec{x}, ?e)$	90.50	**93.50**	3.00
	$hometown^{-1}(\vec{x}, ?e)$	61.14	**74.86**	13.71
Test	$state(\vec{x}, ?e)$	89.06	**99.97**	10.91
	$nearestCity(\vec{x}, ?e)$	87.60	**99.80**	12.20
	$broadcastArea^{-1}(\vec{x}, ?e)$	90.81	**96.63**	5.82
	$isPartOf(\vec{x}, ?e)$	87.66	**98.87**	11.21
	$locationCity(\vec{x}, ?e)$	84.80	**99.10**	14.30
	$residence^{-1}(\vec{x}, ?e)$	61.21	**77.68**	16.47
	$hometown^{-1}(\vec{x}, ?e)$	61.44	**76.83**	15.39

responding relation. For example, in Figure 4.12a with query $state(\vec{x}, ?e)$, the top 3 entities are all states spatially closed to $\vec{x}$. In Figure 4.12c with query $broadcastArea^{-1}(\vec{x}, ?e)$, all top 3 entities are nearby radio stations. In Figure 4.12d with query $isPartOf(\vec{x}, ?e)$, all top 3 entities are states (`dbo:Indiana`) and counties.

3. We notice that the result of query $nearestCity(\vec{x}, ?e)$ in Figure 4.12b is not good enough since the second result - `dbo:Cheboygan,_Michigan` - is outside of Wisconsin. After investigating the triples with `dbo:nearestCity` as the relation, we



(a) $state(\vec{x}, ?e)$ (b) $nearestCity(\vec{x}, ?e)$ (c) $broadcastArea^{-1}(\vec{x}, ?e)$

(d) $isPartOf(\vec{x}, ?e)$

Figure 4.12: The visualization of spatial semantic lifting of *SE-KGE*$_{ssl}$. Figure (a), (b), (c), and (d) shows the top 3 geographic entities which can answer query $r(\vec{x}, ?e)$ where r is the relation we pick. **Red triangle**: the select location $\vec{x}$. **Circles**: top 3 geographic entities ranked by our model, and their colors indicates cosine similarity between the geographic entities and the predicted query embedding.

find out `dbo:nearestCity` usually links a natural resource entity (e.g., lakes, national parks) to a city. These natural resource entities usually cover large area and complex geometries. So `dbo:nearestCity` is not a purely point-wise distance base relation but a complex distance base relation based on their real geometries. Since our model only takes the bounding box of each entity and there are usually no subdivions of these nature resource entities, it is hard for our model to learn the semantics of `dbo:nearestCity`.

Based on the evaluation results and model analysis, we can see that given a relation r, *SE-KGE*$_{ssl}$ is able to link a location $\vec{x}$ to an entity e in $\mathcal{G}$ by considering the semantics of r and spatial proximity.

4.8 Conclusion

In this work, we propose a location-aware knowledge graph embedding model called *SE-KGE* which enables spatial reasoning in the embedding space for its three major components - entity embedding encoder $Enc()$, projection operator $\mathcal{P}()$, and intersection operator $\mathcal{I}()$. We demonstrate how to incorporate spatial information of geographic entities such as locations and spatial extents into $Enc()$ such that *SE-KGE* can handle different types of spatial relations such as point-wise metric spatial relations and partonomy relations. To the best of our knowledge, this is the first KG embedding model which incorporates location encoding into the model architecture instead of relying on some form of distance measure among entities while capturing the scale effect of different geographic entities. Two tasks have been used to evaluate the performance of *SE-KGE* - geographic logic query answering and spatial semantic lifting. Results show that *SE-KGE*$_{full}$ can outperform multiple baselines on the geographic logic query answering task which indicates the effectiveness of spatially explicit models. It also demonstrates the importance to considering the scale effect in location encoding. Also we proposed a new task - spatial semantic lifting, aiming at linking a randomly selected location to entities in the existing KG with some relation. None of the existing KG embedding models can solve this task except our model. We have shown that *SE-KGE*$_{ssl}$ can significantly outperform the baseline *SE-KGE*$'_{space}$ (ΔAUC = 9.86% and ΔAPR = 9.59% on the testing dataset).

Visualizations show that *SE-KGE$_{ssl}$* can successfully capture the spatial proximity information as well as the semantics of relations. In the future, we hope to explore a more concise way to encode the spatial footprints of geographic entities in a KG. Moreover, we want to explore more varieties of the spatial semantic lifting task.

Acknowledgement This work was partially supported by the NSF award 1936677 *Spatially-Explicit Models, Methods, and Services for Open Knowledge Networks* as well as Esri Inc..

Chapter 5

Representation Learning for Complex Polygonal Geometries in the Spectral Domain based on Non-Uniform Fourier Transformation

Many geographic questions such as topological relation questions, cardinal direction relation questions, cannot be correctly answered if a GeoQA model is only aware of the point coordinates and the bounding boxes of geographic entities. So representing polygonal geometries such as simple polygons, polygons with holes, and multipolygons into the embedding space is also critical for a GeoQA system. In fact, polygon encoding is a general problem required by many polygon-based geospatial tasks (e.g., building pattern classification, cartographic building generalization) that goes beyond the context of GeoQA. However, encoding complex polygonal geometries into the embedding space is a non-trivial problem since there are four properties we expect for a polygon encoder: loop invariance, trivial vertex invariance, part permutation invariance, and topology awareness. This chapter mainly focuses on the polygon encoding problem and proposes a Non-Uniform Fourier Transformation-based polygon encoder that satisfies all these properties. In addition, we also propose a 1D ResNet-based polygon encoder to handle simple polygons which can achieve loop invariance. Experiment results on two real-world datasets show that our polygon encoder can outperform multiple baselines on the polygon-based spatial relation prediction task which is an important but missing component of our current GeoQA toolset.

Peer Reviewed Publication	
Title	Representation Learning for Complex Polygonal Geometries in the Spectral Domain based on Non-Uniform Fourier Transformation
Authors	Gengchen Mai, Chiyu Max Jiang, Weiwei Sun, Rui Zhu, Yao Xuan, Ling Cai, Krzysztof Janowicz, Ni Lao
Venue	The Thirty-fifth Conference on Neural Information Processing Systems (NeurIPS 2021)
Editors	Marc'Aurelio Ranzato, Alina Beygelzimer, Percy Liang, Jenn Wortman Vaughan, Yann Dauphin
Publisher	openreview.net
Pages	Under Review
Submit Date	May 02, 2021
Accepted Date	Under Review
Publication Date	Under Review
Copyright	Under Review

Abstract: Developing deep models for data with irregular (in non-Euclidean space)

structures is difficult, but rewarding because these data are heavily used in many scientific disciplines. In this work, we focus on the problem of representation learning on (complex) polygonal geometries given their wide applications in different domains such as spatial relation prediction, geographic question answering, building pattern classification, building shape coding, cartographic building generalization, and so on. We aim to design a polygon encoder satisfying four nice properties: loop invariance, trivial vertex invariance, part permutation invariance, and topology awareness. We develop a polygon encoder called NUFTspec based on the spectral features of a polygonal geometry after a Non-Uniform Fourier Transformation (NUFT). NUFTspec learns the polygon embedding in the spectral domain and naturally satisfies all four properties. We also propose a 1D CNN-based polygon encoder, ResNet1D, which encodes a simple polygon directly from the spatial domain and can achieve loop invariance, but only on handling simple polygons. To show the effectiveness of NUFTspec and ResNet1D, we construct two datasets for the polygon-based spatial relation prediction task - DBSpaRel46K and DBSpaRelComplex46K - based on OpenStreetMap and DBpedia. Evaluation results show that NUFTspec and ResNet1D outperform multiple existing baselines with a significant margin when encoding simple polygons. Moreover, NUFTspec outperforms the previous DDSL model on DBSpaRelComplex46K and is more robust with respect to the number of sampled frequencies in the spectral domain.

5.1 Introduction

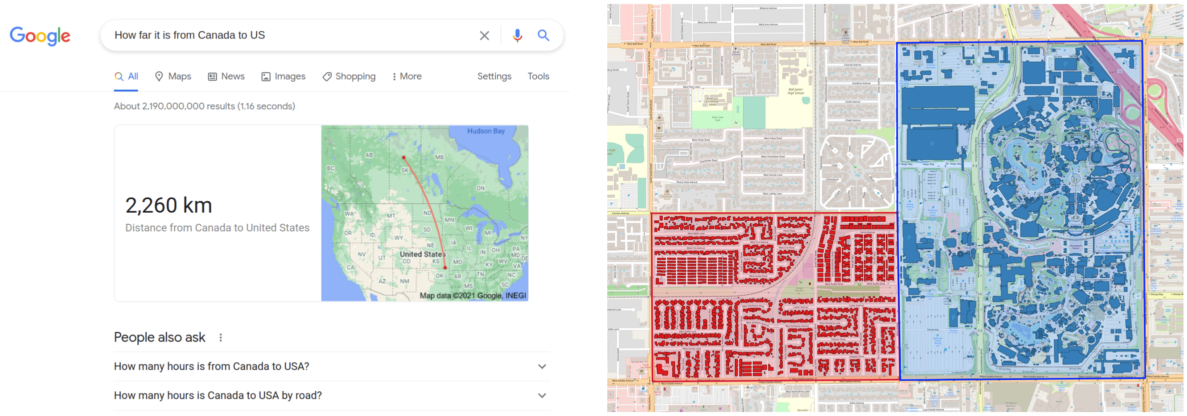
Deep neural networks have shown great success for numerous tasks from computer vision, natural language processing, to audio analysis whose underlining data is usually in regular structure such as grid-like (e.g., images) or sequence-like (e.g., sentences, audios) [142]. These successes can be largely attributed to the fact that the regularity of these structures are built into the neural networks [142] which make it hard to apply similar models on data with irregular structures. Recent years have witnessed growing interests in geometric deep learning [142, 143] which focuses on developing deep models for non-Euclidean geometric data such as graphs [144, 145, 136, 127, 128, 66], points [95, 146, 64, 65], and manifolds [147, 143] which have rather irregular structure. In fact, deep learning models on irregularly structured data or non-Euclidean geometric data have various applications in different domains such as computational social science (e.g., social network [148, 149]), chemistry (e.g., organic molecules [150]), bioinformatics (e.g., gene regulatory network [151]), and geoscience (e.g., traffic network [152, 153], air quality sensor network [154], weather sensor networks [155], and species occurrences [64, 65]).

Compared with other non-Euclidean geometric data, few efforts have been taken to develop deep models on polygons despite the fact that polygon data are widely utilized in multiple applications, especially geospatial applications such as geographic question answering (GeoQA) [10, 31, 47, 66], building pattern classification (BPC) [156], building shape coding [157], cartographic building generalization [158], and so on. Figure 5.1 shows the usefulness of polygon data for two geospatial tasks - GeoQA and BPC. Without proper polygon representations of Canada and the US (Figure 5.1a), Question '*How far it is from Canada to US*' cannot be answered correctly[1] even with the state-of-the-art QA system. As for the BPC task (Figure 5.1b), the shape and arrangement of building polygons in a neighborhood are indicative for its function/land use.

Most existing deep models on polygons focus on constructing a simple polygon[2] from an image as the object mask for the object instance segmentation task [159, 160, 161]. In contrast, in this work, we focus on **a general-purpose polygon encoding model**

[1] The answer to this brain teaser question should be 0 because Canada and the US are adjacent to each other. However, since Google utilizes geometric central points as the spatial representations for geographic entities, Google QA returns 2260 km as the answer as the distance between them.

[2] A simple polygon is a polygon that does not intersect itself and has no holes.



How far it is from Canada to US

About 2,190,000,000 results (1.16 seconds)

2,260 km

Distance from Canada to United States

People also ask

How many hours is from Canada to USA?

How many hours is Canada to USA by road?

(a) Geographic Question Answering

(b) Building Pattern Classification

Figure 5.1: Two geospatial tasks which shows the usability of polygon data: (a) GeoQA: many geographic questions can only be answered correctly based on polygon representations of geographic entities. Otherwise, it will yield an incorrect answer (2260 km) such as '*How far it is from Canada to US*'. (b) BPC: the shape, scale, and arrangement of building footprints in a neighborhood are very indicative for the function/affordance/land use of this neighborhood. The blue neighborhood (Disneyland Park in Los Angles) indicates a recreation area while the red neighborhood are a residential area given its more regularly arranged buildings.

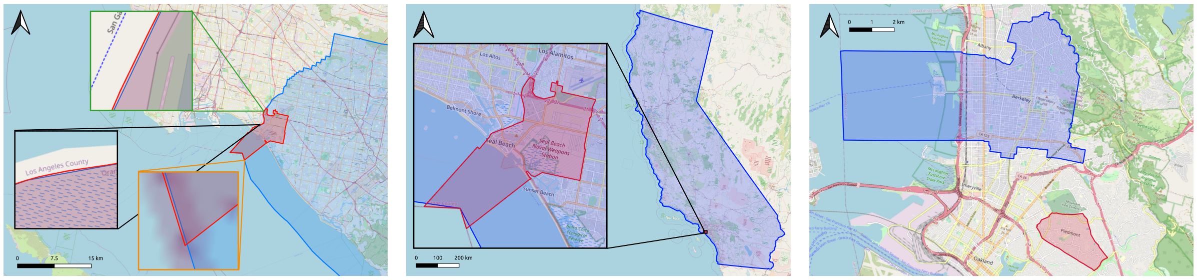
which aims at representing a polygonal geometry (with or without holes, single or multipolygons) into the embedding space. The resulting polygon embedding can be subsequently utilized in multiple downstream tasks such as spatial relation prediction between geographic entities [39], GeoQA [31], shape classification [162, 163, 164, 157], and so on. Given the unique structure of polygons, some important properties are desirable for the polygon encoder: *loop invariance*, *trivial vertex invariance*, *part permutation invariance*, and *topology awareness*. These four properties are illustrated in Figure 5.3 and will be discussed in detail in Section 5.3. We propose a polygon encoding model, *NUFTspec*, which first transforms the polygonal geometry into the spectral domain by the Non-Uniform Fourier Transformation (NUFT) and then directly learns polygon embeddings from these spectral features. Because of the nature of NUFT, NUFTspec directly satisfies these four polygon encoding properties. In addition, in order to explore the potential of encoding polygons directly from the spatial domain, we also propose a 1D CNN based polygon encoder - ResNet1D which can encode simple polygons and achieve loop invariance.

To show the effectiveness of our polygon encoders, we compare our models with various deterministic or deep learning baselines on the polygon-based spatial relation prediction task which is an important component for GeoQA. Two real-world dataset DBSpaRel46K and DBSpaRelComplex46K are constructed for evaluation based on DBpedia and OpenStreetMap. We are able to show that ResNet1D can outperform other polygon encoders except NUFTspec on DBSpaRel46K because of its specially designed KDelta neighborhood point encoder and zero padding structure. Our NUFTspec model is able to outperform multiple baselines on both datasets because it can effectively learns robust polygon embeddings from the spectral domain derived from NUFT. Moreover, unlike most polygon encoders including ResNet1D which can only handle simple polygons, NUFTspec can deal with complex polygonal geometries.

This paper is organized as follow: We discuss the necessity of polygon encoding in Section 5.2. Then, in Section 5.3, we define the problem of representation learning on polygons and discuss four expected polygon encoding properties. Related work are discussed in Section 5.4. We present ResNet1D and NUFTspec polygon encoders in Section 5.5 and compare model properties in Section 5.5.4. Datasets and experiment results are discussed in Section 5.6 and we conclude this work in Section 5.7.

5.2 The Necessity of Polygon Encoding

We would like to first discuss the necessity of polygon encoding specifically for geospatial tasks such as spatial relation prediction. For example, a GIScience expert may ask why we need to represent polygons into the embedding space to do spatial relation prediction given the fact that we have a set of well-defined deterministic spatial operators for spatial relation computation based on region connection calculus (RCC8) [165] such as those implemented by PostGIS or GeoSPARQL [166]. A computer vision researcher might also question this idea by suggesting an alternative approach that first rasterizes two polygons under consideration into two images that share the same bounding box so that a convolutional neural network (CNN) model can be applied on them for relation prediction.



(a) Sliver Polygon Problem (b) Scale Problem (c) Vagueness of Spatial Relation

Figure 5.2: Three real-world examples as illustrations for there different problems when we predict spatial relations between two polygons: (a) Sliver Polygon Problem: *dbr:Seal_Beach,_California* (the red polygon) should be tangential proper part (TPP) of *dbr:Orange_County,_California* (the blue polygon).However, because of those sliver polygons shown in these zoom-in windows, a deterministic spatial reasoner such as GeoSPARQL will return *partially overlapping* as the result. (b) Scale Problem: *dbr:Seal_Beach,_California* (the red polygon) has a extremely smaller map scale compared with *dbr:California* (the blue polygon). On one hand, a deterministic reasoner will pay too much attention to the geometry detail and predict wrong relations because of the sliver polygons as Fig (a). On the other hand, if we convert these polygons into two images (e.g., two 128 × 128 images), the red polygon becomes too small and cannot cover even one pixel which will also affect the result. (c) Vagueness of spatial relation: *dbr:Berkeley,_California* (the blue polygon) sits in the north of *dbr:Piedmont,_California* (the red polygon) according to DBpedia and Wikipedia. However, only based on their polygon representation, their cardinal direction is vague and subjective which can be "north" or "northwest".

