University of California
Santa Barbara

Time Representation and Reasoning over Knowledge Graphs

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

Doctor of Philosophy
in
Geography

by

Ling Cai

Committee in charge:

Professor Krzysztof Janowicz, Chair
Professor Konstadinos Goulias
Professor Kelly Caylor

December 2022

The Dissertation of Ling Cai is approved.

Professor Konstadinos Goulias

Professor Kelly Caylor

Professor Krzysztof Janowicz, Committee Chair

December 2022

Time Representation and Reasoning over Knowledge Graphs

by

Ling Cai

To my beloved family for their love and support.

Acknowledgements

When I look at this title *acknowledgement*, it reminds me of the fact that my chapter at University of California, Santa Barbara as a Ph.D. student is close to an end. I have a mixed feeling - a mixture of happiness, pity, fearfulness and so on. Many past memories are crowding in upon my mind, and I am not sure where to start. There are so many memories that record all the ups and downs, and so many people who have made this challenging but unforgettable journey colorful. I apologize if words are too pale and too powerless, but I really want to express my deepest gratitude to all of you.

I would like to thank my committee members for their invaluable input and advice over the past few years. Working with Jano is absolutely exceptional. He is always active in all sorts of discussion and always open to new ideas. His passion for research is overwhelming. Sometimes it even becomes the driving force of my own research and motivates me to think harder and dig in deeper. He is definitely a critical thinker and an excellent mentor. He always questions about proposed ideas, through which he guides you to approaching and thinking about research questions from different perspectives across different domains. I really liked the way he mentored me. I can feel it is his critical thinking and this way of mentoring that makes me become an independent researcher. His ability to manage time and projects is also impressive and is far beyond me. I am not sure how he could maintain the balance between life, student mentoring, course teaching, project management, and so on and so forth. Probably the rumor is true that he has 25 hours per day, or he must have acquired some magic I don't have.

I am also very glad to learn from my other two committee members: Konstadinos Goulias and Kelly Caylor. Konstadinos Goulias is always humorous and patient while providing invaluable suggestions. His course of Analytical Methods is one of the best I have taken during my Ph.D. His positive altitude towards research and life really inspires me. I am very grateful for having Kelly Caylor in my committee. Although his expertise is outside of my primary field, he always loves to understand my work and provides as much help as he could. Their advice during my Ph.D. exams helps me clarify and convey my research clearly and accurately.

I would also love to express my gratitude to my colleagues at STKO. I am very glad to have all of you as great teachers and helpful friends in my journey. Many thanks to Gengchen Mai and Bo Yan. Gengchen is a smart and meticulous researcher. He is always patient and willing to help me figure out all the details from research ideas to experiments. His productivity and ambition for research is inspiring. Bo is an expert in the machine learning field, and gave me lots of guidance in the beginning of my research. He is always smart and knows what he wants. I really appreciate his continuous help even after he graduated from our lab. I also want to thank Rui Zhu. He is among a few researchers who still care about foundational research questions of GIScience, and wants to contribute to theoretical researches. I really enjoyed the old times when we discussed exciting papers and exchanged ideas. Those moments relieved me from technical details and instead provided opportunities to think freely. Additionally, I want to thank other colleagues: Meilin Shi, Zilong Liu, Zhangyu Wang and Kitty, for their helpful input and company.

I also want to thank the friends I met at UCSB for their support. Thank you, Jinglei Yang. I am so glad to meet you and become a good friend with you. The first two years were so colorful because of your company. Thank you, Yuqing Zhu. Thanks for being such a great roommate and providing me such a cozy living environment. Living with you is always comfortable and relaxing. Thank you, Rongxiang Su. I enjoyed the meals you made and admire your life as always. I will definitely miss your cooking. Thank you, my boyfriend Yutao Zhou. It was an amazing encounter with you and it is my fortune. Thank you for being the best boyfriend I could ever ask for. Thank you for your care, patience, encouragement and love.

Last and most importantly, I am so grateful for my whole family for their unconditional support, selfless care and generous love over the years. Thanks, my grandparents, for being my friends and always being there for me. I am sorry that I cannot come back and see you for the last time, my grandma, but I will always miss you. Thanks, Daddy, Mommy and my younger sister, for respecting and supporting every decision I have made. You are the most caring and supportive family that I could ever wish for. I will always credit my dedicated family for all my achievements I could make.

Curriculum Vitæ
Ling Cai

Education

2022	Ph.D. in Geography, University of California, Santa Barbara.
2018	M.S. in GIScience, Chinese Academy of Sciences.
2015	B.S. in GIScience, Wuhan University.

Publications

Peer-reviewed journal articles

- **Ling Cai**, Krzysztof Janowicz, Rui Zhu, Yan Bo, Gengchen Mai.(2022): HyperQuaternionE: A Hyperbolic Embedding Model for Qualitative Spatiotemporal Reasoning, *Geoinformatica*, 1-39.
- Gengchen Mai, Chiyu Max Jiang, Weiwei Sun, Rui Zhu, Yao Xuan, **Ling Cai**, Krzysztof Janowicz, Stefano Ermon, Ni Lao.(2022): Towards General-Purpose Representation Learning of Polygonal Geometries. *GeoInformatica*.
- Gengchen Mai, Janowicz Krzysztof, Yingjie Hu, Song Gao, Bo Yan, Rui Zhu, **Ling Cai**, and Ni Lao.(2022): A Review of Location Encoding for GeoAI: Methods and Applications. *International Journal of Geographical Information Science*, 36(4): 639-673.
- Gengchen Mai, Weiming Huang, **Ling Cai**, Rui Zhu, Ni Lao.(2022): Narrative Cartography with Knowledge Graphs. *Journal of Geovisualization and Spatial Analysis*, 6(1): 1-24.
- Krzysztof Janowicz, Cogan Shimizu, Pascal Hitzler, Gengchen Mai, Shirly Stephen, Rui Zhu, **Ling Cai**, Lu Zhou, Mark Schildhauer, Zilong Liu, Zhangyu Wang, Meilin Shi.(2022): Diverse Data! Diverse Schemata?.*Semantic Web Journal*, 13(1): 1-3.
- **Ling Cai**, Krzysztof Janowicz, Gengchen Mai, Bo Yan, Rui Zhu.(2020): Traffic Transformer: Capturing the Continuity and Periodicity of Time Series for Traffic Forecasting, *Transaction in GIS*, 24(3): 736-755.
- Krzysztof Janowicz, Pascal Hitzler, Wenwen Li, Dean Rehberger, Mark Schildhauer, Rui Zhu, Cogan Shimizu, Colby Fisher, **Ling Cai**, Gengchen Mai, Joseph Zalewski, Lu Zhou, Shirly Stephens, Seila Gonzalez, AnnaLopez Carr, Andrew Schroeder, Dave Smith, Dawn Wright, Sizhe Wang, Yuanyuan Tian, Zilong Liu.(2022): Know, Know Where, KnowWhereGraph: A Densely Connected, Cross-Domain Knowledge Graph and Geo-Enrichment Service Stack for Applications in Environmental Intelligence. *AI Magazine*.
- **Ling Cai**, Jun Xu, Ju Liu, Ting Ma, Tao Pei, Chenghu Zhou.(2019): Sensing multiple semantics of urban space from crowdsourcing positioning data. *Cities*, 93: 31-42.

- **Ling Cai**, Jun Xu, Ju Liu, Tao Pei.(2018): Integrating spatial and temporal contexts into a factorization model for POI recommendation, *International Journal of Geographical Information Science*, 32(3), 524-546.
- **Ling Cai**, Jun Xu, Aoyong Li.(2017): Hybrid group recommendation with enhanced group preferences. *Computer Engineering and Applications*, 53(9):5-10.
- Gengchen Mai, Krzysztof Janowicz, **Ling Cai**, Rui Zhu, Ni Lao.(2020). SE-KGE: A Location-Aware Knowledge Graph Embedding Model for Geographic Question Answering and Spatial Semantic Lifting, *Transaction in GIS*, 24(3): 623-655.
- Rui Zhu, Krzysztof Janowicz, **Ling Cai** and Gengchen Mai.(2022) Representing and Reasoning Qualitative Higher-order Spatial Relations in Geospatial Knowledge Graphs, *International Journal of Geographical Information Science*, 1-32.
- Ju Liu, Jun Xu, **Ling Cai**, Bin Meng,Tao Pei.(2018). Identifying Functional Regions Based on the Spatio-temporal Pattern of Taxi Trajectories. *Journal of Geo-information Science*, 20(11): 1550-1561.

Peer-reviewed conference articles

- **Ling Cai**, Krzysztof Janowicz, Rui Zhu.(2022): Automatically Discovering Conceptual Neighborhood Graphs Using Machine Learning Methods for Spatiotemporal reasoning, *In Proceedings of the 15th International Conference on Spatial Information Theory (COSIT 2022)*, September 5-9, 2022, Kobe, Japan.
- Zilong Liu, Krzysztof Janowicz, **Ling Cai**, Rui Zhu, Gengchen Mai and Meilin Shi. Geoparsing: Solved or Biased? An Evaluation of Geographic Biases in Geoparsing, *In Proceedings of AGILE 2022.*
- Rui Zhu, Krzysztof Janowicz, Gengchen Mai, **Ling Cai** and Meilin Shi.(2022): COVID-Forecast-Graph: An Open Knowledge Graph for Consolidating COVID-19 Forecasts and Economic Indicators via Place and Time, *In Proceedings of AGILE 2022.*
- **Ling Cai**, Krzysztof Janowicz, Bo Yan, Rui Zhu, and Gengchen Mai.(2021): Time in a Box: Advancing Knowledge Graph Completion with Temporal Scopes. *In Proceedings of the 11th Knowledge Capture Conference (K-CAP '21)*, December 2–3, 2021, Virtual Event, USA. ACM, New York, NY, USA.
- Rui Zhu, Cogan Shimizu, Shirly Stephen, Lu Zhou, **Ling Cai**, Gengchen Mai, Krzysztof Janowicz, Mark Schildhauer, Pascal Hitzler.(2021): SOSA-SHACL: Shapes Constraint for the Sensor, Observation, Sample, and Actuator Ontology, *In Proceedings of IJCKG 2021.*
- Rui Zhu, Shirly Ambrose, Lu Zhou, Cogan Shimizu, **Ling Cai**, Gengchen Mai, Krzysztof Janowicz, Pascal Hitzler, Mark Schildhauer.(2021): Environmental Observations in Knowledge Graphs, *In: DaMaLOS 2021 @ ISWC 2021*, October 24, 2021, Virtual Conference.

- Meilin Shi, Krzysztof Janowicz, **Ling Cai**, Gengchen Mai, and Rui Zhu.(2021): A Socially Aware Huff Model for Destination Choice in Nature-based Tourism. *AGILE 2021*: 1-16.
- Gengchen Mai, Krzysztof Janowicz, Rui Zhu, **Ling Cai**, Ni Lao. Geographic Question Answering: Challenges, Uniqueness, Classification, and Future Directions, *AGILE 2021.*
- **Ling Cai***, Rui Zhu*, Gengchen Mai, Cogan Shimizu, Colby K. Fisher, Krzysztof Janowicz, Anna Lopez-Carr, Andrew Schroeder, Mark Schildhauer, YuanyuanTian, Shirly Stephen, and Zilong Liu.(2021): Providing Humanitarian Relief Support through Knowledge Graphs. *In Proceedings of the 11th Knowledge Capture Conference (K-CAP '21)*, December 2–3, 2021, Virtual Event, USA.ACM, New York, NY, USA. (*equal contribution)
- Gengchen Mai, Krzysztof Janowicz, Sathya Prasad, Meilin Shi, **Ling Cai**, Rui Zhu, Blake Regalia, Ni Lao.(2020): Semantically-Enriched Search Engine for Geoportals: A Case Study with ArcGIS Online, *AGILE 2020.*
- **Ling Cai**, Bo Yan, Gengchen Mai, Krzysztof Janowicz, Rui Zhu. (2019): TransGCN:Coupling Transformation Assumptions with Graph Convolutional Networks for Link Prediction. *In Proceedings of the 10th Knowledge Capture Conference (K-CAP '19)*, November 19 - 21, 2019, Marina del Rey, CA, USA.
- Gengchen Mai, Krzysztof Janowicz, Bo Yan, Rui Zhu, **Ling Cai**, Ni Lao.(2019): Contextual Graph Attention for Answering Logical Queries over Incomplete Knowledge Graphs. *In Proceedings of the 10th Knowledge Capture Conference (K-CAP '19)*, November 19 - 21, 2019, Marina del Rey, CA, USA.
- Gengchen Mai, Krzysztof Janowicz, Bo Yan, Rui Zhu, **Ling Cai**, Ni Lao.(2020): Multi-Scale Representation Learning for Spatial Feature Distributions using Grid Cells, *In Proceedings of ICLR 2020*, April 27-30, 2020, Addis Ababa, Ethiopia.

Professional Experience

09/2018-12/2022	Research Assistant, University of California, Santa Barbara.
07/2019-09/2019	Machine Learn Intern, Siemens.
06/2022-09/2022	Research Engineer Intern, IBM.

Awards

2019	*Jack and Laura Dangermond Travel Award*, Department of Geography, University of California, Santa Barbara.
2019	*Best Paper Award* at the 10th International Conference on Knowledge Capture, Los Angeles, United States.
2020	*Spotlight Paper* at the 8th International Conference on Learning Representations, virtual event.

2021 *Best Paper Nominee* at AGILE 2021, virtual event.

2021 *Best Paper Award* at the 11th International Conference on Knowledge Capture, virtual event.

2022 *Best Paper Nominee* at the 15th International Conference on Spatial Information Theory, Kobe, Japan.

Grant

2019-2020 Microsoft AI for Earth Grant: Gengchen Mai, Krzysztof Janowicz, Ni Lao, WenyunZuo, **Ling Cai** (Co-PI). Deep species spatio-temporal distribution modeling for biodiversity hotspot prediction.

Abstract

Time Representation and Reasoning over Knowledge Graphs

by

Ling Cai

Knowledge Representation and Reasoning (KRR) is an interdisciplinary research field dedicated to the formalization and conceptualization of knowledge in order to enable machines to solve tasks that require logical deduction/induction and to help people retrieve information from KR systems in more intuitive ways. Despite their success stories in semantic search, semantic parsing, question answering, recommender systems, etc., one commonly neglected aspect is that the world is ever-changing, and thus, statements about it may only hold during a certain time period. Unfortunately, temporal information is often inaccurate, incomplete and of different types and forms (e.g., quantitative time versus temporal relations). This poses great challenges to conventional rule-based reasoning systems (i.e., symbolic reasoning), which are deterministic, and unable to address noise (errors or incompleteness). In order to address these challenges, this dissertation focuses on developing new methods to represent and reason about temporal information in a subsymbolic manner. Chapter 3 and Chapter 4 study how to subsymbolically represent various temporal primitives (i.e., time instants and intervals). Chapter 5 shifts focus to qualitative temporal relations and presents a subsymbolic approach to perform (spatio-) temporal reasoning. Chapter 6 investigates why subsymbolic approaches outperform conventional methods in terms of qualitative (spatio-) temporal reasoning, and finds conceptual neighborhood of qualitative relations can be discovered by subsymbolic approaches. Throughout the dissertation, I focus on how domain theory can be used in subsymbolic methods and how theories can be discovered by subsymbolic methods.

Contents

Chapter 1

Introduction

This chapter starts with introducing background materials about geographic information representation, in particular temporal aspects. Then I will present limitations of conventional symbolic methods when reasoning about time. This will motivate the primary research goal of this dissertation - *how to devise new ways to represent and reason about temporal information*. Next, I introduce subsymbolic approaches popular in representation learning (e.g., Machine Learning/Deep Learning techniques), which provide potentials to tackle limitations in conventional methods for representation and reasoning over temporal information. Finally, I propose four separate but related research questions to achieve my primary research goal. The structure of this dissertation is outlined in the end.

1.1 Background

The big research questions of GIScience are all centered around how to collect, represent, store, manipulate, analyze, visualize and understand geographic data and the entities and processes they represent [1, 2]. The past decades have witnessed a massive proliferation of studies on how to computationally represent rich phenomena such places, events, cultural differences, and so on. Geospatial ontologies and geographic knowledge graphs in particular have attracted substantial attention by practitioners as they facilitate conceptualization, data accessibility, and semantic interoperability [3, 4, 5]. Such knowledge graphs also align closely to the rapidly increasing need for FAIR-based research data management [6].

Knowledge graphs (KGs) as sets of statement structured knowledge about the world in the form of machine understandable and reason-able triples (most often in form <*subject, predict, object*>). By their interconnected nature, KGs usually contain various types of information from different domains, e.g., providing cross-walks between geography, supply chains, disaster relief, and so forth. Geographic knowledge graphs typically appear as a subgraph of such cross-domain KGs (such as Wikidata[1] and DBpedia[2]). Concretely, geographic information usually describes relationships between/within geographic concepts and geographic entities, and thus can be represented as geographic statements in the form of triples (e.g., <*Santa Barbara, partOf, California*>). By using such symbolic representation, the semantics of entities and their interactions are formally preserved in KGs (more specifically, in the ontologies that provide schema for these KG). Thus it facilitates humans and machines to perform logic inference to derive new information based on ontologies, axioms and/or rules. For example, given that <*Santa Barbara, partOf, California*> and <*California, partOf, USA*>, we can derive <*Santa Barbara, partOf,*

[1] `https://www.wikidata.org/`

[2] `https://www.dbpedia.org/`

USA> by using the transitive property of the relation *partOf*. Such symbolic way of representing and reasoning over existing information falls into the well-known realm of symbolic artificial intelligence (AI).

However, the world is ever-changing, and thus statements about the world are not static. That is, statements are often temporally scoped, meaning that they are only valid during certain time periods or at some points[7, 8]. For example, the statement that the European Union (EU) has diplomatic relation to Norway is only true after 1994, and the statement that the United Kingdom is part of the EU only holds during a certain period (i.e., 1973 - 2020). Without considering the validity period of statements, we may easily arrive at wrong answers in inference and reasoning. For example, we may ask this question: *find me countries that are members of the EU and share borders with it.* Apparently, we, human beings, easily know there is no correct answer since we know no country can be part of EU and share (exterior) borders with it simultaneously[3]. Our thinking and reasoning process implicitly uses the temporal dimension. However, when machines are used for answering such queries, incorrect answers may be returned, if the temporal information is not expressed explicitly and properly in knowledge representation. For example, Spain and the UK will be returned but both are incorrect, as the former shared borders with the EU before joining and the latter shares borders after exiting the EU. Temporal information needs to be carefully addressed in reasoning when the validity of statements is subject to change. However, addressing temporal information is not as trivial as it appears.

[3] The territory of the EU consists of all its member countries. So we will not say the EU shares borders with its members.

1.2 Challenges

The temporal dimension brings up new challenges for classic knowledge representation and reasoning in particular. First, the validity period of a statement in a KG is often missing and exhibits diverse forms (e.g., time instants, semi-closed intervals and closed intervals). As a direct result, it is difficult to tell apart whether statements in a KG are truly atemporal or time-aware in the first place. Take the following two statements as examples *<Sun, instanceOf, Star>* and *<UK, containsAdministrativeTerritorialEntity, Southern Ireland>*. The first statement is *golden* true and thus no temporal scope is needed. For the second statement, it only holds during the time period from May 3rd 1921 to December 6th 1922. However, due to missing information, these two types of statements are mixed up and we cannot distinguish whether they are temporal statements or not. Similarly, semi-open intervals are also ambiguous, because the semi-openness may either be a true validity period for a statement or caused by missing information on the other end. Some statements may have been valid for a long time and we may be unable to know when they would turn invalid. For instance, the statement that Norway is a member of the United Nations has been valid since it joined in 1945. Thus, a semi-open interval as the validity period will be reasonable and accurate. However, there are cases when semi-open intervals are caused by missing information. For instance, from Wikidata, we only know that Ukraine ceased to share borders with Czechoslovakia after 1993, but do not know when it started. The incompleteness of temporal information makes classic rule-based methods difficult on the one hand. On the other hand, it necessitates a new task of predicting missing validity period of statements, which cannot be accomplished by classic symbolic approaches.

In addition to validity periods of statements (one kind of temporal information), other types of temporal information are also abundant in KGs and potentially useful for reasoning. Temporal literals are one type of temporal information used to describe the start or end date of entities/concepts rather than statements. For example, *<USA, inception, 1776>*. With this being known, we can infer that New York City was not part of USA in 1744. Another type of temporal information in KGs is qualitative temporal relations, which describe the temporal relationships (e.g., during, before, after, etc) between entities (events). For example, *<Cold War, follows, World War II>*. Such ordinal relationships are useful for cause-effect inference. Last but not least, temporal patterns are common in KGs as well, seasons being an intuitive example. For instance, the presidency in the US is limited to at most 8 years (i.e., 2 terms). In order to make use of such temporal information, classic symbolic reasoners usually require human efforts to discover and conclude associations/rules between entities. Therefore, this sets up a barrier for classic reasoning methods as far as the scale of KGs is concerned. Additionally, no existing classic reasoning is able to make full use of all these different types of temporal information as different types of time representation and levels of abstraction are needed.

Furthermore, erroneous temporal information is common due to mistakes in data collection and open information extraction [9, 10]. For example, in some statements, the end time of their validity periods are earlier than their start time, which is clearly wrong. Intuitively we may easily fix them by switching the start time and end time; however, it is hard to ensure this resulting validity period is correct. This poses another treat at classic reasoning methods that are prone to errors and uncertainty.

1.3 Symbolic AI versus Subsymbolic AI

The section above summarizes challenges that classic symbolic inference and reasoning approaches (a.k.a. Symbolic AI) may encounter when temporal information is missing, contains errors, or includes uncertainty. Moreover, the symbolic and logical nature of knowledge representation in KGs prohibits information/knowledge in KGs from being applied to various applications, such as recommendation systems and urban computing, which involve numeric computation in a continuous vector space [11]. Therefore, we need new methods for representing time as well as reasoning over temporal knowledge in KGs.

Recent advances in Machine Learning (ML) and Deep Learning (DL), embedding techniques in particular, have given rise to new solutions. Rather than using symbolic representation, the key idea of such methods is to represent everything as numeric vectors (i.e., subsymbolic representations) in high-dimensional vector spaces. To implement this idea, different approaches have been developed to learn such vector representations automatically through being optimized towards an objective that best captures the characteristics of data. Then a reasoning task can be transformed to perform vector algebra over those numeric representations; thus called subsymbolic reasoning. For KGs specifically, both entities and relations are represented as continuous high-dimensional real-value vectors, which are learned automatically while preserving the underlying graph structures of KGs [12, 13]. Then reasoning or inference over KGs boils down to performing vector operations (such as multiplication, addition, subtraction, etc.) over vector representations of entities and relations. This group of subsymbolic methods is known as Knowledge Graph Embedding methods (KGE), which will be detailed later in Section 2.3.