There are three different problems related to the spatial relation prediction task - *sliver polygon problem*, *scale problem*, and *vagueness of spatial relation* - which show the limitation of the above mentioned alternative approaches. Figure 5.2 illustrates these problems by using three real-world examples from OpenStreetMap. First, as shown in those zoom-in windows of Figure 5.2a, those three tiny polygons yielded from the geometry difference between the red and blue polygon are called sliver polygons[3]. Here, *dbr:Seal_Beach,_California* (the red polygon) should be tangential proper part (TPP) of *dbr:Orange_County,_California* (the blue polygon). However, because of map digitization error, the boundary of the red polygon is crunching out the boundary of the blue polygon. A deterministic spatial operator will return "intersect" instead of "part of" as their relation. Sliver polygons are very common in map data, hard to prevent, and require a lot of efforts to correct, while deterministic spatial operators are very sensitive to them. As shown in Table 5.2, as for those 27,364 polygon pairs annotated with *dbo:isPartOf* relation by DBpedia, there are 6,309 pairs (23%+) intersecting with each other when computing their relations deterministically.

Second, two polygons might be at very different map scale (Figure 5.2b). *dbr:Seal_Beach,_California* (the red polygon) has a extremely smaller map scale compared with *dbr:California* (the blue polygon). When using deterministic spatial operators, sliver polygons will lead to wrong answers while as for the rasterization method, the red polygon become too small to occupy even one pixel of the image.

Last but not least, some spatial relations such as cardinal direction relations are conceptually vague. While this vagueness can be well handled by neural networks through learning from the golden labels, it is hard to design a deterministic method to predict them. As shown in Figure 5.2c, *dbr:Berkeley,_California* (the blue polygon) sits in the north of *dbr:Piedmont,_California* instead of northeast according to DBpedia. However, based on their polygon representations, both "north" and "northeast" seems to be true. In other words, their cardinal direction is vague and subjective.

[3] In GIScience, sliver polygon is a technical term referring to the small unwanted polygons resulting from polygon intersection or difference.

Because of those three problems, we believe designing a general-purpose neural network-based polygon encoder is necessary and can benefit multiple downstream applications.

5.3 Problem Statement

Based on the Open Geospatial Consortium (OGC) standard, we first give the definition of polygons and multipolygons. Let $\mathcal{G} = \{g_i\}$ be a set of polygonal geometries in a 2D Euclidean space $\mathbb{R}^2$ where $\mathcal{G}$ is a union of a polygon set $\mathcal{P} = \{p_i\}$ and a multipolygon set $\mathcal{Q} = \{q_i\}$, a.k.a $\mathcal{G} = \mathcal{P} \cup \mathcal{Q}$ and $\mathcal{P} \cap \mathcal{Q} = \emptyset$. We have $g_i \in \mathcal{P} \vee g_i \in \mathcal{Q}$. Each *polygon* p_i can be represented as a tuple $(\mathbf{B}_i, h_i = \{\mathbf{H}_{ij}\})$ where $\mathbf{B}_i \in \mathbb{R}^{N_{b_i} \times 2}$ indicates a point coordinate matrix for the exterior of p_i defined in a *counterclockwise* direction. $h_i = \{\mathbf{H}_{ij}\}$ is a set of holes for p_i where each hole $\mathbf{H}_{ij} \in \mathbb{R}^{N_{h_{ij}} \times 2}$ is a point coordinate matrix for one interior linear ring of p_i defined in a *clockwise* direction. N_{b_i} indicates the number of unique points in p_i's exterior. The first and last point of $\mathbf{B}_i$ are not the same and $\mathbf{B}_i$ does not intersect with itself. Similar logic applies to each hole $\mathbf{H}_{ij}$ and $N_{h_{ij}}$ is the number of unique points in the jth hole of p_i. It is common that one geographic entity (e.g., Japan) cannot be represented as one single polygon but a multipolygon. Here, a *multipolygon* $q_k \in \mathcal{Q}$ is a set of polygons $q_k = \{p_{ki}\}$ which represents one entity. If a polygonal geometry g_i is a single polygon without any holes, i.e., $g_i = (\mathbf{B}_i, h_i = \emptyset)$, we call it a *simple polygon*. Otherwise, we call it a complex polygonal geometry which might be a multipolygon or a polygon with holes.

Distributed representation of polygonal geometries in the 2D Euclidean space $\mathbb{R}^2$ can be defined as a function $Enc_{\mathcal{G},\theta}(g_i) : \mathcal{G}^* \rightarrow \mathbb{R}^d$ which is parameterized by θ and maps any polygonal geometry $g_i \in \mathcal{G}^*$ in $\mathbb{R}^2$ to a vector representation of d dimension[4]. Here $\mathcal{G}^*$ indicates the set of all possible polygonal geometries in $\mathbb{R}^2$ and $\mathcal{G} \subseteq \mathcal{G}^*$.

[4] We use $Enc(g_i)$ to represent $Enc_{\mathcal{G},\theta}(g_i)$ in the following

Figure 5.3a illustrates a multipolygon $q = \{p_0, p_1\}$ where $p_0 = (\mathbf{B}_0, h_0 = \{\mathbf{H}_{00}\})$ has one hole and $p_1 = (\mathbf{B}_1, h_1 = \emptyset)$ has no hole. An ideal polygon encoder $Enc(g_i)$ should satisfy four properties:

1. **Loop invariance (Loop)**: The encoding result of a polygon p_i should be invariant when starting with different vertices to loop around its exterior/interior. Let consider $p_0'' = (\mathbf{B}_0, \emptyset)$ as a simple polygon made up from the exterior of p_0. $\mathbf{B}_0$ can be written as $\mathbf{B}_0 = [\mathbf{x}_A^T; \mathbf{x}_B^T; \mathbf{x}_C^T; \mathbf{x}_D^T; \mathbf{x}_E^T; \mathbf{x}_F^T] \in \mathbb{R}^{6\times2}$. Let $\mathbf{B}_0^{(s)} = \mathbf{L}_s\mathbf{B}_0$ as another representation of p_0'''s exterior where $\mathbf{L}_s \in \mathbb{R}^{6\times6}$ is a loop matrix which shifts the order of $\mathbf{B}_0$ by s. For example, $\mathbf{B}_0^{(3)} = \mathbf{L}_3\mathbf{B}_0 = [\mathbf{x}_D^T; \mathbf{x}_E^T; \mathbf{x}_F^T; \mathbf{x}_A^T; \mathbf{x}_B^T; \mathbf{x}_C^T]$. Conceptually, we have $p_0'' = (\mathbf{B}_0, \emptyset) = p_0^{(s)} = (\mathbf{L}_s\mathbf{B}_0, \emptyset)\ \forall s \in \{1, 2, ..., 6\}$. Loop invariance expects $Enc(p_0'') = Enc(p_0^{(s)})$.

2. **Trivial vertex invariance (TriV)**: The encoding result of a polygon (or multipolygon) should be invariant when we add/delete trivial vertices to/from its exterior or interiors. Trivial vertices are unimportant vertices such that adding or deleting them from polygons' exteriors or interiors does not change their overall shape and topology. For example, 6 red vertices - A', B', C', D', E', F' - (Figure 5.3b) are trivial vertices of Polygon p_0' since deleting them yield Polygon p_0 which has the same shape as p_0'. We expect $Enc(p_0') = Enc(p_0)$.

3. **Part permutation invariance (ParP)**: The encoding result of a multipolygon q_i should be invariant when permuting the feed-in order of its parts. For instance, the encoding result of $Enc(q)$ (Figure 5.3a) should not change when changing the feed-in order of p_0, p_1.

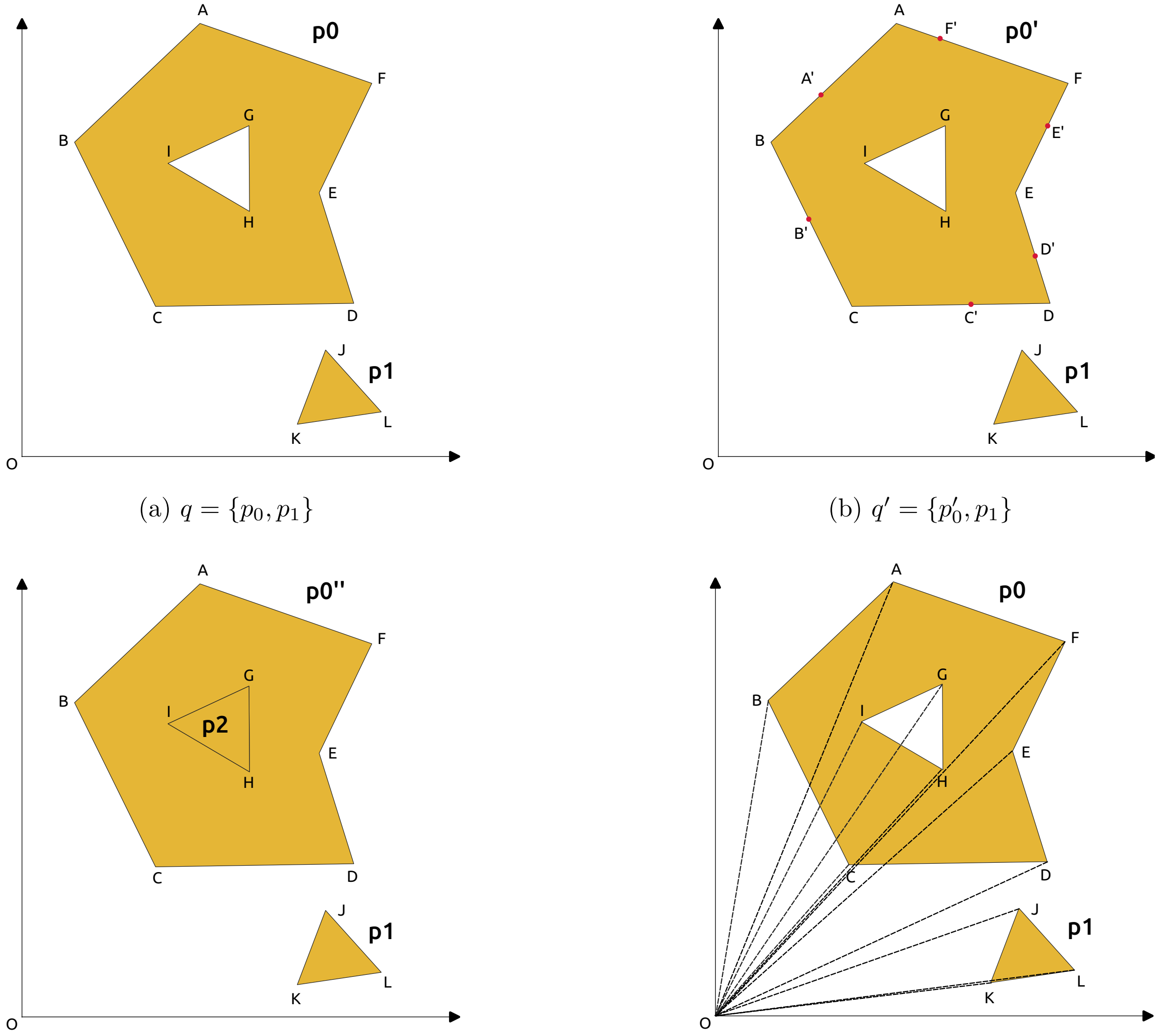
4. **Topology awareness (Topo)**: The polygon encoder $Enc(g_i)$ should be aware of the topology of the polygonal geometry g_i. $Enc(g_i)$ should not only encode

the boundary information of g_i but also be aware of the exterior and interior relationship. For example, as shown in Figure 5.3a and 5.3c, $q = \{p_0, p_1\}$ and $q'' = \{p''_0, p_1, p_2\}$ are two multipolygons and p_2 is inside of p''_0. Although q and q'' have the same boundary information, the encoding results of them should be different given their different topological information.

We call the above properties as **four polygon encoding properties**. As can be seen, these properties are unique requirements for encoding polygonal geometries. In certain scenarios, **translation invariance**, **scale invariance**, and **rotation invariance**, which require the encoding results of a polygon encoder unchanged when polygons are gone through translation/scale/rotation translations, are also expected in many shape related tasks such as shape classification, shape matching, shape retrieval [167]. However, in other tasks such as spatial relation prediction (including topological relations and cardinal direction relations) and GeoQA, translation/scale/rotation invariance are unwanted. For example, after a translation transformation on p_0 (Figure 5.3a), the cardinal direction between p_0 and p_1 changes. So in this work, we primarily focus on the first four properties.

5.4 Related Work

2D Shape Classification Shape classification aims at classifying the silhouette of an object (e.g., animal, leaf) which is usually a polygon/multipolygon into their corresponding class. Multiple shape classification datasets have been constructed for this purpose such as MPEG-7 [168], Animal [162], and Swedish leaf dataset [169]. Wang et al. [163] propose a Bag of Contour Fragment (BCF) method by decomposing one shape polygon into contour fragments each of which is described by a shape descriptor. Then a compact shape representation is max-pooled from them based on a spatial pyramid method. Hofer et al. [170] convert 2D object shapes (images) into topological signatures and input them



(a) $q = \{p_0, p_1\}$

(b) $q' = \{p'_0, p_1\}$

(c) $q'' = \{p''_0, p_1, p_2\}$

(d) 2-simplex mash $\mathcal{S}^{(j)}$

Figure 5.3: An illustration of the structure of multipolygons for explanation of four polygon encoding properties and the illustration of the auxiliary node method. (a) An illustration of a multipolygon $q = \{p_0, p_1\}$ with two parts. $p_0 = (\mathbf{B}_0, h_0 = \{\mathbf{H}_{00}\})$ has one hole and $p_1 = (\mathbf{B}_1, h_1 = \emptyset)$ has no hole. (b) An illustration of a multipolygon $q' = \{p'_0, p_1\}$ where p'_0 has the same shape as p_0 but adding additional 6 trivial vertices (red dots) - A', B', C', D', E', F' - to its exterior. (c) An illustration of a multipolygon $q'' = \{p''_0, p_1, p_2\}$ with three parts where $p''_0 = (\mathbf{B}_0, \emptyset)$ is a simple polygon made up from the exterior of p_0. p_2 is a simple polygon made up from the boundary of p_0's interior $\mathbf{H}_{00}$. (d) An illustration of the auxiliary node method which converts multipolygon q into a 2-simplex mesh $\mathcal{S}^{(j)}$ by adding the origin point $\mathbf{x}_O$ as another vertex.

into a CNN-based model. Most of these shape classification datasets are rather small scale and challenging for deep learning model. For example, according to Kurnianggoro et al. [171], BCF [163], a feature engineering model is still the state-of-the-art model on MPEG-7 and outperforms all deep learning models.

Polygon Encoding Recently, there are a few research focusing on the problem of encoding polygonal geometries directly instead of first rasterizing them into images. Most of them only consider encoding simple polygons. For example, Veer et al. [164] propose a 1D CNN model with zero padding and an RNN model to directly encode the exterior of a simple polygon into the embedding space. None of these two models satisfies four polygon encoding properties. Yan et al. [157] propose a graph convolutional autoencoder model (GCAE) to learn an embedding for each building footprint (a simple polygon) in an unsupervised learning manner. Because a polygon exterior is represented as a graph, not a 1D sequence as Veer et al. [164] did, GCAE is loop invariant, but it cannot satisfy other properties. Instead of directly encoding polygons in the spatial domain, Jiang et al. [172] propose to use a Non-Uniform Fourier Transformation (NUFT) followed by a inverse Fourier transformation (IFT) to transform a polygon, either a simple polygon or a complex polygonal geometry, into a 2D image so that many CNN models can be applied on it for different shape-based tasks. DDSL [167] further extends this NUFT-IFT operation into a differentiable rasterization layer. However, this NUFT-IFT rasterization layer sacrifices a huge information loss as shown in Figure 5.2b. Inspired from DDSL, our NUFTspec model adopt the NUFT idea. Instead of doing a IFT, we directly consume the polygon features in the spectral domain. So NUFTspec has a lower information loss while still satisfies all four polygon encoding properties. We will discuss the NUFT method in detail in Section 5.5. For more detailed related work, please see Appendix 5.8.1.

5.5 Method

In this section, we present two polygon encoders: NUFTspec and ResNet1D. NUFTspec first applies NUFT to transform a polygonal geometry into the spectral domain and learn the polygon embedding from that. In contrast, ResNet1D utilize a 1D CNN to encode the exterior of a simple polygon into the embedding space. Basically, they learns polygon representations in different domains. We wills compare them with other baselines and discuss their properties in Section 5.5.4.

5.5.1 NUFTspec Model

NUFTspec first applies Non-Uniform Fourier Transforms (NUFT) to convert a polygonal geometry g into the spectral domain. Then it directly feeds these spectral features into a multi-layer perceptron to obtain the polygon embedding $\mathbf{p}$ of g. g can be either a polygon with/without holes, or a multipolygon.