The advantages of KGE have been demonstrated in prior works in many aspects [14, 15, 16, 17]. First, KGE models are tolerant to (small) errors, because they learn continuous representations of entities and relations from data by optimizing a global objective[18]. Meanwhile, the learned representation encodes both the local and global graph information between entities and thus is more comprehensive. Second, unlike symbolic reasoning methods, they are able to deal with incomplete and uncertain information, as explicit rules or axioms are not necessary[19]. Third, thanks to the semantic information encoded, the continuous representations can be regarded as external information and used in various downstream tasks, such as recommendation systems, question answering, language modeling[20, 21, 22]. Fourth, KGE models are implicitly non-monotonic inference systems [23] in contrast to classic symbolic reasoners, which are monotonic. In a monotonic system, statements that are true will never be falsified. Clearly, this is unrealistic since old statements are likely to be proven false.

1.4 Research Questions

Despite the fact that KGEs have made massive breakthrough in generic knowledge representation and reasoning, research on the temporal aspect of KGs by using subsymbolic approaches is still at an early stage. One general question I set out to answer in this dissertation is how to accomplish a subsymbolic temporal reasoning system that can address challenges appearing in classic time representation and reasoning. Such a question is very substantial and thus should be addressed step by step. Humans often learn through analogies. I draw insights from a mature symbolic temporal reasoning system and decompose my research question into small ones by analogy. According to Vila [24], a symbolic temporal reasoning system usually consists of three components, including identifying ontological primitives of time, providing ways to formalize temporal information (i.e., how to build the linkage between an atemporal statement and a temporal reference), and devising methods to reason about temporal knowledge (i.e., the mechanism and approaches used to reason). More details regarding symbolic temporal reasoning are presented in Section 2.2. Therefore, I propose to study each component in a subsymbolic temporal reasoning system accordingly, which corresponds to three research questions listed below. In addition, a fourth research question is proposed to examine why subsymbolic temporal reasoning methods perform better than symbolic reasoning methods.

Question 1: *In a subsymbolic reasoning system, what should be the ontological primitive of time? How to represent the temporal primitive in a subsymbolic way?*

This research question focuses on how to build a proper time reference for temporal reasoning, namely what elements and relations should be chose to constitute such a time reference. This is the most fundamental question for temporal reasoning because it is the basis for subsequent time formalization and reasoning. In the study of symbolic temporal reasoning, it was not until the early 80s that general theories of time for symbolic temporal reasoning have been proposed[24, 25], such as temporal logic [26], Allen's general theory of time and action [27], and theories of time [28]. At that time, prior researchers encountered the problem of deciding on the ontological primitives for a time ontology. Two dominating contenders are time instants (i.e., points in time) and periods (i.e., intervals of time), while a third is to take them together. For instance, McCarthy used time instants to develop Situational Calculus for temporal reasoning [25]. In this case, time is modeled as an infinite line, consisting of an infinite and dense collection of instants. However, Allen argued that time intervals/periods should be the primitive as they are inline with our understanding of temporal concepts, and refused to represent points. Based on time periods alone, Allen defined interval calculus [29] for temporal reasoning. Nevertheless, other researchers noticed that instants indeed appear in our common sense of temporal concepts. For instance, many situations (e.g., transitions between states) are instantaneous. Therefore, they proposed a time ontology made up of both time instants and time periods [28, 30]. However, this is not the end of the story because it leads to several semantic problems, such as Divided Instant Problem [31].

Apparently, time instants and intervals are the two core primitives for temporal reasoning no matter which is more in keeping with human's understanding of temporal concepts. In terms of subsymbolic temporal reasoning, the challenge of choosing instants or intervals mostly lies in computational effectiveness of subsequent subsymbolic representation and reasoning methods. Because subsymbolic representation methods are better at consuming discrete/categorical data instead of continuous data (e.g., intervals), dividing time into equal-distance instants seems to be more appropriate and natural. However, this assumes that time is a line. In order to capture other properties of time (such as continuity, periodicity, etc.), the objective for this research question comes down to develop a representation method for time instants in a subsymbolic way. Traffic prediction is an ideal research area for this purpose, since timestamp-ed traffic information is available and the temporal factor plays an important role for forecasting traffic speed. More specifically, in this research question, I will design a method to learn subsymbolic representation for each time instant.

Question 2: *How should we formalize temporal information in a subsymbolic reasoning system, namely how to establish the link between the time reference and atemporal statements?*

This research question focuses on how to introduce time into reasoning mechanisms. In terms of symbolic temporal reasoning systems, this relates to introducing time in logic as logic-based calculus are the basis for generic symbolic reasoning. Three mainstream approaches include temporal arguments [32, 33], modal temporal logics[34, 35] and reified temporal logics[36, 26, 37]. For instance, the well-known method - Temporal Arguments - introduced time as another parameter in first-order logic, and extended functions and predicates with temporal arguments to indicate the specific time when an argument is valid. One example would be *containsAdministrativeEntity(UK; Southern Ireland; [May 1921, Dec. 1922])*. Then the extended first-order logic and temporal ordering relations can be utilized to perform temporal reasoning.

In terms of subsymbolic temporal reasoning approaches, although logic and general reasoning theories of time are not necessary, the challenge remains similar, that is, how to marry time instants/time intervals with their atemporal assertions in the subsymbolic space. As discussed in Research Question 1, time instants seem to be a better ontological primitive for subsymbolic temporal reasoning. In addition, previous works on KGE indicate that generic subsymbolic reasoning is based on vector operations over numeric representations of entities and relations; see Section 1.3. Therefore, one naive way to introduce time into atemporal assertions for subsymbolic reasoning may follow two steps. First, I can utilize the representation method delivered by Research Question 1 to learn numeric representations of time instants. During the learning process, I ensure that numeric representations of time, entities and relations are in the same vector space, so that I can perform vector operations over them to achieve subsymbolic reasoning. Then, I need to design an association function to measure the compatibility of different quadruples (i.e., <*subject, predicate, object, time*>). Generally speaking, the association function should give more scores to quadruples presenting in KGs and versa vice. Following this proposal, the challenge is how to design an association function that can work for statements with different types of validity information. Although we leverage time instants as ontological primitives and learn subsymbolic representations for them, we still need to subsymbolically represent time intervals and measure the compatibility of such temporal statements. This is similar to symbolic temporal reasoning in the sense that we need to define intervals using the ontological primitive - instants [38, 39].

Question 3: *How can I perform subsymbolic temporal reasoning over qualitative temporal reasoning? And how is it compared to conventional symbolic reasoning?*

For this research question I shift my focus from quantitative temporal reasoning to qualitative temporal reasoning. I am interested in exploring how subsymbolic methods, in particular KGE methods, will perform in terms of qualitative reasoning. Unlike quantitative temporal reasoning, qualitative temporal reasoning focuses on qualitative relations between entities [29]. One example of such temporal relations could be <*Bronze age, follows, Iron age*>. Notice that such temporal statements have the exact same triple representation as generic statements do in KGs (i.e., <*subject, predicate, object*>). Therefore, technically speaking, any generic KGE method can be applied to perform qualitative temporal reasoning over temporal statements. However, generic KGE methods usually do not consider any prior knowledge of statements in KGs. As generic statements are mixed information from cross domains, there is no general prior/domain knowledge. However, time representation is a well-developed domain. A plethora of general theories of time and properties of temporal relations have been developed/discovered. First, the transitivity/composition table of temporal relations specifies chain rules for temporal reasoning. For example, if we know event A happened *during* event B, and event B occurred *after* event C, we can easily arrive at the conclusion that event A occurred *before* the happening of event C. Second, some temporal relations are transitive, have an inverse, or are symmetric; these properties also form the basis for temporal reasoning. For example, *before* and *after* are inverse relations. Thus, once we know event A happened before event B, we can easily know event B occurred after event A. Another example would be the relation *meet*, which is a symmetric relation. Thus if event A meets event B, then event B meets event A as well. Such domain knowledge has been widely used for symbolic temporal reasoning. I believe such prior knowledge could also be extremely useful for subsymbolic qualitative temporal reasoning. Thus, here my objective is to develop a domain-specific subsymbolic reasoning method for qualitative temporal reasoning. The challenge lies in how to make full use of such domain knowledge to guide KGE models to prioritize their reasoning process.

Question 4: *Why do subsymbolic reasoning methods perform better than conventional symbolic reasoning methods?*

Experiments conducted in Research Question 3 showed that subsymbolic reasoning methods performed better than conventional approaches by significant margins for both spatial and temporal reasoning. In this research question, I focus on understanding why this may be the case. Conventionally, symbolic qualitative (spatio-) temporal reasoning is built upon well-established theories or calculi, such as Composition tables (CT) and conceptual neighborhood structures (CNS) between temporal relations[40, 41, 42]. For example, Rina presented a unified method to temporal reasoning by executing temporal calculus over a temporal constraint network[43]. It seems to me that domain knowledge is the key to qualitative temporal reasoning. Thus, I assume if subsymbolic reasoning methods are capable of achieving good results for temporal reasoning, would it be possible that the subsymbolic models have learned some general theories of time from data implicitly? There are two challenges in testing my hypothesis. The first challenge is what domain knowledge could have been learned by the model. Since there are different general theories of time, it is hard to know what has been learned and where to start testing. A good starting point to test could be conceptual neighborhood of relations, since this theory is relatively easy. However, an in-depth analysis and understanding of symbolic methods is needed to justify which theory is likely to be learned by them. The second challenge is how to test whether a hypothetical theory has been learned, namely how to distill the hypothetical theory from the subsymbolic model itself.

1.5 Structure of the Dissertation

This chapter described the background of this dissertation and challenges that the subject - time - has posed to conventional symbolic temporal reasoning. Then I pointed out a potential direction that could address the aforementioned challenges. Next, I presented the general research question that this dissertation aims to answer, and raised four questions in Section 1.4 to answer it step by step. The following four articles provide solutions to each question sequentially, corresponding to Chapter 3, 4, 5 and 6 within this dissertation.

- Chapter 3: 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.

- Chapter 4: L. Cai, K. Janowicz, B. Yan, R. Zhu, and G. Mai, Time in a box: advancing knowledge graph completion with temporal scopes, in Proceedings of the 11th on Knowledge Capture Conference, pp. 121–128, 2021.

- Chapter 5: L. Cai, K. Janowicz, R. Zhu, G. Mai, B. Yan, and Z. Wang, Hyperquaternione: A hyperbolic embedding model for qualitative spatial and temporal reasoning, GeoInformatica (2022) 1–39.

- Chapter 6: L. Cai, K. Janowicz, and R. Zhu, Automatically discovering conceptual neighborhoods using machine learning methods, in 15th International Conference on Spatial Information Theory (COSIT 2022).

The rest of this dissertation is structured as follows.

Chapter 2 first defines important terms used in this dissertation, including knowledge graphs and temporal knowledge graphs. Then it provides necessary background about the general knowledge of time. Specifically, it introduces most of the fundamental problems in temporal reasoning, including how to decide on ontological primitives for time ontology, what kinds of time expressions we may encounter in our daily life, and how time is incorporated in traditional knowledge representation and reasoning (symbolic representation and reasoning). Then, it provides a concise summary of related works that apply subsymbolic methods (machine learning/ deep learning) to generic knowledge graphs. Finally, this chapter points to the potential of leveraging subsymbolic methods to develop a subsymbolic temporal reasoning system.