Auxiliary Node Following DDSL, we first convert a given watertight polygonal geometry g into a j-simplex mesh $\mathcal{S}^{(j)} = \{\mathbf{S}_n^{(j)}\}_{n=1}^{N_p} = (\mathbf{V}, \mathbf{E}, \mathbf{D})$ (here $j = 2$) by adding one auxiliary node (the origin point $\mathbf{x}_O$). $(\mathbf{V}, \mathbf{E}, \mathbf{D})$ denote three matrices, known as the vertex matrix, edge matrix, and density matrix. As shown in Figure 5.3d, for the nth boundary segment of each sub-polygon's exterior and interiors of g, we connect its two vertices to the origin $\mathbf{x}_O = [0, 0]$ to construct a 2-simplex (triangle) $\mathbf{S}_n^{(j)}$. For instance, for Edge ε_{AB}, we construction a Triangle $\mathbf{S}_{ABO}^{(j)} = \triangle_{ABO}$. For a polygonal geometry g with in total N_p vertices (including vertices of each sub-polygon's exterior and interior), we can get N_p 2-simplexes which form a 2-simplex mesh $\mathcal{S}^{(j)} = \{\mathbf{S}_n^{(j)}\}_{n=1}^{N_p} = (\mathbf{V}, \mathbf{E}, \mathbf{D})$. See Appendix 5.8.2 for the explanation of $\mathbf{V}, \mathbf{E}, \mathbf{D}$.

NUFT Next, we perform NUFT on this j-simplex mesh $\mathcal{S}^{(j)} = \{\mathbf{S}_n^{(j)}\}_{n=1}^{N_p}$ (here $j = 2$). Compared with the conventional Discrete Fourier transform (DFT) whose input signal is sampled at equally spaced points or frequencies (or both), NUFT can deal with input signal sampled at non-equally spaced points or transform the input into non-equally spaced frequencies. This makes NUFT very suitable for irregular structured data such as point cloud, line meshes, polygonal geometries, and so on [172, 167]. In contrast, DFT is more suitable for regular structured data such as images, videos [173].

J-simplex function For the nth j-simplex $\mathbf{S}_n^{(j)}$ in a j-simplex mesh $\mathcal{S}^{(j)} = \{\mathbf{S}_n^{(j)}\}_{n=1}^{N_p} = (\mathbf{V}, \mathbf{E}, \mathbf{D})$, we define a density function

$$f_n^{(j)}(\mathbf{x}) = \begin{cases} \rho_n, & \mathbf{x} \in \mathbf{S}_n^{(j)} \\ 0, & \mathbf{x} \notin \mathbf{S}_n^{(j)} \end{cases}, \ f_{\mathcal{S}}^{(j)}(\mathbf{x}) = \sum_{n=1}^{N_p} f_n^{(j)}(\mathbf{x}) \tag{5.1}$$

in which ρ_n is the signal density defined on $\mathbf{S}_n^{(j)}$. The Piecewise-Constant Function (PCF) over simplex mesh $\mathcal{S}^{(j)}$ is the superposition of $f_n^{(j)}(\mathbf{x})$ for each simplex $\mathbf{S}_n^{(j)}$:

NUFT of PCF $f_{\mathcal{S}}^{(j)}(\mathbf{x})$ The NUFT of PCF $f_{\mathcal{S}}^{(j)}(\mathbf{x})$ over a j-simplex mesh $\mathcal{S}^{(j)} = \{\mathbf{S}_n^{(j)}\}_{n=1}^{N_p} = (\mathbf{V}, \mathbf{E}, \mathbf{D})$ on a set of N_w Fourier base frequencies $\{\mathbf{w}_k\}_{k=1}^{N_w}$ is a sequence of N_w complex numbers:

$$F_{\mathcal{S}}^{(j)}(\mathbf{x}) = [F_{\mathcal{S},1}^{(j)}(\mathbf{x}), F_{\mathcal{S},2}^{(j)}(\mathbf{x}), ..., F_{\mathcal{S},k}^{(j)}(\mathbf{x}), ..., F_{\mathcal{S},N_w}^{(j)}(\mathbf{x})] \tag{5.2}$$

where the NUFT of $f_{\mathcal{S}}^{(j)}(\mathbf{x})$ on each base frequency $\mathbf{w}_k \in \mathbb{R}^2$ can be written as the weighted sum of the Fourier transform on each j-simplex $\mathbf{S}_n^{(j)}$:

$$F_{\mathcal{S},k}^{(j)}(\mathbf{x}) = \int \cdots \int_{-\infty}^{\infty} f_{\mathcal{S}}^{(j)}(\mathbf{x}) e^{-i\langle \mathbf{w}_k, \mathbf{x} \rangle} d\mathbf{x} = \sum_{n=1}^{N_p} \rho_n \int \cdots \int_{\mathbf{S}_n^{(j)}} e^{-i\langle \mathbf{w}_k, \mathbf{x} \rangle} d\mathbf{x} = \sum_{n=1}^{N_p} \rho_n F_{n,k}^{(j)}(\mathbf{x}) \tag{5.3}$$

$F_{n,k}^{(j)}(\mathbf{x})$ is the NUFT on the nth simplex $\mathbf{S}_n^{(j)}$ with base frequency $\mathbf{w}_k$. See Appendix 5.8.3 for its computation. Finally, we can define the *NUFTspec* polygon encoder $Enc_{NUFTspec}(g)$ as:

$$Enc_{NUFTspec}(g) = MLP_F(\Psi\big(F_{\mathcal{S}}^{(j)}(\mathbf{x})\big)) \tag{5.4}$$

Here, $MLP_F(\cdot)$ is a K_F layer multi-layer perceptron in which each layer is a linear layer followed by a nonlinearity (e.g., ReLU), a skip connection, and a layer normalization layer [174]. $\Psi(\cdot)$ first extract $2N_w$ dimension real value vector from $F_{\mathcal{S}}^{(j)}(\mathbf{x})$ and then do a normalization on this spectral representation such as L2 normalization, or batch normalization.

5.5.2 ResNet1D Model

Except for NUFTspec, we also propose anther 1D ResNet [175] based polygon encoder called ResNet1D. Unlike NUFTspec, ResNet1D cannot deal with polygons with holes or multipolygons but only focuses on simple polygons.

Given a simple polygon $g = (\mathbf{B}, \emptyset)$ where $\mathbf{B} = [\mathbf{x}_0^T; \mathbf{x}_1^T; ...; \mathbf{x}_m^T; ...; \mathbf{x}_{N_p-1}^T] \in \mathbb{R}^{N_p \times 2}$, ResNet1D treats the exterior $\mathbf{B}$ of g as a 1D coordinate sequence. Before feeding $\mathbf{B}$ into the 1D ResNet layer, we first compute a point embedding $\mathbf{l}_m \in \mathbb{R}^{4t+2}$ for the mth point $\mathbf{x}_m$ by concatenating $\mathbf{x}_m$ with its spatial affinity with its neighboring $2t$ points:

$$\mathbf{l}_m = [\mathbf{x}_m; \mathbf{x}_{m-t} - \mathbf{x}_m; ...; \mathbf{x}_{m-1} - \mathbf{x}_m; \mathbf{x}_{m+1} - \mathbf{x}_m; ...; \mathbf{x}_{m+t} - \mathbf{x}_m] \tag{5.5}$$

We call Equation 5.5 **KDelta point encoder**, which adds neighborhood structure information into each point embedding and helps to reduce the need to train very deep encoders. Here, if $m - t < 0$ or $m + t \geq N_p$, we get its coordinates by *circular padding* given the fact that $\mathbf{B}$ represents a circle. The resulting embedding matrix $\mathbf{L} = [\mathbf{l}_0^T; \mathbf{l}_1^T; ...; \mathbf{l}_m^T; ...; \mathbf{l}_{N_p-1}^T] \in \mathbb{R}^{N_p \times (4t+2)}$ is the input of a modified 1D ResNet model which uses *circular padding* instead of zero padding in 1D CNN and max pooling layers to ensure loop invariance. Please refer to Appendix 5.8.4 for the architecture of ResNet1D.

5.5.3 Spatial Relation Prediction Model

Given a polygon pair (g_{sub}, g_{obj}), the spatial relation prediction task aims at predicting a spatial relation between them such as topological relations, cardinal direction relations, and so on. In this work, we adopt a multi-layer perceptron (MLP) based spatial relation prediction model. In Equation 5.6, $[Enc(g_{sub}); Enc(g_{obj})] \in \mathbb{R}^{2d}$ indicates the concatenation of the polygon embeddings of g_{sub} and g_{obj}. $MLP_{rel}(\cdot)$ takes this as input and outputs raw logits over all N_r possible spatial relations. $softmax(\cdot)$ normalizes it into a probability distribution $\log(r|g_{sub}, g_{obj})$ over N_r relations.

$$\log(r|g_{sub}, g_{obj}) = softmax(MLP_{rel}([Enc(g_{sub}); Enc(g_{obj})])) \tag{5.6}$$

5.5.4 Model Comparison

Theorem 2 *Polygon Encoder NUFTspec is (1) loop invariant, (2) trivial vertex invariant, (3) part permutation invariant, and (4) topology aware.*

Theorem 3 *Polygon Encoder ResNet1D is loop invariant.*

Now we compare different polygon encoding models we discussed in Section 5.3 as well as our models - ResNet1D and NUFTspec based on their encoding capabilities such as whether or not it can handle holes and multipolygons as well as four polygon encoding properties (See Section 5.3). In terms of NUFTspec and ResNet1D, we declare Theorem 2 and 3 whose proofs can be seen in Appendix 5.8.5 and 5.8.6. Table 5.1 shows the full comparison result. Only DDSL+LeNet5 [172, 167] and our NUFTspec can handle polygons with holes and multipolygons. They also satisfy all four properties. This is because the beauty of NUFT. VeerCNN [164] does not satisfies any of these properties while GCAE [157] and ResNet1D can only satisfy the loop invariance property. GCAE achieves this by converting the exterior of a simple polygon into a graph which is order invariant. ResNet1D achieves this by using circular padding in each 1D CNN and max pooling layer.

Table 5.1: The comparison among different polygon encoders by properties such as whether it can handle polygon with holes and multipolygons as well as the four polygon encoding properties we discuss in Section 5.3.

Property	Holes	Multipolygons	Loop	TriV	ParP	Topo
Deterministic	-	-	-	-	-	-
VeerCNN [164]	No	No	No	No	No	No
GCAE [157]	No	No	Yes	No	No	No
DDSL+LeNet5 [172, 167]	Yes	Yes	Yes	Yes	Yes	Yes
ResNet1D	No	No	Yes	No	No	No
NUFTspec	Yes	Yes	Yes	Yes	Yes	Yes

5.6 Experiment

5.6.1 Dataset Construction and Deterministic Baselines

To demonstrate the effectiveness of the proposed NUFTspec and ResNet1D polygon encoder compared with multiple baselines, we conduct a spatial relation prediction task. Since there is not existing benchmark dataset available for this task, we construct two real-world dataset - DBSpaRel46K and DBSpaRelComplex46K- based on DBpedia Knowledge Graph as well as OpenStreetMap. DBSpaRel46K and DBSpaRelComplex46K use the same entity set $\mathcal{E}$ and triple set $\mathcal{T}$ and the only different is DBSpaRel46K uses simple polygons as entities' spatial footprints while DBSpaRelComplex46K allows complex polygonal geometries (See Appendix 5.8.7 for details).

To investigate the difficulty of polygon-based spatial relation prediction task, we compute the topological relations between the subject and object entity of each triple in

DBSpaRel46K based on a deterministic RCC8-based spatial operator[5]. The statistics is shown in Table 5.2. We can see almost all triples with cardinal direction relations (the last 8 relations) have subject and object entities that intersect or disjoint with each other. Interestingly, for *dbo:isPartOf* relation, only in 76.5% of the 27,364 triples the subject polygons are inside the corresponding object polygons. Based on manual inspection of these error cases, most of those 'intersects' cases are caused by the *sliver polygon* problem shown in Figure 5.2a. This clearly indicates the necessity of polygon encoding models.

Table 5.2: The deterministic topological relation computation result of DBSpaRel46K dataset.

Relation	Contains	Intersects	Touches	Disjoint
dbo:isPartOf	20923	6309	0	132
dbp:north	4	2451	0	352
dbp:east	2	2447	0	348
dbp:south	5	2405	0	360
dbp:west	3	2408	0	348
dbp:northwest	4	1644	0	415
dbp:southeast	2	1618	0	404
dbp:southwest	4	1622	0	374
dbp:northeast	3	1634	0	357

5.6.2 Spatial Relation Prediction

To show the effectiveness of the proposed ResNet1D and NUFTspec model, we evaluate them on DBSpaRel46K and DBSpaRelComplex46K dataset and compare them with three baselines - Deterministic , VeerCNN [164], and DDSL+LeNet5 [167]. Please refer to Appendix 5.8.9 for the implementation detail of each baselines.

[5] https://shapely.readthedocs.io/en/stable/manual.html#binary-predicates

Table 5.3 shows the overall performance of different models on the training, validation, and test dataset of DBSpaRel46K as well as DBSpaRelComplex46K dataset. Note that, since only DDSL+LeNet5 and NUFTspec can handle polygons with holes and multipolygons, they are the only two models can be used for DBSpaRelComplex46K dataset. From Table 5.3, we can see that NUFTspec can outperform all the other baselines as well as ResNet1D on the training, testing, and validation dataset of DBSpaRel46K. This is because NUFTspec can efficiently utilize the spectral features obtained by NUFT and transform them into a meaningful polygon embedding. Despite the fact that ResNet1D is similar to VeerCNNin the sense that both of them use 1D CNN layers to model simple polygons' exterior coordinate sequence, ResNet1D shows the second best result on DBSpaRel46K dataset. To understand the reason why ResNet1D shows promising results, we do an ablation study and the results are shown in Table 5.4. It shows that when we replace the circular padding with zero padding - ResNet1D(zero padding), the performance of ResNet1D drop a lot. A similar situation happens when we delete the KDelta point encoder component and direct feed the raw point coordinate to ResNet layers - ResNet1D(raw pt). This demonstrates that KDelta uses the spatial affinity features (Equation 5.5) to enrich the point embedding with its neighborhood information which is very helpful for polygon encoding and spatial relation prediction. The circular padding is also critical since it preserve the loop invariance property.

Table 5.3 also shows that NUFTspec can outperform DDSL+LeNet5 on DBSpaRelComplex46Kdataset. We also test the robustness of DDSL+LeNet5 and NUFTspecon DBSpaRelComplex46K when we vary the number of frequencies we use in NUFT. In Equation 5.2, the set of N_w Fourier base frequencies $\{\mathbf{w}_k\}_{k=1}^{N_w}$ is sampled along the X and Y direction. We denote N_{wx} and N_{wy} as the number of frequencies sampled along the X and Y direction, where usually we have $N_{wx} = N_{wy}$. Figure 5.4 compares the model performance of them with different N_{wx} in the training/validation/testing dataset

Table 5.3: Overall evaluation result of different polygon encoding models

	DBSpaRel46K			DBSpaRelComplex46K		
	Train	Valid	Test	Train	Valid	Test
Deterministic	75.42	75.18	73.80	75.17	75.30	73.90
VeerCNN [164]	92.10	77.90	77.59	-	-	-
DDSL+LeNet5 [172, 167]	89.85	74.56	74.00	88.94	72.74	72.98
ResNet1D	91.98	78.13	77.79	-	-	-
NUFTspec	**92.81**	**79.57**	**78.48**	**93.77**	**79.33**	**79.12**

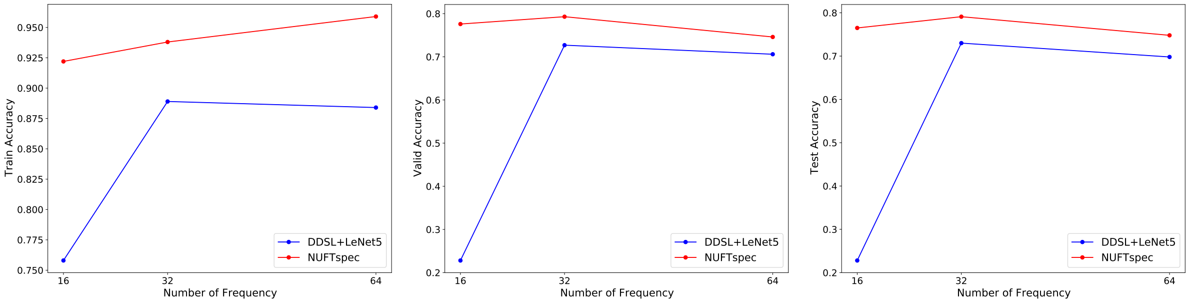
Table 5.4: Ablation study of ResNet1D on DBSpaRel46K dataset.

	dbtopo		
	Train	Valid	Test
ResNet1D	0.920	0.781	0.778
ResNet1D(zero padding)	0.907	0.745	0.731
ResNet1D(raw pt)	0.913	0.759	0.753

of DBSpaRelComplex46K. We can see that compared with DDSL+LeNet5, our NUFTspec is much more robust when we vary N_{wx}. Especially when we have lower N_{wx} (e.g., $N_{wx} = 16$), the performance of DDSL+LeNet5 drops very fast while NUFTspec only shows a slightly drop.