Chapter 3 presents a subsymbolic method for learning representations for the selected ontological primitive - time instants. I choose traffic prediction as my target application to test whether I could learn subsymbolic representations for time instants. Traffic condition at a place always changes from time to time, and thus traffic data usually come with timestamps. However, previous studies on traffic prediction usually treat traffic data as sequential/ordinal data (such as texts) while ignoring the associated timestamps (time instants). In this chapter, instead I preserve each timestamp with its corresponding traffic information and learn numeric representations for each timestamp. A representation function is used to ensure that the learned time representations preserve the continuity of time. Additionally, different encoding strategies are performed over time representations to capture the periodicity of traffic data. The learned representations of timestamps can be used to provide temporal signals for the downstream task - traffic prediction.

Chapter 4 presents a subsymbolic method that is able to perform temporal reasoning over different types of temporal statements present in KGs. Although Chapter 3 demonstrates the efficacy of subsymbolic methods in learning numeric representations for time instants, the problems of how to model time intervals and how to introduce time in atemporal statements remain. Since computers are better at consuming discrete/categorical inputs, I first discretize time intervals by using random sampling methods to pick up representative time instants, which are treated as the modeling result of time intervals in subsymbolic reasoning methods. Then I design an association function to measure the compatibility of temporal statements. Through training, time instants that are closer in the original temporal domain will still be closer in their vector space in terms of numeric representations. This chapter accomplishes a subsymbolic temporal reasoning system that contains all three core components as studied in conventional symbolic temporal reasoning system [24].

Chapter 5 presents a subsymbolic method that aims at performing qualitative temporal reasoning in contrast to quantitative temporal reasoning discussed in the previous two chapters. The method is designed to model various theories of time (such as composition tables) and properties of temporal relations (such as transitivity, asymmetricity v.s. symmetricity and inverse relations). Quaternions and calculus over quaternions are introduced such that the subsymbolic method is able to automatically discover and utilize chain rules in composition tables, inverse relations and symmetricity of relations. Besides, I leverage hyperbolic space rather than Euclidean space as the vector space such that the model is capable of modeling the transitive property of relations. Although the subsymbolic method is motivated by theories of time and properties of temporal relations, it can also be successfully applied to qualitative spatial reasoning. This is important as it shows that my work generalizes well. Last but not least, I introduce comparison experiments between symbolic reasoning methods (e.g., constraint network method) and subsymbolic reasoning methods.

Chapter 6 presents a graph-based method that produces a relation graph for investigating relationships between qualitative temporal relations discovered by subsymbolic methods. This method takes advantage of the reasoning result of subsymbolic methods and reconstructs the relationships between relations. A relation graph is produced, which is a directed weighted graph with nodes denoting different temporal relations and edges denoting the similarity between relations. This graph is well-aligned with conceptual neighborhood structures discovered by prior scholars in the literature. This chapter aims to explain why subsymbolic methods perform better than conventional symbolic methods, and will contribute to Explainable AI and Machine Learning-driven theory discovery more broadly.

This dissertation concludes in Chapter 7. There, I provide a summary of previous chapters and a discussion of the theoretical and practical contributions of my research. Furthermore, I conclude this dissertation by introducing limitations of this work and future research directions.

Chapter 2

Preliminary and Related Work

This chapter provides technical background knowledge and related works that are most related to this dissertation. First it introduces informal definitions of knowledge graphs and temporal knowledge graphs. Then it provides a review of three core components in symbolic temporal reasoning, including the discussion of ontological primitives for time, different time expressions used in common sense, and various ways of formalizing time. These three components lay the foundation for this dissertation and correspond to the first three research questions raised in Section 1.4 under the context of subsymbolic temporal reasoning. The third piece introduces related work on subsymbolic methods used in knowledge graph-based applications.

2.1 Knowledge Graph and Temporal Knowledge Graphs

This section explains what a knowledge graph is. Then it gives the definition of a temporal knowledge graph and points out its unique characteristics that differ from generic knowledge graphs.

Knowledge graphs (KGs) can be thought of as data repositories which store statements about the world around us. Most often, a statement consists of two entities and the relationship between them. Most KGs adopt the graph-structured data model - Resource Description Framework (RDF) to store such statements for computational and reasoning convenience. The data unit in RDF is a triple, consisting of three components corresponding to <*subject*, *predicate* and, *object*>. Subjects and objects are entities (objects can also be literals, such as the date of birth.) and the predicate describes the relation between entities. From a graph point of view, a KG is viewed as a directed labeled multi-relational graph. Each edge represents a statement, connecting a subject and an object with a relation as the label of the edge[44]. Despite the wide usage of this definition, a KG is beyond a data repository. It is made up of two parts: a knowledge base and a reasoner. The knowledge base refers to the data repository described above, which provides a way of explicitly and formally modeling the semantics of statements. In this sense, KGs facilitate information conceptualization, data accessibility, and semantic interoperability [3, 4, 5]. The reasoner emphasizes the ability of a KG in deriving/inferring new knowledge from existing information by using rules, ontologies or axioms. In this dissertation, I mostly focus on the knowledge base itself rather than the broader definition. KGs and KBs are used interchangeably through this dissertation. Note that the machine learning (ML) community also leverages $<h,\ r,\ t>$ to represent a statement, where h is the head (subject), r the relation (predicate), and t the tail (object), respectively. Given the introduction above, a definition of KGs can be formalized as below:

Definition 1 *A Knowledge Graph consists of a set of RDF triples $\mathcal{T} = \bigcup_{\langle h_i, r_i, t_i \rangle}$, where $\langle h_i, r_i, t_i \rangle$ denotes a unique RDF triple. Thus the KG can be denoted as $\mathcal{G} = \langle \mathcal{V}, \mathcal{R} \rangle$, where $\mathcal{V} = \{h_i | \langle h_i, r_i, t_i \rangle \in \mathcal{T}\} \cup \{t_i | \langle h_i, r_i, t_i \rangle \in \mathcal{T}\}$ denotes the set of nodes/entities, and $\mathcal{R} = \{r_i | \langle h_i, r_i, t_i \rangle \in \mathcal{T}\}$ denotes a set of relations.*

The definition of Temporal Knowledge Graphs (TKGs) is controversial, particularly in the ML community. Broadly speaking, a temporal knowledge graph is a knowledge graph that contains certain temporal information. Such temporal information could be quantitative temporal information, such as *the inception year of a country* presented in the object position (such as <*USA, inception, 1776*>), the validity period of a statement <*UK, isMemberOf, EU, [1973, 2020]*>, or qualitative temporal information, such as <*United Arab Republic, followedBy, Egypt*>. More details about different types of time expressions are summarized in Section 2.2.2. In this dissertation, I treat statements that contain any temporal information in any form (as shown above) as temporal statements. In this sense, almost all KGs are essentially temporal KGs. However, most works in the ML community define TKGs as a subgraph of generic KGs, and only include statements that have known validity periods into TKGs. That means they ignore the fact that some (temporal) statements may miss their validity periods due to lack of data. Such works, though, relieve the burden of addressing tricky challenges in TKGs by simplifying the definition itself, they are impractical and do not make advantages of atemporal statements and other types of temporal statements in KGs. In this dissertation, I adopt the broader definition of TKGs as it is more realistic, and study different types of temporal information in different chapters.

2.2 Time Primitives, Expressions and Formalization

The world is dynamic, in which activities and phenomena occur and change over time. Moreover, human usually perceive, interact and understand the real world through the lens of time. Therefore, the statements made by humans usually relate to time in a certain way: they are either time-aware events/states per se or attached to a transaction time (i.e., meta-data information indicates when a statement is made). This section introduces representational and formalizational issues of time when time is conceptualized for computational and reasoning purposes.

2.2.1 Time Primitives

A fundamental problem influencing representation of and reasoning about time is to determine the ontological primitives of time. In the study of temporal reasoning in AI, there are mainly three contenders, including time instants, periods (intervals) and both. The main motivation of thinking of instants as the primitive lies in its expressiveness of the continuity of time [24]. Based on this idea, time is modeled as an infinite line, consisting of individual points on the line. On the other hand, others believe that time periods are closer to common sense of temporal knowledge and should be the only primitive of time. Allen argued that points can be modeled as time intervals with a very short duration, and, thus are unnecessary to be regarded as primitives [29]. Additionally, he developed the well-known Interval Calculus as a formal theory of reasoning about time. Nevertheless, instants seem apparent in our common sense reasoning and are needed in many situations; for instances, transitions of states usually occur instantaneously. Other researchers proposed to acknowledge both time points and intervals as primitives and extended Allen's Interval Calculus to incorporate point-interval and interval-interval relations. However, incorporating both of them brings about some semantic difficulties, such as Divided Instant Problem [45]. Each contender comes with its pros and cons. In terms of subsymbolic temporal reasoning, I choose time instants as the ontological primitive, because machine learning methods, representation learning methods in particular, are better at digesting discrete/categorical inputs (here time instants).

2.2.2 Different Types of Time Expression

Differences in the conceptualization of time reflect the empirical usage, the internal representation and human's understanding of time. Roughly speaking, different expressions of time fall into two groups, corresponding to absolute and relational theories of time. In the absolute theory, time is a temporal reference frame defined precedently and independently of anything else. Upon it, events and activities can be said to happen at certain time points/periods. Everything is timestamped metrically. Quantitative time expressions (e.g., November 2021), also known as metric time, are used to express the temporal dimension of things. For instance, in a TKG, time intervals are used to indicate the validity period of statements (e.g., <*Poland, memberOf, Warsaw Pact, [1955, 1991]*>) and time instants represent the temporal information in a statement (e.g., <*Santa Barbara, inception, 1847*>). On the other hand, the relational theory claims that time exists only because events happening in the world are temporally related. Therefore, time can be defined by events and its properties are just reflections of the characteristics of events. Qualitative temporal relations have to be used to represent the relations among events. For example, we use predicates (e.g., follows, followedBy) to declare the temporal relation between archaeological ages (e.g., Bronze Age is followed by Stone Age). The predicate "hasPart" is used to describe the relation between the World War II and the Pacific War.

These different expressions of time are prevalent in TKGs like the examples shown above. In Chapter 3, Chapter 4 and Chapter 5, I primarily focus on how to model time instants, time intervals, qualitative temporal relations in a subsymbolic way, respectively.

2.2.3 Time Formalization

Reasoning mechanisms of time and the complexity of reasoning are largely determined by ways of introducing time to atemporal statements. Introducing qualitative temporal relations in statements is straightforward, since temporal relations can be just modeled as ordinary predicates. Then general theories of time (such as temporal calculus [46, 47, 48, 29] and temporal conceptual neighborhood structures [41, 49]) can serve as inference techniques to trigger temporal reasoning. However, incorporating quantitative time into atemporal statements is more challenging as it may cause computational and reasoning difficulties. There are five different ways to build the bridge between atemporal arguments and a temporal reference in the Semantic Web.

Standard Reification Reification provides a higher expressive power that permits one to express statement-level metadata (such as the validity of a statement) [50, 51, 52]. Relying on existing methods in Resource Descriptive Framework (RDF), standard reification introduces a blank node to notify an atemporal statement and attaches other metadata into the blank node. Take as an example the statement *<Poland, memberOf, Warsaw Pact, [1955, 1991]>*. It can be decomposed into an atemporal statement *<Poland, memberOf, Warsaw Pact>* and other statement-level statements *<statement1, hasTemporalScope, [1955, 1991]>*, *<statement1, subject, Poland>*, *<statement1, object, Warsaw Pact>* and *<statement1, predicate, memberOf>*, where *statement1* is uniquely determined by the atemporal statement.