5.7 Conclusion

In this work, we formally discuss the problem of polygon encoding. We show that because of sliver polygon problem, scale problem as well as the vagueness of spatial relation, the determinstic spatial operator is not suitable for spatial relation relation prediction task, and more broadly geographic question answering task. And we are in despairing



(a) Training (b) Validation (c) Testing

Figure 5.4: A performance comparison between DDSL+LeNet5 and NUFTspec on the training/validation/testing datset of DBSpaRelComplex46K with different numbers of frequencies used in X - N_{wx} (or Y - N_{wy}) in NUFT.

need for a good polygon encoder which can satisfy four properties - loop invariance, trivial vertex invariance, part permutation invariance, and topology awareness. We propose two polygon encoders - ResNet1D and NUFTspec. While ResNet1D utilizes circular padding to achieve loop invariance on simple polygons, NUFTspec utilizes Non-uniform Fourier transformation (NUFT) which is capable of handling polygons with holes and multipolygons and it also satisfies all four polygon encoding properties. We systematically compare multiple existing polygon encoders on two real-world spatial relation prediction datasets - DBSpaRel46K and DBSpaRelComplex46K which are constructed based on DBpedia and OpenStreetMap. Evaluation results show that NUFTspec can outperform all baselines on both datasets and is very robust when we vary the number of sampled frequencies in NUFT. In contrast, the performance of DDSL+LeNet5 drops a lot with lower frequency number and thus is less robust. This shows the advantage of directly using spectral features to do polygon encoding instead of transforming them back to the spatial domain.

Despite these successful stories, several issues still remain to be solved. First, the training dataset of DBSpaRel46K and DBSpaRelComplex46K is very unbalance and there are much more *dbo:isPartOf* triples than those with other relations and this imbalance cannot be avoid. This imbalance nature causes an overfitting issue for all current models. How to design spatial relation prediction model which is more robust for dataset imbalance is a very interesting research direction. Second, how to effectively utilize features in the spectral domain is also an interesting future research direction.

5.8 Appendix

5.8.1 Additional Related Work

Object Instance Segmentation and Polygon Decoding Most existing machine learning/deep learning research about polygons mainly focus on object instance segmentation and object localization task. They aim at constructing a simple polygon based on an image as the object mask to be localized. For example, in Zhang at el. [176] and Sun et al. [159], the authors first detect the boundary fragments of polygons from images and then they extract the polygon by finding the optimal circle linking these fragments into the object contours. As for more recent deep learning approaches, Polygon-RNN [160] first encodes a given image with a VGG-like CNN structure and then decodes the polygon mask of an object with a two-layer convolutional LSTM with skip connections. The RNN polygon decoder decodes one polygon vertex at one time step until decoding the end-of-sequence token which indicates the polygon is closing. The first vertex is predicted with another CNN using a multi-task loss. Polygon-RNN++ [161] improves Polygon-RNN by adding a Gated Graph Neural Network (GGNN) [177] after the RNN polygon decoder to significantly increase the spatial resolution of the output polygon. Since the cross-entropy loss used by Polygon-RNN over-penalizes the model and is different from the evaluation metric, Polygon-RNN++ changes the learning algorithm to reinforcement learning to directly optimize on the evaluation metric. Compared with our polygon encoding model, these polygon decoding work have different focus.

Polygon Encoding Veer et al. [164] propose a CNN model and an RNN model to directly encode a *simple polygon* into an embedding space for several polygon-shape-based tasks such as neighbourhood population prediction, building footprint classification, and archaeological ground feature classification. They only consider simple polygons and treat each polygon as a sequence of point coordinates in their exterior. As for the RNN model, they directly feed the polygon exterior coordinate sequence into a bi-directional LSTM and take the last state as the polygon embedding. For the CNN model, they feed the polygon exterior sequence into a series of 1D convolutional layers (zero padding) followed by a global average pooling: Conv1D - MaxPooling1D - Conv1D - GlobalAveragePooling1D. Experiments on three tasks show that the CNN model is better than the RNN model on all three tasks. Since these two models can not handle complex polygonal geometries, they can not satisfy the part permutation invariance and topology awareness. Moreover, since the feed-in order and the length of the polygon exterior sequence will affect the results of both models, they are not loop invariant nor trivial vertex invariant. In this work, we denote the CNN model as VeerCNN and use it as one of our baselines.

Yan et al. [157] propose a graph convolutional autoencoder model (GCAE) to learn a shape coding for each building footprint which can be represented as a simple polygon in an unsupervised learning manner. The exterior of each building (simple polygon) can be represented as an undirected graph in which boundary vertices/points are nodes that are connected by boundary segments (edges). Each edge is weighted by its length and the description local and regional features for each vertex are extracted based on its neighborhood structure which serve as its initial input features. The GCAE follows a U-Net [178] like architecture which uses graph convolution layers and graph pooling [144] in the graph encoder and upscaling layers in the decoder. The intermediate representation of a building graph is utilized as its shape coding. The effectiveness of GCAE is demonstrated qualitatively and quantitatively on shape similarity and shape retrieval task. Since the "building graph" does not need to represent as a 1D sequence, GCAE is loop invariant. However, GCAE is sensitive to trivial vertices and cannot handle complex polygonal geometries. So GCAE does not satisfy the last three properties. Moreover, Yan et al. [157] also show that GCAE is sensitive to rotation operation.

Instead of encoding polygonal geometries (more generally, 2-/3-simplex meshes) and 3D shapes directly in the spatial domain, Jiang et al. [172] propose to first do Non-Uniform Fourier Transforms (NUFT) to transform them into the spectral domain and then do a inverse Fourier transformation (IFT) to convert them into 2D images or 3D voxels. The result is an image of the polygonal geometry (or a 3D voxel for a 3D shape) which can be easily consumed by different CNN models such as LeNet5 [179], ResNet [175], and Deep Layer Aggregation (DLA) [180]. DDSL [167] further extends this NUFT-IFT operation into a differentiable layer which is more flexible for back propagation. The effectiveness of DDSL has been shown in shape classification task (MNIST), 3D shape retrival task, and 3D surface reconstruction task. Given the NUFT nature, DDSL naturally satisfies all those four polygon encoding properties and can handle complex polygonal geometries. However, the NUFT-IFT operation is essentially a polygon rasterization approach and sacrifice a huge information loss as shown in Figure 5.2b.

5.8.2 An illustration of Auxiliary Node Method

A 2-simplex mesh $\mathcal{S}^{(j)} = \{\mathbf{S}_n^{(j)}\}_{n=1}^{N_p} = (\mathbf{V}, \mathbf{E}, \mathbf{D})$ is represented as three matrices - float matrix $\mathbf{V} \in \mathbb{R}^{(N_p+1)\times 2}$ for polygon/simplex vertex coordinates, non negative integer matrix $\mathbf{E} \in \mathbb{N}_0^{N_p \times 3}$ for 2-simplex connectivity, and float matrix $\mathbf{D} \in \mathbb{R}^{N_p \times d_d}$ for per-simplex density. $\mathbf{V}$ contains the coordinates of vertices of each sub-polygon's exterior and interiors of g as well as the origin $\mathbf{x}_O = [0, 0]$ (the last row). The nth row of $\mathbf{E}$ corresponds to the nth simplex $\mathbf{S}_n^{(j)}$ in $\mathcal{S}^{(j)}$ whose values indicate the indices of vertices of $\mathbf{S}_n^{(j)}$ in $\mathbf{V}$. d_d is the dimension of per-simplex density features which is the number of features associated with each boundary segment/simplex (e.g., $\mathbf{S}_n^{(j)}$). Here, we can assume that g has a constant density of 1, i.e., $\mathbf{D} = \mathbf{1}$ where $\mathbf{1}$ is a $N_p \times 1$ constant one matrix. Note that we should make sure the boundary of g is oriented correctly - the exteriors of all sub-polygons should be oriented in a counter-clockwise fashion while all interiors should be oriented in a clockwise fashion as we defined in Section 5.3. This makes sure that we can use the *right-hand rule*[6] to infer the correct orientation of each boundary segment and we can compute the signed content (area) of each simplex $\mathbf{S}_n^{(j)}$ so that the topology of g is preserved - *topology awareness*.

To concretely show how to convert a polygonal geometry g into a 2-simple mesh $\mathbf{S} = (\mathbf{V}, \mathbf{E}, \mathbf{D})$, we use the example of Multipolygon g shown in Figure 5.3d. It can be converted into Simplex $\mathbf{S} = (\mathbf{V}, \mathbf{E}, \mathbf{D})$ where $\mathbf{V} = [\mathbf{x}_A^T; \mathbf{x}_B^T; \mathbf{x}_C^T; \mathbf{x}_D^T; \mathbf{x}_E^T; \mathbf{x}_F^T; \mathbf{x}_G^T; \mathbf{x}_H^T; \mathbf{x}_I^T; \mathbf{x}_J^T; \mathbf{x}_K^T; \mathbf{x}_L^T; \mathbf{x}_O^T] \in \mathbb{R}^{13\times 2}$, $\mathbf{E} = [[0, 1, 12], [1, 2, 12], [2, 3, 12], [3, 4, 12], [4, 5, 12], [5, 0, 12], [6, 7, 12], [7, 8, 12], [8, 6, 12], [9, 10, 12], [10, 11, 12], [11, 9, 12]] \in \mathbb{N}_0^{12\times 3}$, and $\mathbf{D}$ is a 12×1 constant one matrix.

5.8.3 Compute $F_{n,k}^{(j)}$

In Equation 5.3, we can define the NUFT on the nth simplex $\mathbf{S}_n^{(j)}$ as

$$F_{n,k}^{(j)}(\mathbf{x}) = i^j \mu(\mathbf{S}_n^{(j)}) \gamma_n^{(j)} \sum_{t=1}^{j+1} \frac{e^{-i\langle \mathbf{w}_k, \mathbf{x}_t \rangle}}{\prod_{l=1}^{j+1} (e^{-i\langle \mathbf{w}_k, \mathbf{x}_t \rangle} - e^{-i\langle \mathbf{w}_k, \mathbf{x}_l \rangle})} \tag{5.7}$$

$\mu(\cdot) : \mathcal{S}^{(j)} \to \{-1, 1\}$ is a sign function which determines the sign of the content (area) of $\mathbf{S}_n^{(j)}$. $\gamma_n^{(j)}$ is the content distortion factor, which is the ratio of the unsigned content of

[6] https://mapster.me/right-hand-rule-geojson-fixer/

$\mathbf{S}_n^{(j)}$ - $C_n^{(j)}$ - and the content of the unit orthogonal j-simplex - $C_I^{(j)} = 1/j!$. $\mu(\mathbf{S}_n^{(j)})\gamma_n^{(j)}$ is the signed content distortion factor of $\mathbf{S}_n^{(j)}$ and the signed content[7] $\mu(\mathbf{S}_n^{(j)})C_n^{(j)}$ can be computed based on the determinant of the Jacobian matrix J_n of simplex $\mathbf{S}_n^{(j)}$. Let $\mathbf{x}_{n,1}, \mathbf{x}_{n,2}, \mathbf{x}_O$ are the three vertices of $\mathbf{S}_n^{(j)}$, we have:

$$\mu(\mathbf{S}_n^{(j)})\gamma_n^{(j)} = \frac{\mu(\mathbf{S}_n^{(j)})C_n^{(j)}}{C_I^{(j)}} = \frac{1/j!\,\det(J_n)}{1/j!} = \det([\mathbf{x}_{n,1} - \mathbf{x}_O, \mathbf{x}_{n,2} - \mathbf{x}_O]) = \det([\mathbf{x}_{n,1}, \mathbf{x}_{n,2}]) \tag{5.8}$$

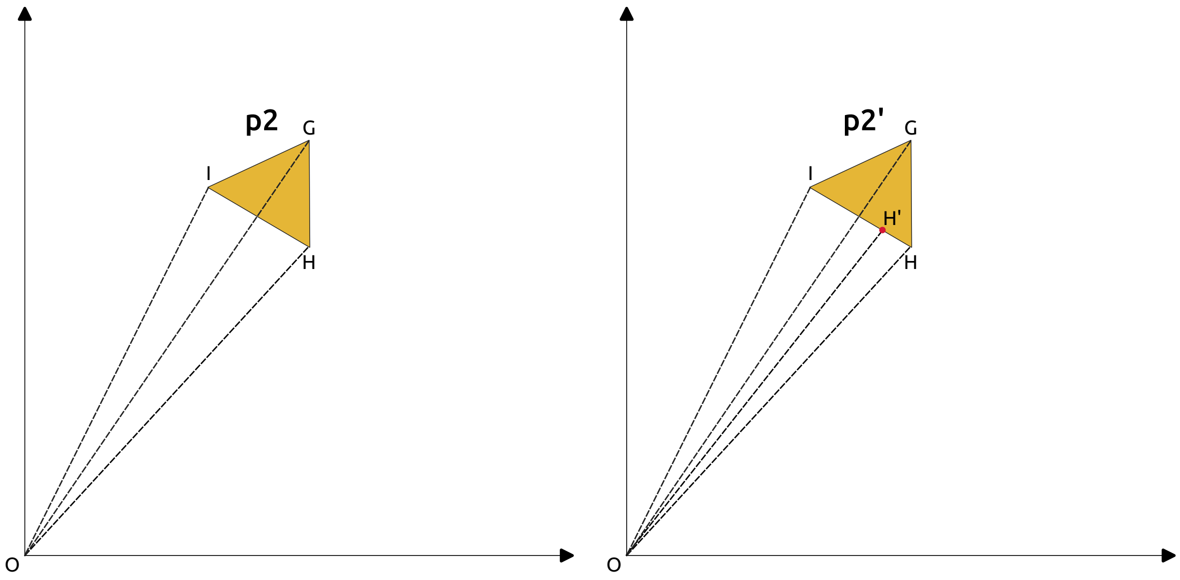
The proof of Equation 5.3, 5.7, and 5.8 can be found in [172].

5.8.4 ResNet1D Model Architecture

Given the resulting embedding matrix $\mathbf{L} = [\mathbf{l}_0^T; \mathbf{l}_1^T; ...; \mathbf{l}_1^T; ...; \mathbf{l}_{N_p-1}^T] \in \mathbb{R}^{N_p \times (4t+2)}$ from the KDelta point encoder (See Equation 5.5) as the input, the whole ResNet1D architecture is illustrated as Equation 5.9.

$$\Big[\mathbf{L} \rightarrow \text{CNN1D}_{3\times3}^{d,1,1} \rightarrow \text{BN1D} \rightarrow \text{ReLU} \rightarrow \text{MP1D}_{2\times2}^{2,0} \rightarrow \big(\text{ResN1D}\big)_{u=1}^{U} \rightarrow \text{GMP1D} \rightarrow \text{DP} \rightarrow \mathbf{p}\Big] \tag{5.9}$$

$\text{CNN1D}_{3\times3}^{d,1,1}$ indicates a 1D CNN layer with 1 stride, 1 padding (circular padding) and d 3×3 kernel. BN1D and ReLU indicate 1D batch normalization layer and a ReLU activation layer. $\text{MP1D}_{2\times2}^{2,0}$ indicates a 1D Max Pooling layer with 2 stride, 0 padding, and kernel size 2×2. $\big(\text{ResN1D}\big)_{u=1}^{U}$ indicates U standard 1D ResNet layers. GMP1D and DP are a global max pooling layer and dropout layer. The final output $Enc_{ResNet1D}(g) = \mathbf{p} \in \mathbb{R}^d$ is the polygon encoding of the simple polygon g.



(a) p_2 and $\mathcal{S}_2^{(j)} = \{\mathbf{S}_{GIO}^{(j)}, \mathbf{S}_{IHO}^{(j)}, \mathbf{S}_{HGO}^{(j)}\}$ (b) p_2' and $\mathcal{S}_{2'}^{(j)} = \{\mathbf{S}_{GIO}^{(j)}, \mathbf{S}_{IH'O}^{(j)}, \mathbf{S}_{H'HO}^{(j)}, \mathbf{S}_{HGO}^{(j)}\}$

Figure 5.5: A illustration to facilitate the proof of Theorem 2 . (a) Polygon p_2 is a simple polygon from Figure 5.3c. We convert it to a 2-simplex mesh $\mathcal{S}_2^{(j)} = \{\mathbf{S}_{GIO}^{(j)}, \mathbf{S}_{IHO}^{(j)}, \mathbf{S}_{HGO}^{(j)}\}$. (b) Polygon p_2' is also a simple polygon which has the same shape as p_2 but add an additional trivial vertex H'. We convert it to a 2-simplex mesh $\mathcal{S}_{2'}^{(j)} = \{\mathbf{S}_{GIO}^{(j)}, \mathbf{S}_{IH'O}^{(j)}, \mathbf{S}_{H'HO}^{(j)}, \mathbf{S}_{HGO}^{(j)}\}$.

5.8.5 Proofs of Theorem 2

Proof of Theorem 2 (1). Given a simple polygon $p = (\mathbf{B}, \emptyset)$, we convert it to a 2-simplex mesh $\mathcal{S}^{(j)} = \{\mathbf{S}_n^{(j)}\}_{n=1}^{N_p}$ similar to what is shown in Figure 5.5a. Since $\{\mathbf{S}_n^{(j)}\}_{n=1}^{N_p}$ is an *unordered set* of 2-simplexes, for any loop matrix $\mathbf{L}_s$, Polygon $p^{(s)} = (\mathbf{L}_s\mathbf{B}, \emptyset)$ will have exactly the same 2-simplex mesh as p - $\mathcal{S}^{(j)}$. 2-simplex mesh $\mathcal{S}^{(j)}$ is the input of our NUFTspec. So the output polygon embedding $Enc_{NUFTspec}(p)$ is invariant to any loop transformation $\mathbf{L}_s$ on Polygon p's exterior $\mathbf{B}$.

Proof of Theorem 2 (2). We use Polygon p_2 and p_2' shown in Figure 5.5a and 5.5b as an example to show the proof. The only difference between them is that p_2' has an additional trivial vertex H' while p_2 and p_2' have the same *shape*. They have different

[7] https://en.m.wikipedia.org/wiki/Simplex#Volume

2-simplex meshes: $\mathcal{S}_{2}^{(j)} = \{\mathbf{S}_{GIO}^{(j)}, \mathbf{S}_{IHO}^{(j)}, \mathbf{S}_{HGO}^{(j)}\}$ and $\mathcal{S}_{2'}^{(j)} = \{\mathbf{S}_{GIO}^{(j)}, \mathbf{S}_{IH'O}^{(j)}, \mathbf{S}_{H'HO}^{(j)}, \mathbf{S}_{HGO}^{(j)}\}$. The Piecewise-Constant Function (PCF) $f_{\mathcal{S}}^{(j)}(\mathbf{x}) = \sum_{n=1}^{N_p} f_{n}^{(j)}(\mathbf{x})$ defined on the simplex mesh $\mathcal{S}_{2}^{(j)}$ is essentially a *summation* over the individual density function $f_{n}^{(j)}(\mathbf{x})$ defined on each simplex of $\mathcal{S}_{2}^{(j)}$. It is the same for $\mathcal{S}_{2'}^{(j)}$. Since p_2 and p_2' have the same *shape*, the PCF defined on them should be exactly the same. Since NUFTspec polygon embedding is derived from the NUFT of the PCF over a 2-simple mesh. So we can conclude that the NUFTspec polygon embeddings of p_2 and p_2' should be the same. In other words, NUFTspec is trivial vertex invariant.

Proof of Theorem 2 (3). Given a multipolygon $q = \{p_i\}$, its 2-simplex mesh $\mathcal{S}^{(j)} = \{\mathbf{S}_{n}^{(j)}\}_{n=1}^{N_p}$ (similar to Figure 5.3d) is an unordered set of signed 2-simplexes/triangles. Changing the feed-in order of polygon set $\{p_i\}$ will not affect the resulting 2-simplex mesh. So NUFTspec is part permutation invariant.

Proof of Theorem 2 (4). Given a polygon $p = (\mathbf{B}, h = \{\mathbf{H}_j\})$ with holes, its 2-simplex mesh $\mathcal{S}^{(j)} = \{\mathbf{S}_{n}^{(j)}\}_{n=1}^{N_p}$ is consist of oriented 2-simplexes. Since we require each polygonal geometry is oriented correctly, the right-hand rule can be used to compute the signed content of each 2-simplex. So the topology of polygon $p = (\mathbf{B}, h = \{\mathbf{H}_j\})$ is preserved during the polygon-simple mesh conversion. Thus, NUFTspec is aware of the topology of the input polygon geometry.

5.8.6 Proofs of Theorem 3

In ResNet1D , circular padding is used in convolution layer with stride 1. Given a polygon $p = (\mathbf{B}, h = \emptyset)$, circular padding wraps the vector $\mathbf{B}$ on one end around to the other end to provide the missing values in the convolution computations near the boundary. Thus, $\mathrm{CNN1D}_{3\times3}^{d,1,1}(\mathbf{L}_s\mathbf{B}) = \mathbf{L}_s\mathrm{CNN1D}_{3\times3}^{d,1,1}(\mathbf{B})$ for any input $\mathbf{B}$. Max pooling layer with stride 1 and circular padding has the similar property. $\mathrm{MP1D}_{2\times2}^{1,1}(\mathbf{L}_s\mathbf{B}) =$

$\mathbf{L}_s(\text{MP1D}^{1,1}_{2\times 2}\mathbf{B})$. Trivially, $\text{BN1D}(\mathbf{L}_s\mathbf{B}) = \mathbf{L}_s\text{BN1D}(\mathbf{B})$ and $\text{ReLU}(\mathbf{L}_s\mathbf{B}) = \mathbf{L}_s\text{ReLU}(\mathbf{B})$. In the end, there is a global maxpooling $\text{GMP1D}(\mathbf{L}_s\mathbf{B}) = \text{GMP1D}(\mathbf{B})$. With these layers as components, ResNet1D would keep the loop invariance.

5.8.7 Dataset Construction

We construct a real-world dataset - DBSpaRel46K- based on DBpedia Knowledge Graph as well as OpenStreetMap with the following steps.

1. We first select a meaningful set of properties $R = \{r_i\}_{i=1}^{N_r}$ from DBpedia which represents different spatial relations.

2. Then we collect all triples $\{(e_{sub}, r_i, e_{obj})\}$ from DBpedia whose relation $r_i \in R$ and e_{sub}, e_{obj} are geographic entities.

3. Next, we filter out triples whose subject e_{sub} or object e_{obj} is located outside the mainland of United States. The resulting triple set $\mathcal{T} = \{(e_{sub}, r_i, e_{obj})\}$ forms a sub-graph of DBpedia with the entity set $\mathcal{E}$.

4. For each entity $e \in \mathcal{E}$, we obtain their corresponding Wikidata ID by using *owl:sameas* links.

5. With each entity e's Wikidata ID, we can obtain their polygonal geometries from OpenStreetMap by using Overpass API[8].

6. The raw polygonal geometries from OpenStreetMap are very detail and complex. For example, Keweenaw County, Michigan is represented as a multipolygon that is consist of 462 sub-polygons. Lake Superior has in total 3130 holes and 206661 vertices in its polygon representation. United States has 128873 vertices. Figure

[8] `https://wiki.openstreetmap.org/wiki/Overpass_API`

5.7a-5.7c in Appendix 5.8.8 shows some statistic about the complexity of these raw geometries. Although DDSL [167] and NUFTspec can handle multipolygons and polygon with holes. ResNet1D and VeerCNN (also GCAE [157]) can only deal with simple polygons. In order to make a fair comparison, we simplify those polygonal geometries collected from OpenStreetmap and make two datasets: DBSpaRel46K and DBSpaRelComplex46K.

7. As for DBSpaRel46K , we delete all holes and only keep one single simple polygon with the largest area as the geometry for each geographic entity. So each geographic entity is associated with one simple polygon. We also simplify the exterior of each simple polygon such that they all have 300 unique vertices. For those polygons who has less than 300 vertices, we do a equal distance interpolation on the exteriors to upsample the vertices to 300. The reason to do so is that same number of vertices make it easier for mini-batch training.

8. As for DBSpaRelComplex46K , we also simplify the polygonal geometries but we still keep holes and multipolygons if necessary. We delete holes if their area are less than 2.5% of the total area of their corresponding polygonal geometries. Figure 5.7d-5.7f show the statistics (number of holes, multipolygons, vertices) of the simplified geometries. Note that in order to do mini-batch training, we simplify the polygon exteriors and interiors to make sure its total number of vertices N_p is 300. DBSpaRel46K and DBSpaRelComplex46K use the same entity set $\mathcal{E}$ and triple set $\mathcal{T}$ and the only different between them is the polygonal geometry of each entity.

9. Figure 5.6 shows the polygon geometries of the total 23264 unique geographic entities used in DBSpaRel46K and DBSpaRelComplex46K. Table 5.5 shows the number of entities with different place types.

10. We split $\mathcal{T}$ into training/validation/test dataset by roughly 80:5:15. By following the traditional transductive knowledge graph embedding literature [85, 66], we make sure all entities appear in the training dataset. We also make sure we have a balance dataset for each spatial relation in the validation and testing dataset. Table 5.6 shows the statistic of the dataset split. Note that the balance of triples with different spatial relations can not be achieved at the same time with the need to include all entities in the graph. So we can see that in training dataset we have far more *dbo:isPartOf* triples than other spatial relations. It depends on the nature of DBpedia and we need to keep this imbalance.

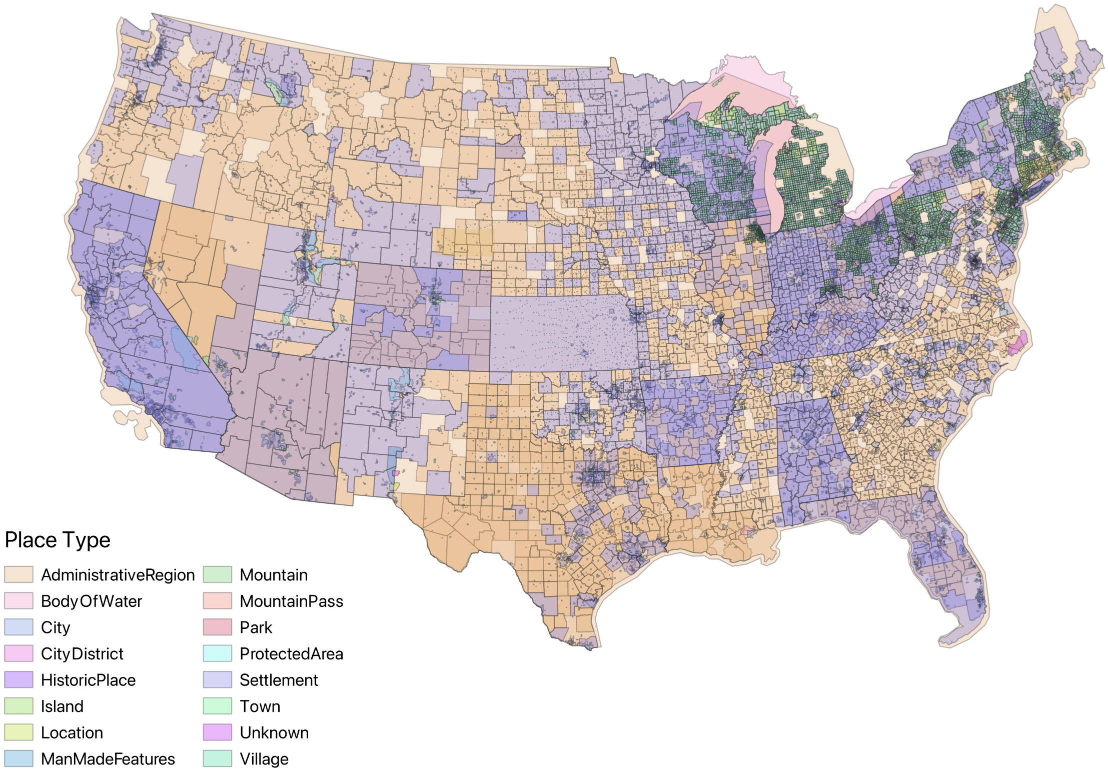


Figure 5.6: A map of geographic entities in DBSpaRel46K dataset.

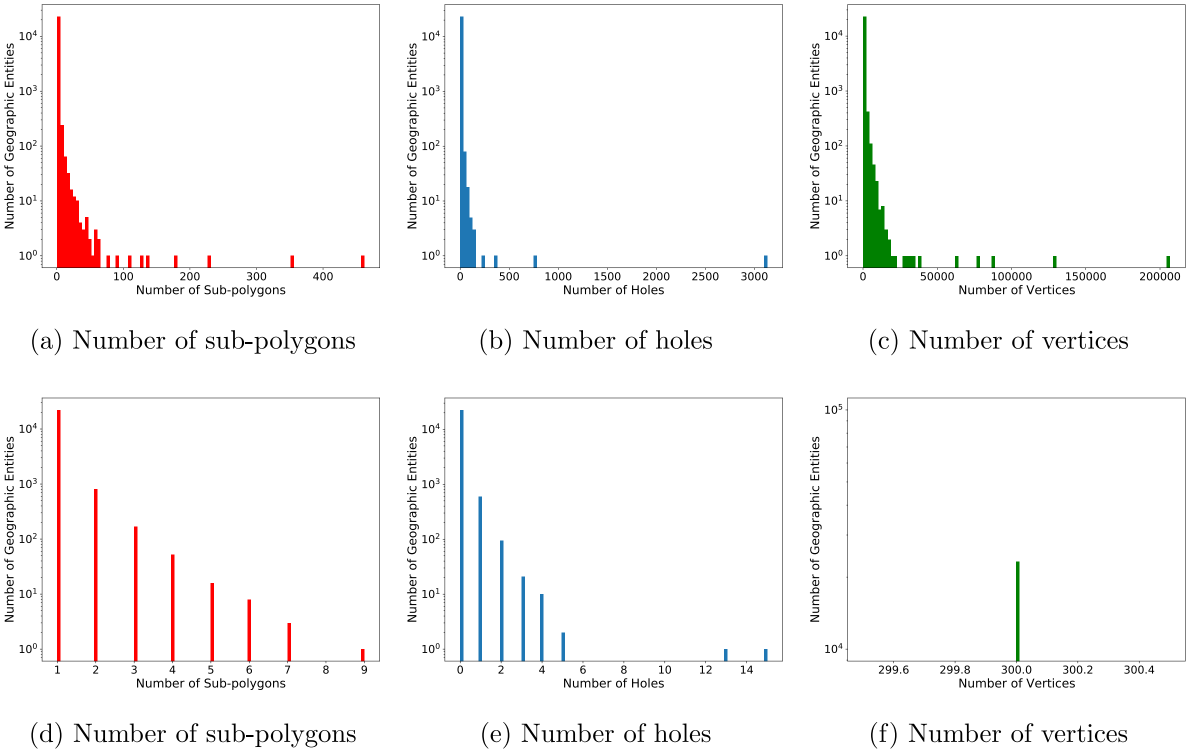
Table 5.5: The place type statistic of geographic entities in DBSpaRel46K and DBSpaRelComplex46K dataset.

Place Type	Count
City	7887
Town	6668
Settlement	3688
Village	2502
AdministrativeRegion	1420
CityDistrict	980
Unknown	57
ProtectedArea	20
BodyOfWater	12
ManMadeFeatures	12
Park	9
HistoricPlace	3
Island	2
Location	2
MountainPass	1
Mountain	1
Total	23264

Table 5.6: The train/valid/test dataset split of DBSpaRel46K and DBSpaRelComplex46K dataset.

Relation	All	Train	Valid	Test
dbo:isPartOf	27364	26164	300	900
dbp:north	2807	1607	300	900
dbp:east	2797	1597	300	900
dbp:south	2770	1570	300	900
dbp:west	2759	1559	300	900
dbp:northwest	2063	863	300	900
dbp:southeast	2024	824	300	900
dbp:southwest	2000	904	274	822
dbp:northeast	1994	1175	205	614
All	46578	36263	2579	7736
ratio	100%	77.85%	5.54%	16.61%

5.8.8 The Complexity of Polygonal Geometries in DBSpaRel-Complex46K



(a) Number of sub-polygons (b) Number of holes (c) Number of vertices

(d) Number of sub-polygons (e) Number of holes (f) Number of vertices

Figure 5.7: A statistic of the complexity of the raw polygonal geometries as well as the simplfied geometries in DBSpaRelComplex46K from OpenStreetMap. (a)-(c) indicate the statistics on the raw polygonal geometries retrieved from OpenStreetMap while (d)-(f) are the same statistics on DBSpaRelComplex46K. (a) & (d) A histogram of the number of sub-polygons per geographic entities. (b) & (e) A histogram of the total number of holes per geographic entities. (c) & (f) A histogram of the total number of unique vertices on each polygonal geometry's exterior and interiors per geographic entities.

5.8.9 Baselines

We use three baselines for the spatial relation prediction tasks:

1. Deterministic: We implement a deterministic baseline based on deterministic topological and cardinal direction spatial relation operations. First, given a triple

(e_{sub}, r_i, e_{obj}), if the geometry of e_{sub} is inside of the geometry of e_{obj}, Deterministic gives $dbo : isPartOf$ as the prediction. Otherwise, Deterministic will compute the geometric center of the subject and object geometry and compute their cardinal direction. This is the exact way how the current SQL or GeoSPARQL-based geographic question answering models [30, 31] do to answer spatial relation questions.

2. VeerCNN: We strictly follow the TensorFlow implementation[9] of Veer et al. [164] and re-implement their model in PyTorch so that it can share the same spatial relation prediction model shown in Equation 5.6 with the other models.

3. DDSL+LeNet5: We directly utilize the original DDSL implementation for the MNIST dataset[10] and modify it for mini-batch training. Since the output of NUFT and inverse Fourier transform of a polygonal geometry is an image, we concatenate the output images of the subject and object geometry on the channel dimension and feed it into the LeNet5 for spatial relation prediction as DDSL does.

The whole model architecture of ResNet1D and NUFTspec is implemented in PyTorch.

[9] `https://github.com/SPINlab/geometry-learning`

[10] `https://github.com/maxjiang93/DDSL`

Chapter 6

Conclusion and Future Work

This dissertation mainly discusses the problem of geographic question answering and proposes a series of spatially-explicit machine learning models to account for the uniqueness of geographic questions. This chapter concludes this dissertation by summarizing its theoretical contributions and practical implications. In the end, we list several limitations and point out the future research directions of GeoQA.

6.1 Summary and Discussions

As one of the fundamental tasks in natural language processing (NLP) and artificial intelligence (AI), question answering (QA) aims at generating answers to natural language questions automatically. An ideal QA system is expected to pass the Turing Test [3] and make itself become indistinguishable from a real human respondent based on their answers. With the development of deep learning technologies, we see substantial progress in the QA research, especially in open domain question answering. In the meantime, recent years have witnessed increasing interests in GeoAI research within and outside of the GIScience domain [181, 182, 58, 183]. There is more and more research studying question answering in a geospatial context [10, 48, 30, 31, 54, 46, 57, 23, 49, 8, 47, 66, 55]. However, in order to establish geographic question answering (GeoQA) as another important research field in GIScience like others such as geographic information retrieval, geo-ontology engineering, and so on, several conceptual and practical questions need to be answered first:

1. What are geographic questions? What is geographic question answering?
2. Why geographic questions are difficult to answer compared to other questions? What is the uniqueness of geographic questions?
3. What type of questions can geographic questions be divided into?
4. What/How can GIScientists contribute to the QA research from the geographic aspect?