N-ary Relations This approach introduces a higher-order data structure via a new relationship concept class connecting all components in a temporal argument [53, 54]. The example above can be expressed by four statements: (1) <*memberOfRelationship1, type, memberOfRelationship*> (2) <*memberOfRelationship1, partner1, Poland*> (3) <*memberOfRelationship1, partner2, Warsaw Pact*> (4) <*memberOfRelationship1, hasTemporalScope, [1955, 1991]*>.

Singleton Properties Naively, this approach adds a statement identifier to the predicate [55]. The example can be easily represented as two triples: (1) <*Poland, memberOf_1, Warsaw Pact*> (2) <*memberOf_1, hasTemporalScope, [1955, 1991]*>. Note that the 1 in *memberOf_1* is the unique identifier for the atemporal statement.

Named Graphs As the name indicates, it uses the so-called named graphs and is a variation of the singleton property [56, 57]. A fourth element is introduced to the triple *(*<subject, predicate, object>) to indicate that this statement is part of a concrete named graph (also a node in the RDF graph). Then temporal information can be attached to the named graph. The example above can be expressed as two triples: (1)<*Poland, memberOf, Warsaw Pact, statement_1*> (2)<*statement_1, hasTemporalScope, [1955, 1991]*>.

RDF* It is an extension of RDF standard that allows for triples to be directly used as the subject or object of other triples that represent their metadata. Namely, it introduces the notion of nested triples. It provides a more efficient way of reifying statements, thus being able to handle more statement-level information [58, 59]. From the perspective of query answering and data storage, it requires less storage and shorter queries. The example above can be represented by two triples: (1)<*Poland, memberOf, Warsaw Pact*> (2) < <*Poland, memberOf, Warsaw Pact*>, *hasTemporalScope, [1955, 1991]*>

These different ways of introducing time to atemporal statements have pros and cons. They may bring up different impacts on data exchange, query efficiency, data maintenance, reasoning, etc. In terms of subsymbolic temporal reasoning, I also need to think about how to establish the link between the time reference and atemporal statements, and how those different ways would impact the performance of the reasoning system. This corresponds to Research Question 2. However, unlike symbolic temporal reasoning, this way of introducing time is most concerned with its impact on temporal reasoning, since other parts will not cause much difficulty to subsymbolic methods. Chapter 4 provides an answer to this concern.

2.3 Knowledge Graph Embedding

Knowledge graphs are commonly incomplete, necessitating the task of knowledge graph completion (KGC), i.e., inferring missing knowledge based on existing statements in KGs. Reasoners relying on formal logic would easily fail when missing links emerge. Moreover, the symbolic and logical nature of knowledge representation in KGs restricts it from being applied to other applications that involve numeric computation, such as recommendation systems [11]. In order to address these issues, recently many studies have proposed knowledge graph embedding (KGE) methods, which fall into representation learning. These methods aim to learn numeric vector representations of entities and relations in a high-dimensional vector space while preserving the underlying semantic and structural information presented in KGs [12, 60, 61].

Different training objectives are developed to guide how to learn those numeric representations of entities and relations. One of the most classic assumptions used for training purpose is Translation [62, 63, 64, 11]. For a triple $<h, r, t>$ in a KG, Translation-based models first project h, t and r onto vectors $\mathbf{h}$, $\mathbf{t}$ and $\mathbf{r}$ in a high-dimensional vector space. Then it assumes $\mathbf{h}$ can be translated to $\mathbf{t}$ by the relation r. Thus, a training objective should be $||\mathbf{h}+\mathbf{r}-\mathbf{t}|| = 0$. This training objective is then applied to each statement presented in KGs. Finally, numeric representations of entities and relations will be learned after iterative optimization. Meanwhile, the learned representations contain the global information thanks to the linkages between entities. Besides the translation assumption, another transformation assumption is rotation. Sun et al. thought of a relation as a rotation from the subject to the object in the complex vector space and proposed RotatE, which was the first model that can deal with symmetry/anti-symmetry, inversion, and composition relations simultaneously [65].

Unlike the aforementioned two branches, another group of embedding methods is called semantic matching energy methods. They measure the existence of a statement as the compatibility of the subject, the object and the relation between in a latent vector space. For instance, DistMult [66] used a 3-way inner product as the scoring function. Other approaches along this line include RESCAL [67] and ComplEx [68].

The aforementioned methods all fall into triple-based methods, in which triples/ statements are regarded as atomic training samples. Instead, some other works incorporate higher-order information (e.g., multi-hop paths, neighbourhood structures) to training objectives such that more diverse information can be used for representation learning, such as PTransE [16], R-GCN [17], and Trans-GCN [69].

To summary, the goal of KGE methods is to learn representations for entities and relations in a high-dimensional space. Then downstream tasks boil down to performing vector operations over numeric representations of involved entities and/or relations. Despite pronounced breakthrough in time-agnostic knowledge graphs, these models neglect the fact that most statements are only true during a certain time period, thus being incapable of addressing temporal statements in TKGs. In this dissertation, I will take full advantage of representation learning methods, knowledge graph embedding methods in particular, to develop subsymbolic methods for temporal reasoning specifically.

2.4 Summary

This chapter presented essential background knowledge of this dissertation and reviewed related works most relevant to this dissertation. It first clarified the definition of knowledge graphs and temporal knowledge graphs, which points out the study scope of this dissertation. Then it provided a detailed introduction of three core components in symbolic temporal reasoning, which will be utilized as the guidance to the study of subsymbolic temporal reasoning. The connections between the three core components, the research questions raised in Section 1.4 and the contents of following Chapters were further clarified. In the end, the general idea of subsymbolic methods was described by using the success stories of KGE methods in reasoning over generic knowledge graphs.

Chapter 3

Traffic Transformer: Capturing the Continuity and Periodicity of Time Series

In this chapter, I focus on learning numeric representations for the chose ontological primitive - time instants. Specifically, I use Traffic Prediction as a case study since traffic data usually come with time information (in terms of time instants/timestamps) and time is an important factor for traffic prediction. The developed subsymbolic methods can preserve the continuity of time and the periodicity of time series. Experiments on real-life datasets show that explicitly considering the continuity and periodicity of time series can boost the prediction performance.

Peer Reviewed Publication	
Title	Traffic Transformer: Capturing the Continuity and Periodicity of Time Series for Traffic Forecasting
Author	Ling Cai, Krzysztof Janowicz, Gengchen Mai, Bo Yan, and Rui Zhu
Venue	Transactions in GIS (Volume 24, Issue 3)
Editors	John P. Wilson, Qiming Zhou, and Peter Mooney
Publisher	Wiley
Pages	736-755
Publication Date	11 June 2020
Copyright	Reprinted with permission from Wiley
DOI	https://doi.org/10.1111/tgis.12644

Abstract Traffic forecasting is a challenging problem due to the complexity of jointly modeling spatiotemporal dependencies at different scales. Recently, several hybrid deep learning models have been developed to capture such dependencies. These approaches typically utilize Convolutional Neural Networks (CNNs) or Graph Neural Networks (GNNs) to model spatial dependency and leverage Recurrent Neural Networks (RNNs) to learn temporal dependency. However, RNNs are only able to capture sequential information in the time series, while being incapable of modeling their periodicity e.g., weekly patterns. Moreover, RNNs are difficult to parallelize, making training and prediction less efficient. In this work, we propose a novel deep learning architecture called *Traffic Transformer* to capture the continuity and periodicity of time series and an additional GNN to model spatial dependency. Our work takes inspiration from Google's Transformer framework for machine translation. We conduct extensive experiments on two real-world traffic datasets and demonstrate that our model outperforms baseline models by a substantial margin.

3.1 Introduction

Traffic forecasting is concerned with estimating future traffic conditions, such as the density of vehicles and their speed, to enable the prediction of future events such as congestion or travel duration, more generally, by analyzing historical traffic conditions and patterns. Highly accurate forecasts provide guidance to decision makers, provide safety and convenience for citizens, and reduce environmental impacts.

Traffic forecasting, however, is challenging due to the complexity of modeling spatiotemporal dependencies of traffic conditions at varying scales [70, 71]. For instance, the traffic flow on a road is influenced by both its historical traffic conditions and the conditions of upstream roads. Due to the increasing availability of massive traffic data, high-performance computing, and novel deep learning models, recent work has pushed the envelop on learning spatiotemporal dependency models for accurate traffic forecasting [72, 70, 73, 74, 75].

RNN-based models, e.g. Gated Recurrent Unit (GRU) and Long Short Term Memory (LSTM), can be used effectively to capture temporal dependencies [74, 73]. For example, [76] proposed a convolutional LSTM model for traffic forecasting, in which the traffic flow at each time step was fed into an LSTM architecture recursively. Similarly, [77] proposed a convolutional RNN model, where graph diffusion convolutional operators were used to model spatial dependencies and GRU was employed instead of LSTM to capture temporal dependencies.

Although these RNN-based models can capture temporal sequential dependency, RNNs have several inherent deficiencies. First, they struggle to preserve very long-term sequential information, which leads to the loss of long-term temporal dependency in time series during the forward path [78]. Second, RNNs are unable to capture the periodicity of time series, since they treat different time steps equally in the time series. This is an important shortcoming as time series usually convey periodic patterns, e.g. hourly, daily, weekly, and seasonally [79, 75]. Third, RNNs are difficult to parallelize, making the training and prediction process less efficient.

Recently, researchers have introduced the Transformer architecture to replace RNNs for machine translation [80]. It replaces convolutional and recurrent neural networks and is solely built on attention mechanisms to model sequential data. Hence Transformer does not require sequential data to be fed recursively. This makes the architecture computationally more efficient than RNNs. More importantly, so as to preserve the order of elements in a sequence (e.g., the order of words in a sentence) when modeling sequences, Transformer introduces a position encoding strategy. It encodes positions of elements (e.g., words) in the sequence (e.g., a sentence) by first indexing them by their positions and then passing the indexes through a series of sinusoidal functions. Transformer and its variants have achieved significant success in natural language processing, including machine translation, text generation, and so on [81, 82, 83].

Interestingly machine translation and traffic prediction share some structural similarities. In machine translation [84], the aim is to translate a source sentence written in one language to a target sentence in another language by using a sequence-to-sequence learning framework [85], where the source and target sequences both consist of tokens. Likewise, traffic prediction can be formulated in a similar way. More specifically, the task is to utilize data about historical traffic conditions in such a way that they become indicative of future conditions, where the source sequence consists of a series of traffic data (e.g., traffic volume, speed) in the past and the target sequence is composed of a series of traffic conditions at future time steps. Put differently, each time step in the source and target traffic sequences amounts to the position index of each word in the input and output sentences in the machine translation task.