This dissertation systematically discusses the problem of geographic question answering with respect to these four questions. Chapter 1 answers the first three conceptual questions by discussing the definition, challenges, uniqueness, as well as the classification of GeoQA. As for the last practical question, Chapter 2, 3, 4, 5 provide a series of spatially-explicit machine learning models as solutions for GeoQA.

Chapter 2 presents a spatially-explicit knowledge graph embedding model called TransGeo for geographic question/query relaxation by doing distance decay-based triple resampling. TransGeo allows a QA system to relax or rewrite a geographic query when it is unanswerable based on the existing QA system. This shows the usability of spatial thinking and spatial knowledge in GeoQA.

After showing the power of spatially-explicit models in GeoQA, we step outside of the GeoQA research and think about a more fundamental way to encode spatial information, especially geographic locations, into machine learning models. By following this thought, Chapter 3 proposes a general-purpose location encoder called Space2Vec which can represent geographic locations in the embedding space. Its effectiveness is demonstrated in the POI type classification task and geo-aware image classification task.

Next, Chapter 4 proposes a location-aware knowledge graph embedding model called SE-KGE based on Space2Vec. We show that by encoding the point coordinates as well as the bounding boxes of geographic entities into the knowledge graph embedding space and training them jointly with the knowledge graph structure information, SE-KGE captures both the spatial information as well as other semantic information among geographic and non-geographic entities. Experiment results show that SE-KGE can outperform multiple baselines on both the geographic query answering task and spatial semantic lifting task.

However, encoding points and bounding boxes of geographic entities is not sufficient for a GeoQA system to answer many geographic questions, especially those that involve topological relations, cardinal direction relations, and so on. So Chapter 5 proposes a general-purpose polygon encoding model which can represent polygonal geometries (either simple polygons, polygons with holes, or multipolygons) into the embedding space.

We show the superiority of this polygon encoder in the polygon-based spatial relation prediction task which is an important but missing component in our current GeoQA solution set.

In summary, this dissertation focuses on using spatial thinking and spatial principles to develop spatially-explicit machine learning models for geographic question answering. Since there are many well-established QA models and GeoQA is a sub-problem of QA, we do not aim at establishing a GeoQA system from scratch. Instead, we focus on the uniqueness of geographic questions and provide a set of spatially-explicit solutions which can be easily integrated into the existing QA models. Interestingly, we also go beyond the problem of GeoQA and provide general-purpose location encoders (Chapter 3) and polygon encoders (Chapter 5) which can be potentially utilized in multiple geospatial tasks in different domains such as Ecology, Earth Science, Environment Science, Urban Data Science, and Human Mobility. We believe these models will have a large impact on the whole geoscience community.

6.2 Research Contribution

The main contribution of this dissertation can be divided into theoretical contributions and practical implications.

6.2.1 Theoretical Contributions

The Uniqueness of Geographic Question Answering. One important theoretical contribution of this dissertation is the deep investigation of the uniqueness of geographic questions and GeoQA. This is the fundamental reason why we want to develop a specialized QA system for geographic questions in the first place. The uniqueness, later on, becomes the guideline of this dissertation.

Spatially-Explicit Machine Learning. The idea of *spatially explicit models* is first proposed by Dr. Micheal Goodchild [184], 'A model is said to be *spatially explicit* when it differentiates behaviors and predictions according to spatial location.' Four tests have been proposed to investigate whether a model is spatially explicit: invariance test, representation test, formulation test, and outcome test. In the context of machine learning and deep learning, we propose the idea of spatially-explicit machine learning models [58] which aim at improving the performance of current state-of-the-art machine learning models by using spatial thinking and spatial inductive bias such as spatial heterogeneity [100, 64, 65], distance decay effect [70, 71, 69, 47, 185, 73, 186], and map projection [187, 188, 189, 190, 191]. This dissertation develops a series of spatially-explicit machine learning models for different sub-problem of GeoQA including geographic query relaxation (Chapter 2), location representation learning (Chapter 3), geographic logic query answering (Chapter 4), polygon representation learning (Chapter 5), and spatial relation prediction (Chapter 5). Our contributions are not only these models themselves, but also the thought process which guides the model development. We hope our proposed models can highlight the importance of spatial thinking and the importance of method development in GIScience research.

Geometric Deep Learning. Compared with many traditional deep models such as convolutional neural networks (CNN) and recurrent neural network (RNN) which are usually deployed on regularly structured data such as grid-like (e.g., images, videos) and sequence-like (e.g., sentences, time series) data, geometric deep learning [142, 192] focuses on developing deep models for non-Euclidean geometric data such as graphs, points, and manifolds. Our location encoders (Chapter 3) and polygon encoders (Chapter 5) follow this idea by focusing on learning representations for two types of irregularly structured data - points and polygonal geometries, which are two fundamental spatial data types.

In fact, most spatial data, especially vector data, are organized in a rather irregular way. We believe geometric deep learning should be a very important research direction for GIScience in the future and this dissertation can be treated as an example to show how spatial thinking can help to develop deep models for non-Euclidean geometric data.

6.2.2 Practical Implications

These spatially-explicit machine learning models proposed in this dissertation also have various potential practical application areas. I list several important applications below.

Spatial Context Modeling. The space-aware graph attention neural network proposed in Section 3.4.2 of Chapter 3 provides a general approach to model the spatial relations between a center point and its spatial neighbors. In real-world applications, many tasks require to do this kind of spatial context modeling. For example, Yan et al. [70] and Yan et al. [71] proposed to use the types of nearby POIs to predict the center POI's type. Sheehan et al. [193] utilized the spatially nearby geolocated Wikipedia articles to predict the wealth index at a specific location. Li et al. [154] used the air pollution measures from nearby air monitoring stations to predict the air pollution measure at the central station. Unlike our model, most previous works did not fully consider the spatial relations between the center and neighboring locations such as directions which might not be optimal for modeling anisotropic processes [194, 195]. For example, as for air pollution forecasting, it is critical to know the wind directios since it will highly affect the air pollution diffusion process. Thus, we believe our space-aware graph attention neural network can benefit a lot of geospatial applications.

Applications of Location Encoding in Different Geoscience Domains. The Space2Vec location encoder proposed in Chapter 3 is general-purpose which means it can be utilized in multiple geospatial applications. In this dissertation, we have already shown the usability of Space2Vec in geospatial tasks from different domains - POI type classification (urban data science), geo-aware image classification (computer vision), geographic question answering (natural language processing), and spatial semantic lifting (knowledge graph). In fact, since geographic locations are the fundamental way how geospatial data is organized, we can use location encoders to handle different types of spatial data such as species occurrences (point patterns), trajectories (polylines), and precipitation map (rasters). So the proposed location encoder can potentially be utilized in multiple domains such as Ecology (e.g., species distribution modeling [119, 196]), human mobility (e.g., traffic forecasting [152, 153], trajectory prediction [197], trajectory synthesis[198]), Meteorology (e.g., precipitation prediction [199]), Oceanography (e.g., sea surface temperature prediction [200]), Geospatial Semantics (e.g., place name disambiguation [201]), Economy (e.g., wealth index prediction [193]), and so on.

Applications of Polygon Encoding in Different Geoscience Domains. Similarly, aside from the spatial relation prediction task discussed in Chapter 5, the proposed general-purpose polygon encoder can also be applied in many polygon-based tasks in different domains such as Urban Planning (e.g., building pattern classification [156], building shape coding [157]), Cartography (e.g., cartographic building generalization [158]), Computer Vision (e.g., shape retrieval [167, 157], shape classification [171]), and so on.

6.3 Limitations and Future Work

Despite the theoretical contributions and practical implications, this dissertation also has several limitations which need to be enhanced in the future. Moreover, there are several exciting future directions we would like to point out.

The Necessity of Spatially-Explicit Machine Learning Models - Tradeoff between Model Performance and Model Complexity & Generalizability. Janowicz et al. [58] challenged the necessity of spatially explicit models by asking, '*what is the tradeoff between designing a machine learning architecture that explicitly accounts for space versus a more general setup that would have to learn to value space implicitly?*' In this dissertation, we have shown a series of spatially-explicit machine learning models which excel at the GeoQA task as well as several other related tasks. However, the price we have to pay is that a spatially-explicit machine learning model is usually more complex and has less generalizability than its counterpart with a more general setup such as the comparison between geographically weighted regression and the normal linear regression. This additional complexity comes from the need to model the spatial aspects such as distance decay, map projection, and so on. The question is whether this complexity is worth paying given the additional benefits brought by spatially-explicit machine learning models. This question has not been answered by this dissertation. And one interesting future research direction is to design *a measure of the worthiness of developing spatially-explicit machine learning models* for a given problem.

Geographic Bias and Social Justice. Data bias such as sampling bias and label imbalance will largely affect the model performance since the bias of training data will cause the representational bias of the resulting machine learning models. Sometimes, this bias becomes not only a technical issue but a social issue as well. For example, Bolukbasi et al. [202] showed that a widely used word embedding model trained on Google News articles, g2vNEWS, exhibits an extreme gender bias and suggests *homemaker* as a career choice for females but *captain* for males. Similarly, racial bias is also a concern when we see Google Photos classifies images of black people as gorillas [1], and the search engine auto-suggests to kill all Jews in the typeahead search [203]. Similarly, geographic bias also becomes a concern for GeoAI models. For instance, Bright et al. [204] have reconfirmed that geographic areas with higher levels of wealth and education typically exhibit higher levels of completeness in OpenStreetMap. Janowicz et al. [141] also showed that DBpedia, one of the world's largest knowledge graphs, has much better geographic coverage in North America and Europe but poor coverage on South America and Russia. This geographic bias will largely affect the machine learning models trained on these open-source datasets. In fact, in our geo-aware image classification experiment discussed in Chapter 3, we also see the negative effects of geographic bias on the performance of location encoders. Our model and all baseline models turn to perform poorly in regions with sparse training data (i.e., species occurrences) while excelling in regions with massive training data. Geographic bias is not a sporadic case but a common issue shared by almost all geospatial datasets. Mitigating geographic bias should not only be a technical requirement but also a way to achieve social justice.

Geographic Artificial Intelligence and Spatial Turing Test. Although using artificial intelligence in geography research has been proposed decades ago by multiple GIScience pioneers such as Smith et al. [205], Couclelis et al. [206], and Openshaw et al. [207], geographic artificial intelligence (GeoAI) has just caught researchers' attention recently because of several major advancements in deep learning research [58]. With the recent development of GeoAI, it seems to be natural for a GIScientist to start to

[1] `https://www.bbc.com/news/technology-33347866`

investigate QA systems, as one important component of AI, in a geographic context. In fact, this is one of the motivations of this dissertation. However, with the development of GeoQA technology, people will not satisfy only asking simple geographic questions such as *how to go to LAX from UCLA*. They expect to ask geographic questions which require more context information and more intelligent computations such as *Find a quiet hotel near Santa Barbara harbor with a great view and a reasonable price.* In fact, a moonshot for GeoQA is to develop an artificial GIS analyst which can answer complex geographic questions automatically and pass a specially-designed spatial Turing Test [58].

Bibliography

[1] L. Gomes, *Machine-learning maestro michael jordan on the delusions of big data and other huge engineering efforts*, 2014.

[2] A. Mishra and S. K. Jain, *A survey on question answering systems with classification*, *Journal of King Saud University-Computer and Information Sciences* **28** (2016), no. 3 345–361.

[3] A. M. Turing, *Computing machinery and intelligence*, *Mind* **59** (1950), no. 236 433–460.

[4] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, *SQuAD: 100,000+ questions for machine comprehension of text*, in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392, 2016.

[5] A. Miller, A. Fisch, J. Dodge, A.-H. Karimi, A. Bordes, and J. Weston, *Key-value memory networks for directly reading documents*, in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1400–1409, 2016.

[6] F. Yang, J. Nie, W. W. Cohen, and N. Lao, *Learning to organize knowledge with n-gram machines*, *arXiv preprint arXiv:1711.06744* (2017).

[7] D. Chen, A. Fisch, J. Weston, and A. Bordes, *Reading wikipedia to answer open-domain questions*, in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, pp. 1870–1879, 2017.

[8] G. Mai, K. Janowicz, C. He, S. Liu, and N. Lao, *POIReviewQA: A semantically enriched POI retrieval and question answering dataset*, in *Proceedings of the 12th Workshop on Geographic Information Retrieval*, p. 5, ACM, 2018.

[9] P. Pasupat and P. Liang, *Compositional semantic parsing on semi-structured tables*, in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, vol. 1, pp. 1470–1480, 2015.

[10] J. M. Zelle and R. J. Mooney, *Learning to parse database queries using inductive logic programming*, in *Proceedings of the national conference on artificial intelligence*, pp. 1050–1055, 1996.

[11] W.-t. Yih, M. Richardson, C. Meek, M.-W. Chang, and J. Suh, *The value of semantic parse labeling for knowledge base question answering*, in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, vol. 2, pp. 201–206, 2016.

[12] C. Liang, J. Berant, Q. Le, K. D. Forbus, and N. Lao, *Neural symbolic machines: Learning semantic parsers on freebase with weak supervision*, in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, pp. 23–33, 2017.

[13] J. Berant, A. Chou, R. Frostig, and P. Liang, *Semantic parsing on freebase from question-answer pairs*, in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1533–1544, 2013.

[14] C. Liang, M. Norouzi, J. Berant, Q. V. Le, and N. Lao, *Memory augmented policy optimization for program synthesis and semantic parsing*, in *Advances in Neural Information Processing Systems*, pp. 10014–10026, 2018.

[15] X. Chen, C. Liang, A. W. Yu, D. Zhou, D. Song, and Q. V. Le, *Neural symbolic reader: Scalable integration of distributed and symbolic representations for reading comprehension*, in *International Conference on Learning Representations*, 2020.

[16] W. Yang, Y. Xie, A. Lin, X. Li, L. Tan, K. Xiong, M. Li, and J. Lin, *End-to-end open-domain question answering with bertserini*, in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pp. 72–77, 2019.

[17] T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee, *et. al.*, *Natural questions: a benchmark for question answering research*, *Transactions of the Association for Computational Linguistics* **7** (2019) 453–466.

[18] A. Asai, K. Hashimoto, H. Hajishirzi, R. Socher, and C. Xiong, *Learning to retrieve reasoning paths over wikipedia graph for question answering*, in *International Conference on Learning Representations*, 2020.

[19] V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-t. Yih, *Dense passage retrieval for open-domain question answering*, in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6769–6781, 2020.

[20] W. Xiong, X. L. Li, S. Iyer, J. Du, P. Lewis, W. Y. Wang, Y. Mehdad, W.-t. Yih, S. Riedel, D. Kiela, *et. al.*, *Answering complex open-domain questions with multi-hop dense retrieval*, *arXiv preprint arXiv:2009.12756* (2020).

[21] H. Sun, B. Dhingra, M. Zaheer, K. Mazaitis, R. Salakhutdinov, and W. Cohen, *Open domain question answering using early fusion of knowledge bases and text*, in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 4231–4242, 2018.

[22] W. Xiong, M. Yu, S. Chang, X. Guo, and W. Y. Wang, *Improving question answering over incomplete kbs with knowledge-aware reader*, in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4258–4264, 2019.

[23] S. Scheider, E. Nyamsuren, H. Kruiger, and H. Xu, *Geo-analytical question-answering with gis*, *International Journal of Digital Earth* (2020) 1–14.

[24] R. Billen and E. Clementini, *A model for ternary projective relations between regions*, in *International Conference on Extending Database Technology*, pp. 310–328, Springer, 2004.

[25] W. Yin, M. Yu, B. Xiang, B. Zhou, and H. Schütze, *Simple question answering by attentive convolutional neural network*, in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pp. 1746–1756, 2016.

[26] K. Janowicz, F. Van Harmelen, J. A. Hendler, and P. Hitzler, *Why the data train needs semantic rails*, *AI Magazine* **36** (2015), no. 1 5–14.

[27] P. Kordjamshidi, J. Pustejovsky, and M. F. Moens, *Representation, learning and reasoning on spatial language for downstream nlp tasks*, in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Tutorial Abstracts*, pp. 28–33, 2020.

[28] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh, *VQA: Visual question answering*, in *Proceedings of the IEEE international conference on computer vision*, pp. 2425–2433, 2015.

[29] T. Ramalho, T. Kočiskỳ, F. Besse, S. Eslami, G. Melis, F. Viola, P. Blunsom, and K. M. Hermann, *Encoding spatial relations from natural language*, *arXiv preprint arXiv:1807.01670* (2018).

[30] W. Chen, *Parameterized spatial SQL translation for geographic question answering*, in *Semantic Computing (ICSC), 2014 IEEE International Conference on*, pp. 23–27, IEEE, 2014.