Analogies are partial by definition. Hence, Transformer cannot be directly applied to traffic forecasting due to the following reasons. First, the position encoding strategy is inapplicable. The semantics of the source sequence and the target sequence in machine translation and traffic forecasting are different. In machine translation, the source and target sequences represent two sentences with the same meaning in different languages; thereby the corresponding words in the two sequences should share similar position index. In contrast, in traffic forecasting, the source-target sequence is consecutive; hence there is no correspondence between elements in the source and target sequences. Instead, traffic forecasting takes into account *the continuity of time series* when indexing the traffic by their time steps. Additionally, traffic data are also characterized by several other properties of time, e.g., periodicity. For instance, the traffic condition on one road at Wednesday 3:00 pm is similar to its traffic condition on Thursday at the same time. *The periodic characteristics of traffic data* should also be considered when adapting Transformer to this domain. Consequently, this calls for new ways of encoding temporal features in the Transformer architecture. Second, Transformer is only able to handle the sequential dependency between the source and target sequences in machine translation, while spatial (network information) and temporal (sequential information) dependencies in traffic data are prominent [86, 87]. Hence, we need to enable Transformer to handle spatial and temporal dependencies coherently.

To solve these problems, we propose to design different strategies of encoding temporal information so that both the continuity and periodicity of traffic data can be preserved, and extend Transformer to modeling temporal dependencies and spatial dependencies jointly with the help of Graph Convolutional Networks (GCNs). **The main contributions of our research are as follows:**

- We design four novel position encoding strategies to encode the continuity and periodicity of time series to facilitate the modeling of temporal dependencies in traffic data. In total, we propose seven temporal encoding methods by combining different strategies.

- We introduce a hybrid encoder-decoder architecture, called *Traffic Transformer*, to coherently model spatial and temporal dependencies of traffic data in an end-to-end training manner, where Transformer is leveraged to model temporal dependencies and a GCN contributes to the modeling of spatial dependencies.
- Experimental results on two real-world benchmark datasets show the performance of our model compared to state-of-the-art methods, demonstrating the effectiveness of our temporal encoding methods and the hybrid architecture.

The rest of this paper is structured as follows. Section 3.2 reviews existing work on traffic forecasting. Section 3.3 defines the traffic forecasting task and introduces Transformer in a nutshell. Next, section 3.4 presents different strategies for encoding temporal characteristics and the proposed architecture for traffic forecasting. Section 3.5 explains our experiments and presents the results. Finally, section 3.6 concludes our work and points to directions for future research.

3.2 Related Work

Big data-driven machine learning models for traffic forecasting have attracted extensive attention from both academia and industries for several years [88, 89, 75, 83, 90]. As far as deep learning is concerned, [91] were among the first to applying deep learning models to forecasting traffic, by designing a deep belief network for unsupervised feature learning and then passing these features through a regression layer for traffic forecasting. Since then, most of deep learning models for traffic forecasting were built by utilizing RNN due to its capability to memorize temporal dependencies in time series via self-circulation. [92] compared different RNN models, namely LSTM and GRU, finding that GRU achieved better performance than LSTM in forecasting traffic. However, these aforementioned methods are limited to only capturing forward temporal dependencies. In contrast, [73] proposed a deeply stacked bidirectional and unidirectional LSTM architecture, which is able to capture both forward and backward dependencies in time series.

While these aforementioned models account for temporal dependencies in traffic data, spatial correlations were often neglected. To fill the gap, [89] proposed a novel deep learning architecture to inherently consider temporal and spatial dependencies, where autoencoders are first introduced to serve as the building block to learn latent features. [93] designed a deep learning model, which takes into account both temporal aspects – temporal closeness, periods, and trends of crowd traffic – and spatial proximity. More recently, CNNs and GNNs became the most widely used models in detecting patterns thanks to their progress in capturing spatial/topological dependencies in images, videos, and graphs [94, 95, 72, 70]. In addition to a GRU for capturing temporal features, [96] converted network-wide traffic matrices into images, after which a CNN was imposed to learn global spatial interactions of the converted images. [70] proposed a hybrid deep learning framework by marrying the CNN with LSTM architecture, where an 1-D CNN was utilized to model spatial dependencies while two LSTMs were exploited to learn temporal patterns. Considering that a graph is a more appropriate abstraction of a road network, [77] suggested to replace CNNs with GCNs in order to extract spatial dependencies, and proposed a new model, called DCRNN, in which spatial dependencies are modeled as a diffusion process. Similarly, [87] defined another graph convolutional operator, incorporating the adjacency matrix and a free-flow reachable matrix to capture localized spatial features. More generally the future seems to lie in combining CNNs/GCNs with RNNs to extract both spatial and temporal dependencies for traffic forecasting. However, one limitation of such architectures stems from RNNs themselves, which are well-known for their high computational cost during the training process.

To circumvent the inherent deficiencies of RNNs, research started to investigate non-recurrent models. For example, [97] proposed a universal framework that completely consists of spatiotemporal convolutional blocks. Experiments showed that this model yielded better results than the model proposed by Li et al. [77]. Guo et al. [79] designed a spatiotemporal attention mechanism as well as a spatiotemporal convolutional module to capture spatiotemporal dependencies of traffic data. However, this approach solely relied on the temporal attention mechanism to assign different importance to traffic in the past and by doing so it ignored the fact that the most recent traffic condition should have a greater influence on predicting current traffic.

Recently, Transformer [80] has been developed as a new architecture in deep learning, which employs attention mechanisms along with a position encoding strategy for sequence modeling. In light of this, several attempts have been made to tailor Transformer towards time series forecasting [98, 99]. For example, [98] argued that the basic Transformer architecture is not sensitive to local contexts, and suggested to add a convolutional self-attention layer to improve it. Although this work has been demonstrated effective in capturing long-term temporal dependencies, it is insufficient for network-wide traffic forecasting. First, it ignored spatial dependencies on a road network and thus failed to model spatiotemporal correlations. Second, it captured periodical patterns at the cost of feeding very long sequences (e.g. 768) into models, which may be impossible for network-wide traffic forecasting. The amount of traffic data in a network is far larger than that at a single spot as there are hundreds of sensors on a road network reporting the traffic frequently. Very long sequences will, hence, consume too much memory. As a result, the question of how to preserve periodical patterns when using Transformer for network-wide traffic forecasting remains under-explored. In our paper, we focus on explicitly modeling spatial and temporal dependencies all together – the spatial correlations of the traffic in a network as well as the continuity and periodicity of time series.

3.3 Preliminary

3.3.1 The Network-wide Traffic Forecasting Task

The goal of traffic forecasting is to predict the future traffic given a sequence of historical traffic observations (here speed, density, and volume) that are detected by sensors on a road network. Such a sensor network deployed to monitor roads is usually represented as a weighted directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$, where $\mathcal{V}$ is a set of sensors with $|\mathcal{V}| = N$, $\mathcal{E}$ is a set of edges, connecting sensors, and $\mathbf{W} \in \mathbb{R}^{N \times N}$ is the adjacency matrix storing the distance between sensors in the network. $\mathbf{X}^t \in \mathbb{R}^{N \times P}$ denotes the feature matrix of the graph that is observed at time t, where P is the number of features. The prediction problem can then be formalized as learning a mapping function F from M previously observed feature matrices to H future feature matrices on the premise of a network $\mathcal{G}$:

$$\mathcal{X}_{t+1}^{t+H} = F\big(\mathcal{G}; \mathcal{X}_{t-(M-1)}^{t}\big) \tag{3.1}$$

where $\mathcal{X}_i^{i+n}$ denotes an array of feature matrices from time stamp i to $i+n$ and $\mathcal{X}_i^{i+n} = [\mathbf{X}^i, \mathbf{X}^{i+1}, \ldots, \mathbf{X}^{i+n}]$.

3.3.2 Transformer in Machine Translation

Unlike RNNs, Transformer belongs to the family of non-recurrent neural networks. It is solely built upon attention mechanisms, which makes it possible to access any part of a sequence regardless of its distance to the target [80, 98].

In essence, Transformer is organized in an encoder-decoder manner, in which identical encoder modules are stacked at the bottom of stacked decoder modules. Each encoder module is composed of a multi-head self-attention layer and a position-wise feed forward layer, while each decoder module has one more layer, namely encoder-decoder attention layer, which is inserted between the self-attention layer and feed forward layer to bridge the encoder part and decoder part.

The aim of multi-head attention layers is to attach different importance of words/tokens to each other in a sequence from multiple aspects (heads). Then the outputs of those different heads are concatenated and then passed through a linear transformation to aggregate all the information. As the attention mechanisms behind the self-attention layer and the encoder-decoder attention layer are the same, we take the self-attention layer as an example. It operates on a sequence of tokens $x = (x_1, x_2, ..., x_n)$, each of which is initialized by a random vector, and is updated by using a weighted sum of any other word after being passed through a linear transformation. The weights, called attention scores, are assigned by their similarities.

Take the update of $\mathbf{x}_i$ as an example:

$$\mathbf{y}_i = \sum_{j=1}^{n} a_{ij}(\mathbf{x}_j \mathbf{W}_V) \tag{3.2}$$

where $\mathbf{y}_i$ is the updated $\mathbf{x}_i$, and a_{ij} is the attention score, measuring the similarity between $\mathbf{x}_i$ and $\mathbf{x}_j$, calculated by Eq. 3.3.

$$a_{ij} = \frac{\exp(e_{ij})}{\sum\limits_{k=1}^{n} \exp(e_{ik})} \tag{3.3}$$

where e_{ij} measures the compatibility of two linearly transformed $\mathbf{x}_i$ and $\mathbf{x}_j$, calculated by using the scaled dot product:

$$e_{ij} = \frac{(\mathbf{x}_i \mathbf{W}_Q)(\mathbf{x}_j \mathbf{W}_K)^T}{\sqrt{h}} \tag{3.4}$$

where h is the dimension of the output. Note that $\mathbf{W}_V$, $\mathbf{W}_Q$, $\mathbf{W}_K$ are three linear transformation matrices to strengthen the expressiveness of Transformer .

More importantly, since Transformer does not involve any convolutional and recurrent module, position embeddings are designed to preserve the sequential order in a sequence. The positions of words in a sentence are first indexed starting from 0 to the length of the sentence and then position indexes are fed into Eq. 3.5 to gain position embeddings for each word. The corresponding words in the source sentence and the target sentence share the same position embedding. The encoding strategy guarantees that the position embedding at $pos + k$ is a linear function of that at pos. Then an element-wise additive operator is imposed on the position embeddings and the initialized vectors of their corresponding word. The result is fed into encoder/decoder modules subsequently.

$$\begin{aligned} pos\_embedding(pos, 2i) &= sin(pos/10000^{\frac{2i}{h}}) \\ pos\_embedding(pos, 2i+1) &= cos(pos/10000^{\frac{2i}{h}}) \end{aligned} \tag{3.5}$$

where pos is the position index of a word in a sequence, i is the i-th dimension of the position embedding and h is the dimension of hidden layers. We denote the position embedding of a word at pos as $pos\_embedding_{pos} \in \mathbb{R}^h$.

Although this encoding strategy is useful for machine translation, it is inappropriate for traffic forecasting as temporal features such as continuity and periodicity have to be carefully incorporated.

3.4 Proposed Architecture

In this section, we introduce four strategies of encoding temporal information, more specifically the *continuity* and *periodicity* of time series. Next, we introduce two graph convolutional filters which can help capture spatial dependencies. Finally, we illustrate our proposed hybrid deep learning architecture.

3.4.1 Transformer for Capturing Temporal Dependencies

For ease of expression, we assume the sampling frequency of the time series is s times per day. We use $[t-(M-1), t-(M-2), ..., t]$ and $[t+1, t+2, ..., t+H]$ to denote the time steps of the source sequence (from the past) and that of the target sequence (in the future).