[31] D. Punjani, K. Singh, A. Both, M. Koubarakis, I. Angelidis, K. Bereta, T. Beris, D. Bilidas, T. Ioannidis, N. Karalis, *et. al.*, *Template-based question answering over linked geospatial data*, in *Proceedings of the 12th Workshop on Geographic Information Retrieval*, pp. 1–10, 2018.

[32] M. F. Worboys, *Nearness relations in environmental space*, *International Journal of Geographical Information Science* **15** (2001), no. 7 633–651.

[33] A. U. Frank, *Qualitative spatial reasoning about distances and directions in geographic space*, *Journal of Visual Languages & Computing* **3** (1992), no. 4 343–371.

[34] B. Regalia, K. Janowicz, and S. Gao, *Volt: a provenance-producing, transparent sparql proxy for the on-demand computation of linked data and its application to spatiotemporally dependent data*, in *European Semantic Web Conference*, pp. 523–538, Springer, 2016.

[35] B. Regalia, K. Janowicz, and G. McKenzie, *Revisiting the representation of and need for raw geometries on the linked data web.*, in *LDOW@ WWW*, 2017.

[36] B. Bennett, *What is a forest? on the vagueness of certain geographic concepts*, in *Topoi*, Citeseer, 2002.

[37] A. G. Cohn, B. Bennett, J. Gooday, and N. M. Gotts, *Qualitative spatial representation and reasoning with the region connection calculus*, *GeoInformatica* **1** (1997), no. 3 275–316.

[38] R. Battle and D. Kolas, *Enabling the geospatial semantic web with parliament and geosparql*, *Semantic Web* **3** (2012), no. 4 355–370.

[39] B. Regalia, K. Janowicz, and G. McKenzie, *Computing and querying strict, approximate, and metrically refined topological relations in linked geographic data*, *Transactions in GIS* **23** (2019), no. 3 601–619.

[40] M. J. Egenhofer and M. P. Dube, *Topological relations from metric refinements*, in *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 158–167, 2009.

[41] W. Kuhn, *Semantic reference systems*, *International Journal of Geographical Information Science* **17** (2003), no. 5 405–409.

[42] D. R. Montello, A. Friedman, and D. W. Phillips, *Vague cognitive regions in geography and geographic information science*, *International Journal of Geographical Information Science* **28** (2014), no. 9 1802–1820.

[43] D. R. Montello, M. F. Goodchild, J. Gottsegen, and P. Fohl, *Where's downtown?: Behavioral methods for determining referents of vague spatial queries*, *Spatial Cognition & Computation* **3** (2003), no. 2-3 185–204.

[44] S. Gao, K. Janowicz, D. R. Montello, Y. Hu, J.-A. Yang, G. McKenzie, Y. Ju, L. Gong, B. Adams, and B. Yan, *A data-synthesis-driven method for detecting and extracting vague cognitive regions*, *International Journal of Geographical Information Science* **31** (2017), no. 6 1245–1271.

[45] W.-t. Yih, M.-W. Chang, X. He, and J. Gao, *Semantic parsing via staged query graph generation: Question answering with knowledge base*, in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 1321–1331, 2015.

[46] E. Hamzei, H. Li, M. Vasardani, T. Baldwin, S. Winter, and M. Tomko, *Place questions and human-generated answers: A data analysis approach*, in *International Conference on Geographic Information Science*, pp. 3–19, Springer, 2019.

[47] G. Mai, B. Yan, K. Janowicz, and R. Zhu, *Relaxing unanswerable geographic questions using a spatially explicit knowledge graph embedding model*, in *AGILE: The 22nd Annual International Conference on Geographic Information Science*, pp. 21–39, Springer, 2019.

[48] W. Chen, E. Fosler-Lussier, N. Xiao, S. Raje, R. Ramnath, and D. Sui, *A synergistic framework for geographic question answering*, in *Semantic Computing (ICSC), 2013 IEEE Seventh International Conference on*, pp. 94–99, IEEE, 2013.

[49] H. Xu, E. Hamzei, E. Nyamsuren, H. Kruiger, S. Winter, M. Tomko, and S. Scheider, *Extracting interrogative intents and concepts from geo-analytic questions*, *AGILE: GIScience Series* **1** (2020) 1–21.

[50] S. Scheider, A. Ballatore, and R. Lemmens, *Finding and sharing gis methods based on the questions they answer*, *International journal of digital earth* **12** (2019), no. 5 594–613.

[51] Y. Hu, K. Janowicz, S. Prasad, and S. Gao, *Metadata topic harmonization and semantic search for linked-data-driven geoportals: A case study using arcgis online*, *Transactions in GIS* **19** (2015), no. 3 398–416.

[52] G. Mai, K. Janowicz, S. Prasad, M. Shi, L. Cai, R. Zhu, B. Regalia, and N. Lao, *Semantically-enriched search engine for geoportals: A case study with arcgis online*, *AGILE: GIScience Series* **1** (2020) 1–17.

[53] Y. Jiang, Y. Li, C. Yang, F. Hu, E. M. Armstrong, T. Huang, D. Moroni, L. J. McGibbney, and C. J. Finch, *Towards intelligent geospatial data discovery: A machine learning framework for search ranking*, *International Journal of Digital Earth* **11** (2018), no. 9 956–971.

[54] Z. Huang, Y. Shen, X. Li, G. Cheng, L. Zhou, X. Dai, Y. Qu, *et. al.*, *Geosqa: A benchmark for scenario-based question answering in the geography domain at high school level*, in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 5869–5874, 2019.

[55] D. Contractor, S. Goel, P. Singla, *et. al.*, *Joint spatio-textual reasoning for answering tourism questions*, *arXiv preprint arXiv:2009.13613* (2020).

[56] D. Contractor, K. Shah, A. Partap, P. Singla, *et. al.*, *Large scale question answering using tourism data*, *arXiv preprint arXiv:1909.03527* (2019).

[57] S. Lobry, D. Marcos, J. Murray, and D. Tuia, *Rsvqa: Visual question answering for remote sensing data*, *IEEE Transactions on Geoscience and Remote Sensing* **58** (2020), no. 12 8555–8566.

[58] K. Janowicz, S. Gao, G. McKenzie, Y. Hu, and B. Bhaduri, *Geoai: Spatially explicit artificial intelligence techniques for geographic knowledge discovery and beyond*, *International Journal of Geographical Information Science* (2020).

[59] K. Khan, B. Baharudin, A. Khan, and A. Ullah, *Mining opinion components from unstructured reviews: A review*, *Journal of King Saud University-Computer and Information Sciences* **26** (2014), no. 3 258–275.

[60] F. Moretti, *Atlas of the European novel, 1800-1900.* London: Verso, 1998.

[61] S. Reddy, D. Chen, and C. D. Manning, *Coqa: A conversational question answering challenge*, *Transactions of the Association for Computational Linguistics* **7** (2019) 249–266.

[62] N. Ngomo, *9th challenge on question answering over linked data (qald-9)*, *language* **7** (2018) 1.

[63] C. Granell, P. G. Pesántez-Cabrera, L. M. Vilches-Blázquez, R. Achig, M. R. Luaces, A. Cortiñas-Álvarez, C. Chayle, and V. Morocho, *A scoping review on the use, processing and fusion of geographic data in virtual assistants*, *Transactions in GIS* (2021).

[64] O. Mac Aodha, E. Cole, and P. Perona, *Presence-only geographical priors for fine-grained image classification*, in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 9596–9606, 2019.

[65] G. Mai, K. Janowicz, B. Yan, R. Zhu, L. Cai, and N. Lao, *Multi-scale representation learning for spatial feature distributions using grid cells*, in *The Eighth International Conference on Learning Representations*, openreview, 2020.

[66] G. Mai, K. Janowicz, L. Cai, R. Zhu, B. Regalia, B. Yan, M. Shi, and N. Lao, *Se-kge: A location-aware knowledge graph embedding model for geographic question answering and spatial semantic lifting*, *Transactions in GIS* **24** (2020), no. 3 623–655.

[67] W. Kuhn, *Core concepts of spatial information for transdisciplinary research*, *International Journal of Geographical Information Science* **26** (2012), no. 12 2267–2276.

[68] W. Kuhn and A. Ballatore, *Designing a language for spatial computing*, in *AGILE 2015*, pp. 309–326. Springer, 2015.

[69] B. Yan, K. Janowicz, G. Mai, and R. Zhu, *A spatially explicit reinforcement learning model for geographic knowledge graph summarization*, *Transactions in GIS* **23** (2019), no. 3 620–640.

[70] B. Yan, K. Janowicz, G. Mai, and S. Gao, *From itdl to place2vec: Reasoning about place type similarity and relatedness by learning embeddings from augmented spatial contexts*, in *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, p. 35, ACM, 2017.

[71] B. Yan, K. Janowicz, G. Mai, and R. Zhu, *xnet+ sc: Classifying places based on images by incorporating spatial contexts*, in *10th International Conference on Geographic Information Science (GIScience 2018)*, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

[72] M. Kejriwal and P. Szekely, *Neural embeddings for populated geonames locations*, in *International Semantic Web Conference*, pp. 139–146, Springer, 2017.

[73] P. Qiu, J. Gao, L. Yu, and F. Lu, *Knowledge embedding with geospatial distance restriction for geographic knowledge graph completion*, *ISPRS International Journal of Geo-Information* **8** (2019), no. 6 254.

[74] G. Mai, K. Janowicz, R. Zhu, L. Cai, and R. Zhu, *Rgeographic question answering: Challenges, uniqueness, classification, and future directions*, in *AGILE: The 24nd Annual International Conference on Geographic Information Science*, 2021.

[75] D. Laurent, P. Séguéla, and S. Nègre, *QA better than IR?*, in *Proceedings of the Workshop on Multilingual Question Answering*, pp. 1–8, Association for Computational Linguistics, 2006.

[76] B. Bennett, D. Mallenby, and A. Third, *An ontology for grounding vague geographic terms.*, in *FOIS*, vol. 183, pp. 280–293, 2008.

[77] S. Elbassuoni, M. Ramanath, and G. Weikum, *Query relaxation for entity-relationship search*, in *Extended Semantic Web Conference*, pp. 62–76, Springer, 2011.

[78] V. S. Pulla, C. S. Jammi, P. Tiwari, M. Gjoka, and A. Markopoulou, *Questcrowd: A location-based question answering system with participation incentives*, in *2013 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 75–76, IEEE, 2013.

[79] S. Scheider, A. Ballatore, and R. Lemmens, *Finding and sharing GIS methods based on the questions they answer*, *International Journal of Digital Earth* (2018) 1–20.

[80] P. Rajpurkar, R. Jia, and P. Liang, *Know what you dont know: Unanswerable questions for squad*, in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 784–789, 2018.

[81] G. Fokou, S. Jean, A. Hadjali, and M. Baron, *Handling failing rdf queries: from diagnosis to relaxation*, *Knowledge and Information Systems* **50** (2017), no. 1 167–195.

[82] M. Wang, R. Wang, J. Liu, Y. Chen, L. Zhang, and G. Qi, *Towards empty answers in sparql: Approximating querying with rdf embedding*, in *International Semantic Web Conference*, pp. 513–529, Springer, 2018.

[83] W. Hamilton, P. Bajaj, M. Zitnik, D. Jurafsky, and J. Leskovec, *Embedding logical queries on knowledge graphs*, in *Advances in Neural Information Processing Systems*, pp. 2027–2038, 2018.

[84] L. Zhang, X. Zhang, and Z. Feng, *TrQuery: An embedding-based framework for recommanding sparql queries*, *arXiv preprint arXiv:1806.06205* (2018).

[85] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, *Translating embeddings for modeling multi-relational data*, in *Advances in neural information processing systems*, pp. 2787–2795, 2013.

[86] Z. Wang, J. Zhang, J. Feng, and Z. Chen, *Knowledge graph embedding by translating on hyperplanes.*, in *AAAI*, vol. 14, pp. 1112–1119, 2014.

[87] G. Mai, K. Janowicz, and B. Yan, *Support and centrality: Learning weights for knowledge graph embedding models*, in *International Conference on Knowledge Engineering and Knowledge Management*, pp. 212–227, Springer, 2018.

[88] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, *Learning entity and relation embeddings for knowledge graph completion.*, in *AAAI*, vol. 15, pp. 2181–2187, 2015.

[89] Q. Wang, Z. Mao, B. Wang, and L. Guo, *Knowledge graph embedding: A survey of approaches and applications*, *IEEE Transactions on Knowledge and Data Engineering* **29** (2017), no. 12 2724–2743.

[90] J. R. Firth, *A synopsis of linguistic theory, 1930-1955*, *Studies in linguistic analysis* (1957).

[91] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, *Distributed representations of words and phrases and their compositionality*, in *Advances in neural information processing systems*, pp. 3111–3119, 2013.

[92] J. Pennington, R. Socher, and C. Manning, *Glove: Global vectors for word representation*, in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

[93] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, *Deep contextualized word representations*, *arXiv preprint arXiv:1802.05365* (2018).

[94] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, *Bert: Pre-training of deep bidirectional transformers for language understanding*, *arXiv preprint arXiv:1810.04805* (2018).

[95] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, *Pointnet: Deep learning on point sets for 3d classification and segmentation*, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 652–660, 2017.

[96] N. Kussul, M. Lavreniuk, S. Skakun, and A. Shelestov, *Deep learning classification of land cover and crop types using remote sensing data*, *IEEE Geoscience and Remote Sensing Letters* **14** (2017), no. 5 778–782.

[97] Y. Li, R. Yu, C. Shahabi, and Y. Liu, *Diffusion convolutional recurrent neural network: Data-driven traffic forecasting*, *arXiv preprint arXiv:1707.01926* (2017).

[98] T. Berg, J. Liu, S. Woo Lee, M. L. Alexander, D. W. Jacobs, and P. N. Belhumeur, *Birdsnap: Large-scale fine-grained visual categorization of birds*, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2011–2018, 2014.

[99] K. Tang, M. Paluri, L. Fei-Fei, R. Fergus, and L. Bourdev, *Improving image classification with location context*, in *Proceedings of the IEEE international conference on computer vision*, pp. 1008–1016, 2015.

[100] G. Chu, B. Potetz, W. Wang, A. Howard, Y. Song, F. Brucher, T. Leung, and H. Adam, *Geo-aware networks for fine-grained recognition*, in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 0–0, 2019.

[101] G. McKenzie, K. Janowicz, S. Gao, J.-A. Yang, and Y. Hu, *Poi pulse: A multi-granular, semantic signature–based information observatory for the interactive visualization of big geosocial data*, *Cartographica: The International Journal for Geographic Information and Geovisualization* **50** (2015), no. 2 71–85.

[102] A. Abbott and E. Callaway, *Nobel prize for decoding brains sense of place*, *Nature News* **514** (2014), no. 7521 153.

[103] H. T. Blair, A. C. Welday, and K. Zhang, *Scale-invariant memory representations emerge from moire interference between grid fields that produce theta oscillations: a computational model*, *Journal of Neuroscience* **27** (2007), no. 12 3211–3229.

[104] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, *Attention is all you need*, in *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.

[105] C. J. Cueva and X.-X. Wei, *Emergence of grid-like representations by training recurrent neural networks to perform spatial localization*, *arXiv preprint arXiv:1803.07770* (2018).

[106] A. Banino, C. Barry, B. Uria, C. Blundell, T. Lillicrap, P. Mirowski, A. Pritzel, M. J. Chadwick, T. Degris, J. Modayil, *et. al.*, *Vector-based navigation using grid-like representations in artificial agents*, *Nature* **557** (2018), no. 7705 429.

[107] R. Gao, J. Xie, S.-C. Zhu, and Y. N. Wu, *Learning grid cells as vector representation of self-position coupled with matrix representation of self-motion*, in *Proceedings of ICLR 2019*, 2019.

[108] G. Baudat and F. Anouar, *Kernel-based methods and function approximation*, vol. 2, pp. 1244 – 1249 vol.2, 02, 2001.

[109] H. J. Bierens, *The nadarayawatson kernel regression function estimator*, *Topics in Advanced Econometrics* **16** (1994) 212247.

[110] V. Maz'ya and G. Schmidt, *On approximate approximations using gaussian kernels*, *IMA Journal of Numerical Analysis* **16** (01, 1996) 13–29.

[111] S. Openshaw, *The modifiable areal unit problem*, *Concepts and techniques in modern geography* (1984).

[112] A. S. Fotheringham and D. W. Wong, *The modifiable areal unit problem in multivariate statistical analysis*, *Environment and planning A* **23** (1991), no. 7 1025–1044.

[113] J. Moat, S. P. Bachman, R. Field, and D. S. Boyd, *Refining area of occupancy to address the modifiable areal unit problem in ecology and conservation*, *Conservation biology* **32** (2018), no. 6 1278–1289.

[114] A. M. Lechner, W. T. Langford, S. D. Jones, S. A. Bekessy, and A. Gordon, *Investigating species–environment relationships at multiple scales: Differentiating between intrinsic scale and the modifiable areal unit problem*, *Ecological Complexity* **11** (2012) 91–102.

[115] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola, *Deep sets*, in *Advances in neural information processing systems*, pp. 3391–3401, 2017.

[116] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, *Graph attention networks*, in *ICLR 2018*, 2018.

[117] G. Mai, K. Janowicz, B. Yan, R. Zhu, L. Cai, and N. Lao, *Contextual graph attention for answering logical queries over incomplete knowledge graphs*, in *Proceedings of the 10th International Conference on Knowledge Capture*, pp. 171–178, 2019.