The Continuity of Time Series

Relative Position Encoding This strategy aims at encoding relative continuity, which means that we care about the continuity of time in the window of the source-target sequence regardless of the *position* of one time step in the whole time series under consideration. This can be achieved by indexing the time step $(t-(M-1))$ with 0 as the starting position and raising the index position by one per time step. These position indexes are simply passed through Eq. 3.5 to get position embeddings for each time step. By doing so, the continuity of time within the source-target sequence pair is encoded. All concatenated source-target sequences in the traffic prediction task share the same position embeddings.

Global Position Encoding Relative position encoding only preserves the local continuity of time within $H+M-1$ time steps. However, we may notice that most time steps in two consecutive source-target sequence pairs are common. For example, the previous sequence pair may range from 3:00 pm to 5:00 pm with 5 minutes as the sampling time interval, while the subsequent sequence pair ranges from 3:05 pm to 5:05 pm. So the potential limitation of the relative position encoding strategy is that the same time step is assigned with a different position embedding depending on its position in a sequence pair. Hence, we additionally propose a global/absolute position encoding strategy so that a time step occurring in the whole time period in question has only one position embedding even if it appears in different sequences. All the time steps in question are sorted by time first and then are indexed starting from 0. Same as for the relative position encoding strategy, position embeddings for each time step are obtained by passing the position index into Eq. 3.5. Hence, we assume that both the local and global continuity of time series are preserved.

The Periodicity of Time Series

Aside from the continuity, time series also conveys periodicity, namely weekly patterns and daily patterns. There are two potential ways to go about this. One is centered around the position encoding design: the position embedding for each time step is enriched with periodic patterns. The other one is by using different time series segments corresponding to different temporal features.

Periodic Position Encoding Based on any position encoding strategy in subsection 3.4.1, here we enrich position embeddings with periodic patterns. The idea is to design another position encoding strategy to cover weekly-periodic and daily-periodic information, which implies applying the relative/global position encoding strategy to position indexes in terms of weeks and days, respectively. For a single day, the sampling times per day are regarded as the number of positions for daily-periodic position encoding. For instance, if the sampling time interval is 5 minutes, then the sampling times per day are 288 (=60*24/5), namely the total number of positions we need to encode for a daily-periodic pattern. The derived position embedding is called *daily-periodic position embedding.* On the other hand, weekly-periodic patterns imply that traffic on the same days of the week are more similar as compared to other days. For instance, usually the traffic pattern on a future Friday should be more similar to those from past Fridays as compared to, say, Wednesdays. In this case, only seven positions are used that correspond to different week days. We call the resulting kind of position embeddings *weekly-periodic position embedding.* We impose element-wise addition over relative/global position embedding, daily-periodic position embedding and weekly-periodic position embedding to gain a hybrid position embedding for each time step. Note that all embeddings are members of the same vector space in $\mathbb{R}^h$, with h being the number of dimensions.

Time Series Segments In essence, periodic position encoding employs different flags (e.g., flags to differentiate different time steps on one day and flags to denote days with distinct week attributes) to explicitly differentiate sequences in three aspects. Although it seems compelling to solely rely on the hybrid position embedding, this may hide other commonalities among sequences. This would make training models more difficult, especially when a training dataset is small. Hence, one may enrich the existing time series segment by concatenating two more intercepted time series segments - a daily component and weekly component, along the time axis as the input for modeling as Guo et al. [79] did and then to encode positions in this hybrid segment jointly.

For daily-periodic segment, denoted as $\mathcal{X}_{t+1}^{t+H}(D)$, we intercept segments at the same time period as the predicting time period on last few days, say d days. The segment can be formulated as Eq. 3.6.

$$\mathcal{X}_{t+1}^{t+H}(D=d) = [\mathcal{X}_{t+1-s*d}^{t+H-s*d}, \mathcal{X}_{t+1-s*(d-1)}^{t+H-s*(d-1)}, ..., \mathcal{X}_{t+1-s}^{t+H-s}] \tag{3.6}$$

The weekly-periodic segment, denoted as $\mathcal{X}_{t+1}^{t+H}(W)$, consists of the time series segment at the same time period as the predicting period on the days with the same week attribute in the past few weeks, say w weeks. This segment can be written as follows:

$$\mathcal{X}_{t+1}^{t+H}(W=w) = [\mathcal{X}_{t+1-s*7*w}^{t+H-s*7*w}, \mathcal{X}_{t+1-s*7*(w-1)}^{t+H-s*7*(w-1)}, ..., \mathcal{X}_{t+1-s*7}^{t+H-s*7}] \tag{3.7}$$

Finally, the hybrid segment, enriched by the recent, daily-periodic and weekly-periodic information is composed of $[\mathcal{X}_{t+1}^{t+H}(W), \mathcal{X}_{t+1}^{t+H}(D), \mathcal{X}_{t-(M-1)}^{t}]$ as the input to replace $\mathcal{X}_{t-(M-1)}^{t}$ in Eq. 3.1.

Note that although we concatenate these three segments along the time axis, it is impossible for Transformer to figure out the order information as it is completely attention-based. Therefore a position encoding strategy is also required. Since the time periods of a daily-periodic and week-periodic sub-segment on one day are the same as the forecasting period, the time steps in a daily-periodic/week-periodic sub-segment share the same position embeddings as that of the forecasting segment. Here we do not explicitly take into account the different contributions of daily-periodic/week-periodic segments to the prediction and leave it to the model to determine.

A Summary of Encoding Methods

In total, there are seven different encoding methods for capturing the continuity and the periodicity of time series by combining the aforementioned strategies. A summary can be found in Table 3.1. To provide a concrete example, given the traffic at *8:00am - 8:55am on 01/01/2020 (Wed.)*, we want to predict the future traffic at *9:00am - 9:55am on 01/01/2020 (Wed.)* in the table below.

Table 3.1: A summary of encoding methods

Embedding Method	**Relative Position Embedding**	**Global Position Embedding**	**Relative and Periodic Embedding**	**Global and Periodic Embedding**	**Time Series Segment Method**
Encoding Strategy	Relative Position Encoding	Global Position Encoding	Relative Position Encoding and Periodic Position Encoding	Global Position Encoding and Periodic Position Encoding	Time Series Segments
Abbreviation	RPE	GPE	RPPE	GPPE	TSE
Captured Features	relative continuity of time series	global continuity of time series	relative continuity and periodicity of time series	global continuity and periodicity of time series	relative continuity and periodicity of time series
Example (position index)	[0, 1, ... , 11] → [12, 13, ... , 23]	[99, 100, ... , 110] → [111, 112, ... , 122]	[0, 1, ... , 11] * [3, 3, ..., 3] * [97, 98,..., 108] → [12, 13, ... , 23]	[99, 100, ... , 110] * [3, 3, ..., 3] * [97, 98, ..., 108] → [111, 112, ... , 122]	[12, 13, ... , 23, 12, 13, ... , 23, 0, 1, ... , 11] → [12, 13, ... , 23]
Notes		*Assume among the whole time steps in question, 01/01/2020 Wed. 8:00am is indexed as 99.*	"*" here means we consider these three kinds of position indexes - relative, weekly, daily.	See the previous.	Here, d=w=1.

After obtaining these position indexes from different encoding methods, they are used in Eq. 3.5 to derive position embeddings for each time step in the source-target sequence. Similar to the original Transformer, one option of incorporating position embeddings into time series is by element-wise addition between the traffic features and its position embeddings. Although it is easy to implement, it is hard to interpret such combination since the vector spaces of the traffic features and the position embeddings would not be the same. We call this approach an *addition-based combination*. However, we suggest another approach here, namely a *similarity-based combination*. We adjust the attention score a_{ij} in Eq. 3.2 by using the similarity between two time steps in terms of position embeddings. In this way, the similarity between two time steps serves as a decay factor. That is to say, when two time steps are adjacent, then their similarity is high. So in the traffic prediction task, we can rewrite Eq. 3.2 and Eq. 3.3 as,

$$\mathbf{Y}^i = \sum_{j=1}^{L} a'_{ij}(\mathbf{X}^j W_V) \tag{3.8}$$

where a'_{ij} is the adjusted attention score. Note that in the traffic prediction scenario $\mathbf{X}^i \in \mathbb{R}^{N \times h}$ is the traffic at time i and L is the length of a sequence in question.

$$a'_{ij} = \frac{\exp(e'_{ij})}{\sum\limits_{k=1}^{L} \exp(e'_{ik})} \tag{3.9}$$

where e'_{ij} is the adjusted compatibility of two linearly transformed $\mathbf{X}^i$ and $\mathbf{X}^j$ with the similarity between corresponding temporal features being considered.

$$e'_{ij} = b_{ij} e_{ij} \tag{3.10}$$

$$b_{ij} = \frac{\exp(d_{ij})}{\sum\limits_{k=1}^{L} \exp(d_{ik})} \tag{3.11}$$

$$d_{ij} = pos\_embedding_i \times (pos\_embedding_j)^T \tag{3.12}$$

where d_{ij} is the similarity between two position embeddings of $\mathbf{X}^i$ and $\mathbf{X}^j$ and e_{ij} is defined in Eq. 3.4.

3.4.2 Graph Convolutional Neural Networks for Capturing Spatial Dependency

Another important dependency we need to address is spatial dependency. Simply put, we need to account for the fact that the change in traffic on one road is influenced by the traffic on its upstream roads. Since the traffic network is modeled as a graph and the traffic observations (e.g. speed, volume, etc.) can be considered as features of that road (at some time), we take advantage of GNNs to perform the convolution operation over graph-structured data to capture topological properties such as adjacency.

In general, a GNN built on the spectral graph theory is a generalization of a traditional convolutional neural network. In spectral graph theory, a graph is usually represented by its Laplacian matrix. The topological features of a graph can be obtained by analyzing the Laplacian matrix. Formally, graph spectral convolution can be defined as the multiplication of a signal $\mathbf{X} \in \mathbb{R}^N$ with a kernel g_θ, and written as,

$$g_\theta *_\mathcal{G} \mathbf{X} = \mathbf{U} g_\theta(\mathbf{\Lambda}) \mathbf{U}^T \mathbf{X} \tag{3.13}$$

where $*_\mathcal{G}$ is the notion of graph convolutional operator, $\mathbf{U} \in \mathbb{R}^{N\times N}$ is the matrix of eigenvectors decomposed from the normalized graph Laplacian $\mathbf{L} = \mathbf{I}_N - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}} = \mathbf{U}\Lambda\mathbf{U}^T \in \mathbb{R}^{N\times N}$, where $\mathbf{I}_N$ is an identity matrix, $\mathbf{D} \in \mathbb{R}^{N\times N}$ is the diagonal degree matrix and $\mathbf{\Lambda}$ is a diagonal matrix of its eigenvalues.

Since it is computationally expensive to directly decompose the Laplacian matrix, especially when a graph is large, several approximation approaches have been proposed. Two kinds of graph convolutional operators based on different approximation strategies have been used for traffic forecasting.