[118] B. Adams, G. McKenzie, and M. Gahegan, *Frankenplace: interactive thematic mapping for ad hoc exploratory search*, in *Proceedings of the 24th international conference on world wide web*, pp. 12–22, International World Wide Web Conferences Steering Committee, 2015.

[119] W. Zuo, N. Lao, Y. Geng, and K. Ma, *Geosvm: an efficient and effective tool to predict species' potential distributions*, *Journal of Plant Ecology* **1** (2008), no. 2 143–145.

[120] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, *Rethinking the inception architecture for computer vision*, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

[121] M. Nickel, K. Murphy, V. Tresp, and E. Gabrilovich, *A review of relational machine learning for knowledge graphs*, *Proceedings of the IEEE* **104** (2015), no. 1 11–33.

[122] J. Hoffart, F. M. Suchanek, K. Berberich, and G. Weikum, *Yago2: A spatially and temporally enhanced knowledge base from wikipedia*, *Artificial Intelligence* **194** (2013) 28–61.

[123] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, and W. Zhang, *Knowledge vault: A web-scale approach to probabilistic knowledge fusion*, in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 601–610, ACM, 2014.

[124] N. Lao, T. Mitchell, and W. W. Cohen, *Random walk inference and learning in a large scale knowledge base*, in *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, (Edinburgh, Scotland, UK.), pp. 529–539, Association for Computational Linguistics, July, 2011.

[125] M. Nickel, V. Tresp, and H.-P. Kriegel, *Factorizing yago: scalable machine learning for linked data*, in *WWW*, pp. 271–280, ACM, 2012.

[126] M. Nickel, L. Rosasco, T. A. Poggio, *et. al.*, *Holographic embeddings of knowledge graphs.*, in *AAAI*, pp. 1955–1961, 2016.

[127] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, and M. Welling, *Modeling relational data with graph convolutional networks*, in *European Semantic Web Conference*, pp. 593–607, Springer, 2018.

[128] L. Cai, B. Yan, G. Mai, K. Janowicz, and R. Zhu, *Transgcn: Coupling transformation assumptions with graph convolutional networks for link prediction*, in *K-CAP*, ACM, 2019.

[129] K. Janowicz, S. Scheider, T. Pehle, and G. Hart, *Geospatial semantics and linked spatiotemporal data–past, present, and future*, *Semantic Web* **3** (2012), no. 4 321–332.

[130] A. Kristiadi, M. A. Khan, D. Lukovnikov, J. Lehmann, and A. Fischer, *Incorporating literals into knowledge graph embeddings*, in *International Semantic Web Conference*, pp. 347–363, Springer, 2019.

[131] P. Pezeshkpour, L. Chen, and S. Singh, *Embedding multimodal relational data for knowledge base completion*, in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3208–3218, 2018.

[132] R. Zhu, Y. Hu, K. Janowicz, and G. McKenzie, *Spatial signatures for geographic feature types: Examining gazetteer ontologies using spatial statistics*, *Transactions in GIS* **20** (2016), no. 3 333–355.

[133] B. D. Trisedya, J. Qi, and R. Zhang, *Entity alignment between knowledge graphs using attribute embeddings*, in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 297–304, 2019.

[134] A. De Nicola, T. Di Mascio, M. Lezoche, and F. Tagliano, *Semantic lifting of business process models*, in *2008 12th Enterprise Distributed Object Computing Conference Workshops*, pp. 120–126, IEEE, 2008.

[135] B. Schölkopf, *The kernel trick for distances*, in *Advances in neural information processing systems*, pp. 301–307, 2001.

[136] W. Hamilton, Z. Ying, and J. Leskovec, *Inductive representation learning on large graphs*, in *Advances in Neural Information Processing Systems*, pp. 1024–1034, 2017.

[137] B. Scholkopf, K.-K. Sung, C. J. Burges, F. Girosi, P. Niyogi, T. Poggio, and V. Vapnik, *Comparing support vector machines with gaussian kernels to radial basis function classifiers*, *IEEE transactions on Signal Processing* **45** (1997), no. 11 2758–2765.

[138] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, *et. al.*, *Relational inductive biases, deep learning, and graph networks*, *arXiv preprint arXiv:1806.01261* (2018).

[139] W. L. Hamilton, R. Ying, and J. Leskovec, *Representation learning on graphs: Methods and applications*, *arXiv preprint arXiv:1709.05584* (2017).

[140] B. Yang, W.-t. Yih, X. He, J. Gao, and L. Deng, *Embedding entities and relations for learning and inference in knowledge bases*, in *The Third International Conference on Learning Representations*, 2015.

[141] K. Janowicz, Y. Hu, G. McKenzie, S. Gao, B. Regalia, G. Mai, R. Zhu, B. Adams, and K. Taylor, *Moon landing or safari? a study of systematic errors and their causes in geographic linked data*, in *The Annual International Conference on Geographic Information Science*, pp. 275–290, Springer, 2016.

[142] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, *Geometric deep learning: going beyond euclidean data*, *IEEE Signal Processing Magazine* **34** (2017), no. 4 18–42.

[143] F. Monti, D. Boscaini, J. Masci, E. Rodola, J. Svoboda, and M. M. Bronstein, *Geometric deep learning on graphs and manifolds using mixture model cnns*, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5115–5124, 2017.

[144] M. Defferrard, X. Bresson, and P. Vandergheynst, *Convolutional neural networks on graphs with fast localized spectral filtering*, in *NIPS*, 2016.

[145] T. N. Kipf and M. Welling, *Semi-supervised classification with graph convolutional networks*, *arXiv preprint arXiv:1609.02907* (2016).

[146] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, *Pointcnn: Convolution on x-transformed points*, *Advances in neural information processing systems* **31** (2018) 820–830.

[147] J. Masci, D. Boscaini, M. Bronstein, and P. Vandergheynst, *Geodesic convolutional neural networks on riemannian manifolds*, in *Proceedings of the IEEE international conference on computer vision workshops*, pp. 37–45, 2015.

[148] D. Lazer, A. S. Pentland, L. Adamic, S. Aral, A. L. Barabasi, D. Brewer, N. Christakis, N. Contractor, J. Fowler, M. Gutmann, *et. al.*, *Life in the network: the coming age of computational social science*, *Science (New York, NY)* **323** (2009), no. 5915 721.

[149] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, and D. Yin, *Graph neural networks for social recommendation*, in *The World Wide Web Conference*, pp. 417–426, 2019.

[150] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, *Neural message passing for quantum chemistry*, in *ICML*, 2017.

[151] E. H. Davidson, J. P. Rast, P. Oliveri, A. Ransick, C. Calestani, C.-H. Yuh, T. Minokawa, G. Amore, V. Hinman, C. Arenas-Mena, *et. al.*, *A genomic regulatory network for development*, *science* **295** (2002), no. 5560 1669–1678.

[152] Y. Li, R. Yu, C. Shahabi, and Y. Liu, *Diffusion convolutional recurrent neural network: Data-driven traffic forecasting*, in *International Conference on Learning Representations*, 2019.

[153] L. Cai, K. Janowicz, G. Mai, B. Yan, and R. Zhu, *Traffic transformer: Capturing the continuity and periodicity of time series for traffic forecasting*, *Transactions in GIS* **24** (2020), no. 3 736–755.

[154] Y. Lin, N. Mago, Y. Gao, Y. Li, Y.-Y. Chiang, C. Shahabi, and J. L. Ambite, *Exploiting spatiotemporal patterns for accurate air quality forecasting using deep learning*, in *Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 359–368, ACM, 2018.

[155] G. Appleby, L. Liu, and L.-P. Liu, *Kriging convolutional networks*, in *Proceedinngs of AAAI 2020*, 2020.

[156] X. Yan, T. Ai, M. Yang, and H. Yin, *A graph convolutional neural network for classification of building patterns using spatial vector data*, *ISPRS journal of photogrammetry and remote sensing* **150** (2019) 259–273.

[157] X. Yan, T. Ai, M. Yang, and X. Tong, *Graph convolutional autoencoder model for the shape coding and cognition of buildings in maps*, *International Journal of Geographical Information Science* **35** (2021), no. 3 490–512.

[158] Y. Feng, F. Thiemann, and M. Sester, *Learning cartographic building generalization with deep convolutional neural networks*, *ISPRS International Journal of Geo-Information* **8** (2019), no. 6 258.

[159] X. Sun, C. M. Christoudias, and P. Fua, *Free-shape polygonal object localization*, in *European Conference on Computer Vision*, pp. 317–332, Springer, 2014.

[160] L. Castrejon, K. Kundu, R. Urtasun, and S. Fidler, *Annotating object instances with a Polygon-RNN*, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5230–5238, 2017.

[161] D. Acuna, H. Ling, A. Kar, and S. Fidler, *Efficient interactive annotation of segmentation datasets with Polygon-RNN++*, in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 859–868, 2018.

[162] X. Bai, W. Liu, and Z. Tu, *Integrating contour and skeleton for shape classification*, in *2009 IEEE 12th international conference on computer vision workshops, ICCV workshops*, pp. 360–367, IEEE, 2009.

[163] X. Wang, B. Feng, X. Bai, W. Liu, and L. J. Latecki, *Bag of contour fragments for robust shape classification*, *Pattern Recognition* **47** (2014), no. 6 2116–2125.

[164] R. v. Veer, P. Bloem, and E. Folmer, *Deep learning for classification tasks on geospatial vector polygons*, *arXiv preprint arXiv:1806.03857* (2018).

[165] D. A. Randell, Z. Cui, and A. G. Cohn, *A spatial logic based on regions and connection*, in *3rd International Conference on Knowledge Representation and Reasoning*, pp. 165–176, 1992.

[166] R. Battle and D. Kolas, *Geosparql: enabling a geospatial semantic web*, *Semantic Web Journal* **3** (2011), no. 4 355–370.

[167] C. Jiang, D. Lansigan, P. Marcus, M. Nießner, *et. al.*, *DDSL: Deep differentiable simplex layer for learning geometric signals*, in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8769–8778, 2019.

[168] L. J. Latecki, R. Lakamper, and T. Eckhardt, *Shape descriptors for non-rigid shapes with a single closed contour*, in *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, vol. 1, pp. 424–429, IEEE, 2000.

[169] O. Söderkvist, *Computer vision classification of leaves from swedish trees*, *PhD thesis* (2001).

[170] C. Hofer, R. Kwitt, M. Niethammer, and A. Uhl, *Deep learning with topological signatures*, in *NIPS*, 2017.

[171] L. Kurnianggoro, K.-H. Jo, *et. al.*, *A survey of 2d shape representation: Methods, evaluations, and future research directions*, *Neurocomputing* **300** (2018) 1–16.

[172] C. M. Jiang, D. Wang, J. Huang, P. Marcus, and M. Niessner, *Convolutional neural networks on non-uniform geometrical signals using euclidean spectral transformation*, in *International Conference on Learning Representations*, 2019.

[173] O. Rippel, J. Snoek, and R. P. Adams, *Spectral representations for convolutional neural networks*, in *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 2*, pp. 2449–2457, 2015.

[174] J. L. Ba, J. R. Kiros, and G. E. Hinton, *Layer normalization*, *arXiv preprint arXiv:1607.06450* (2016).

[175] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[176] Z. Zhang, S. Fidler, J. Waggoner, Y. Cao, S. Dickinson, J. M. Siskind, and S. Wang, *Superedge grouping for object localization by combining appearance and shape information*, in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3266–3273, IEEE, 2012.

[177] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, *Gated graph sequence neural networks*, in *ICLR 2016*, 2016.

[178] O. Ronneberger, P. Fischer, and T. Brox, *U-net: Convolutional networks for biomedical image segmentation*, in *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, Springer, 2015.

[179] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, *Gradient-based learning applied to document recognition*, *Proceedings of the IEEE* **86** (1998), no. 11 2278–2324.

[180] F. Yu, D. Wang, E. Shelhamer, and T. Darrell, *Deep layer aggregation*, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2403–2412, 2018.

[181] M. N. K. Boulos, G. Peng, and T. VoPham, *An overview of geoai applications in health and healthcare*, *International journal of health geographics* **18** (2019), no. 1 1–9.

[182] Y. Hu, S. Gao, D. Lunga, W. Li, S. Newsam, and B. Bhaduri, *Geoai at acm sigspatial: progress, challenges, and future directions*, *SIGSPATIAL Special* **11** (2019), no. 2 5–15.

[183] W. Li, *Geoai: Where machine learning and big data converge in giscience*, *Journal of Spatial Information Science* **2020** (2020), no. 20 71–77.

[184] M. F. Goodchild and D. G. Janelle, *Thinking spatially in the social sciences*, *Spatially integrated social science* (2004) 3–22.

[185] K. Klemmer and D. B. Neill, *Sxl: Spatially explicit learning of geographic processes with auxiliary tasks*, *arXiv preprint arXiv:2006.10461* (2020).

[186] W. Li, C.-Y. Hsu, and M. Hu, *Toblers first law in geoai: A spatially explicit deep learning model for terrain feature detection under weak supervision*, *Annals of the American Association of Geographers* (2021) 1–19.

[187] T. S. Cohen, M. Geiger, J. Köhler, and M. Welling, *Spherical cnns*, in *International Conference on Learning Representations*, 2019.

[188] C. M. Jiang, J. Huang, K. Kashinath, P. Marcus, M. Niessner, *et. al.*, *Spherical cnns on unstructured grids*, in *International Conference on Learning Representations*, 2019.

[189] M. Defferrard, M. Milani, F. Gusset, and N. Perraudin, *Deepsphere: a graph-based spherical cnn*, in *International Conference on Learning Representations*, 2020.

[190] M. Izbicki, E. E. Papalexakis, and V. J. Tsotras, *Exploiting the earths spherical geometry to geolocate images*, in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 3–19, Springer, 2019.

[191] M. Izbicki, V. Papalexakis, and V. Tsotras, *Geolocating tweets in any language at any location*, in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pp. 89–98, 2019.

[192] M. M. Bronstein, J. Bruna, T. Cohen, and P. Velikovi, *Geometric deep learning: Grids, groups, graphs, geodesics, and gauges*, *arXiv preprint arXiv:2104.13478* (2021).

[193] E. Sheehan, C. Meng, M. Tan, B. Uzkent, N. Jean, M. Burke, D. Lobell, and S. Ermon, *Predicting economic development using geolocated wikipedia articles*, in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2698–2706, 2019.

[194] G. Mai, K. Janowicz, Y. Hu, and S. Gao, *Adcn: An anisotropic density-based clustering algorithm for discovering spatial point patterns with noise*, *Transactions in GIS* **22** (2018), no. 1 348–369.

[195] R. Zhu, K. Janowicz, and G. Mai, *Making direction a first-class citizen of tobler's first law of geography*, *Transactions in GIS* **23** (2019), no. 3 398–416.

[196] E. Cole, B. Deneu, T. Lorieul, M. Servajean, C. Botella, D. Morris, N. Jojic, P. Bonnet, and A. Joly, *The geolifeclef 2020 dataset*, *arXiv preprint arXiv:2004.04192* (2020).

[197] Y. Xu, Z. Piao, and S. Gao, *Encoding crowd interaction with deep neural network for pedestrian trajectory prediction*, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5275–5284, 2018.

[198] J. Rao, S. Gao, Y. Kang, and Q. Huang, *LSTM-TrajGAN: A deep learning approach to trajectory privacy protection*, in *Proceedings of the 11th International Conference on Geographic Information Science, Poznan, Poland*, pp. 12:1–12:17, 2020.

[199] S. Agrawal, L. Barrington, C. Bromberg, J. Burge, C. Gazen, and J. Hickey, *Machine learning for precipitation nowcasting from radar images*, *arXiv preprint arXiv:1912.12132* (2019).

[200] P. P. Sarkar, P. Janardhan, and P. Roy, *Prediction of sea surface temperatures using deep learning neural networks*, *SN Applied Sciences* **2** (2020), no. 8 1–14.

[201] Y. Ju, B. Adams, K. Janowicz, Y. Hu, B. Yan, and G. McKenzie, *Things and strings: improving place name disambiguation from short texts by combining entity co-occurrence with topic modeling*, in *European Knowledge Acquisition Workshop*, pp. 353–367, Springer, 2016.

[202] T. Bolukbasi, K.-W. Chang, J. Y. Zou, V. Saligrama, and A. T. Kalai, *Man is to computer programmer as woman is to homemaker? debiasing word embeddings*, *Advances in Neural Information Processing Systems* (2016).

[203] K. Janowicz, B. Yan, B. Regalia, R. Zhu, and G. Mai, *Debiasing knowledge graphs: Why female presidents are not like female popes.*, in *International Semantic Web Conference (P&D/Industry/BlueSky)*, 2018.

[204] J. Bright, S. De Sabbata, and S. Lee, *Geodemographic biases in crowdsourced knowledge websites: Do neighbours fill in the blanks?*, *GeoJournal* **83** (2018), no. 3 427–440.

[205] T. R. Smith, *Artificial intelligence and its applicability to geographical problem solving*, *The Professional Geographer* **36** (1984), no. 2 147–158.

[206] H. Couclelis, *Artificial intelligence in geography: Conjectures on the shape of things to come*, *The professional geographer* **38** (1986), no. 1 1–11.

[207] S. Openshaw and C. Openshaw, *Artificial intelligence in geography.* John Wiley & Sons, Inc., 1997.