The most popular approximation was proposed by Kipf et al. [100]. Basically they followed the idea of Hammond et al. [101] that g_θ can be well-approximated by a truncated expansion of Chebyshev polynomials. But they only adopted the 1^{st}-order polynomials as the graph convolutional filter, since it is more computationally efficient. This setting can be achieved by a single neural layer with the 1^{st}-order neighbors being considered. In order to allow for multi-hop neighbors, they proposed to stack multiple such neural layers. Consequently, the structural neighborhood information on graphs can be incorporated by a deep neural network architecture without explicitly parameterizing polynomials. Specifically, Eq. 3.13 can be simplified as,

$$g_\theta *_\mathcal{G} \mathbf{X} = \theta_0 \mathbf{X} - \theta_1 \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}\mathbf{X} \tag{3.14}$$

with two shared parameters θ_0 and θ_1. To reduce the number of parameters in practice, θ is used to replace θ_0 and θ_1 with $\theta = \theta_0 = -\theta_1$. Furthermore, the Eq. 3.14 can be expressed as,

$$g_\theta *_\mathcal{G} \mathbf{X} = \theta(\mathbf{I}_N + \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}})\mathbf{X} = \theta(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}})\mathbf{X} \tag{3.15}$$

where $\hat{A} = \mathbf{A} + \mathbf{I}_N$ and $\hat{D} = \sum_j \hat{A}_{ij}$ are the renormalized matrices of $\mathbf{A}$ and $\mathbf{D}$ to deal with exploding/vanishing gradient problems.

The second graph convolutional filter also belongs to spectral convolutional operators, but this one is derived by modeling the traffic flow as a diffusion process, which is characterized by Markov process [95]. The assumption is that after several time steps, the diffusion process would stop and converge to a stationary distribution $\mathcal{P} \in \mathbb{R}^{N \times N}$. A K-step truncated stationary distribution $\mathcal{P}$ is used to characterize the transition probabilities between nodes. Additionally, bidirectional diffusion process is included in this model such that the model can flexibly capture the impact of the traffic on both upstream and downstream roads. The resulting diffusion convolutional operation over graphs can be formulated as:

$$g_\theta *_{\mathcal{G}} \mathbf{X} = \sum_{k=0}^{K-1} (\theta_{k,1}(\mathbf{D}_{out}^{-1}\mathbf{W})^k + \theta_{k,2}(\mathbf{D}_{in}^{-1}\mathbf{W}^{\mathbf{T}})^k)\mathbf{X} \tag{3.16}$$

where $\theta \in \mathbb{R}^{K \times 2}$ are the parameters of the bidirectional filter g_θ, and $\mathbf{D}_{out}^{-1}\mathbf{W}$ and $\mathbf{D}_{in}^{-1}\mathbf{W}^{\mathbf{T}}$ are the state transition matrices of the diffusion process.

There are some similarities and differences between these two kinds of graph convolutions. Both GCN and DCN are designed from a spectral perspective and operate over non-Euclidean data structure, namely graphs. The difference is that GCN is defined on undirected graphs while DCN can be applied both in directed and undirected graphs. By introducing a similarity transformation, GCN can be considered as a special case of DCN.

In our proposed architecture, we can use either approach to capture spatial interactions over graphs, with the traffic condition $\mathbf{X}_{t-(M-1)}^{t}$ as the input. We want to investigate which model would work better with Transformer to model spatiotemporal dependencies.

3.4.3 Traffic Transformer Architecture

Since Transformer itself can only deal with sequential information, it is unable to deal with complicated spatiotemporal dependencies. In order to address this problem, here we extend Transformer to ***Traffic Transformer*** by introducing a GNN. As illustrated in Figure 3.1, our proposed architecture conforms to an end-to-end sequence framework, composed of an encoder and a decoder.

In the encoder, a sequence of traffic $\mathcal{X}_{t-(M-1)}^{t}$ is first passed through a GCN layers by using Eq. 3.15/3.16 to aggregate the neighborhood information on a road network, which captures the spatial dependency over nearby nodes. Then a fully-connected neural network is employed to strengthen the expressiveness of the model. These features then are fed into the Transformer encoder cell to learn temporal features. A typical transformer encoder cell is depicted in 3.2

As for the decoder, it has a similar structure but has one more dense layer in the end. During the training process, a sequence of traffic $\mathcal{X}_{t}^{t+H-1}$ is fed into the decoder through the same series of neural networks as the input for the encoder and then is passed through the additional dense layer to map the outputs of the Transformer decoder cell to the predicted traffic $\mathcal{X}_{t+1}^{t+H}$. Unlike other recurrent models where the traffic data at different time steps are fed recursively, the input sequence for the encoder/decoder can be sent into our architecture simultaneously. Thereby, the architecture is more computationally efficient during the training process. However, in the prediction phase, this decoder functions a little differently as any other model for time series prediction does, because in practice the input sequence for the decoder is unknown. In order to yield the predicted traffic at future time steps, we use $\mathbf{X}^{t}$ as the first input for the decoder to output the predicted traffic $\mathbf{X}^{t+1}$, which then serves as the next input for the decoder to gain $\mathbf{X}^{t+2}$. This whole process continues until the predicted horizon meets. Note that here $\mathbf{X}^{t}$ rather than a zero matrix, which is widely adopted by other models, is used as the first input for the decoder, since it is supposed to provide more useful information than zeros.

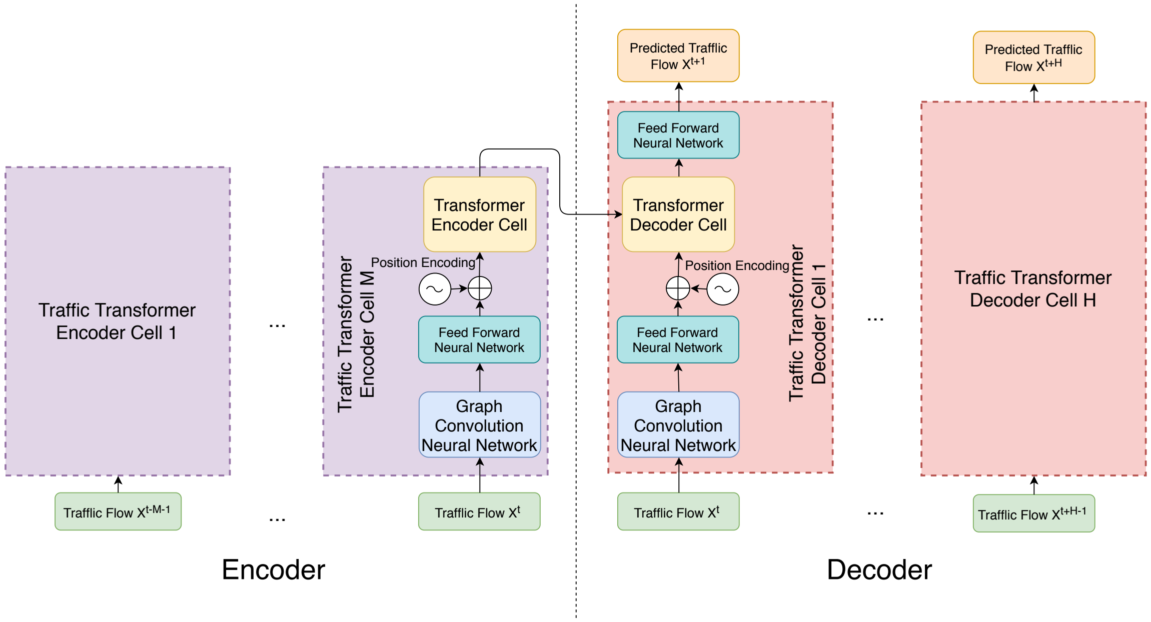

Traffic Transformer Encoder Cell 1
...
Traffic Transformer Encoder Cell M
Transformer Encoder Cell
Position Encoding
Feed Forward Neural Network
Graph Convolution Neural Network
Trafflic Flow X^{t-M-1}
Trafflic Flow X^t
Predicted Trafflic Flow X^{t+1}
Feed Forward Neural Network
Transformer Decoder Cell
Position Encoding
Feed Forward Neural Network
Graph Convolution Neural Network
Traffic Transformer Decoder Cell 1
Trafflic Flow X^t
Predicted Trafflic Flow X^{t+H}
Traffic Transformer Decoder Cell H
Trafflic Flow X^{t+H-1}
Encoder
Decoder

Figure 3.1: The architecture of Traffic Transformer

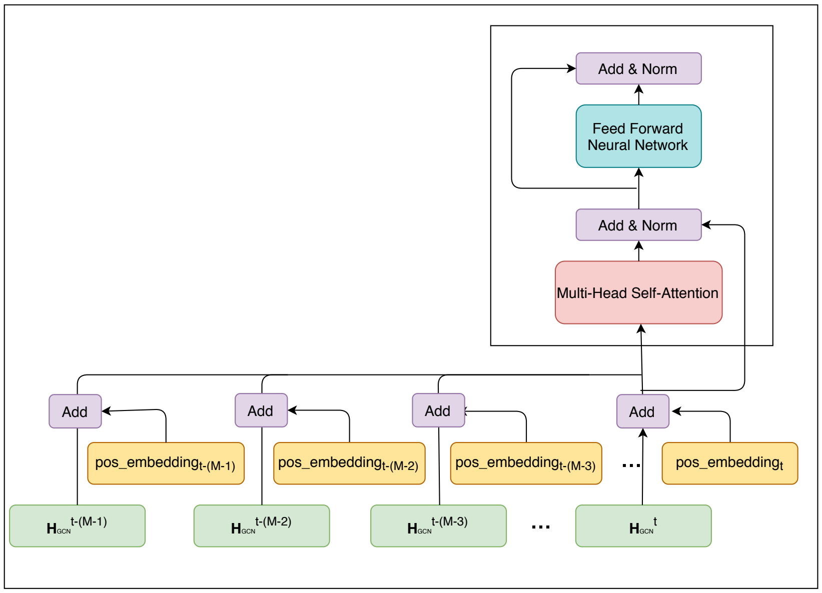


Figure 3.2: The diagram of Transformer encoder cell

During the training phase, the goal of our model is to minimize the difference between the real traffic and the predicted at each future time step, measured by Mean Absolute Error (MAE). The loss function of Traffic Transformer can be formulated as,

$$Loss = \frac{1}{H}\sum_{t=1}^{H}\frac{1}{N}\sum_{j=1}^{N}|\mathbf{X}_j^t - \hat{X}_j^t| \tag{3.17}$$

where $\mathbf{X}_j^t$ is the predicted traffic expected to be observed at sensor j at time t, and $\hat{X}_j^t$ is the corresponding ground truth.

3.5 Experiments

In this section, we report on the experiments we carried out to evaluate the performance of the proposed architecture.

3.5.1 Dataset Description

Two real-world benchmark datasets are used for evaluation: ***METR-LA*** and ***PEMS-BAY***. Those two datasets were collected and aggregated in a 5-min window by [77] from loop detectors in the highway of Los Angeles County and California Transportation Agencies Performance Measurement System (PeMS), respectively. Both data are sorted by time in an ascending order (from the past to the present) and are split into three parts for training (70%), validation (10%), and testing (20%). Z-score normalization with the mean and standard derivation of training data is applied to these three sets. To keep comparable with baselines, we use the sensor graphs of both datasets constructed by [77]. Table 3.2 shows the basic descriptions.

Table 3.2: Basic description of two datasets

Dataset	METR-LA	PEMS-BAY
#sensors	207	325
region	Los Angeles County	Bay Area
time periods	Mar.1st-Jun.30th, 2012	Jan.1st-May 31st, 2017
#training_pairs	23974	36465
#validation_pairs	3425	5209
#testing_pairs	6850	10419

3.5.2 Experimental Details

Baselines

We compare our proposed Traffic Transformer with multiple baselines:

- Historical Average model - *HA* which studies the seasonal trend of the traffic flow and then calculate the weighted average of seasonal traffic flow as the predictions Liu et al. [102].
- Auto-regressive Integrated Moving Average model with Kalman filter - $ARIMA\_kal$, which is a typical parametric model in time series community [103].
- Linear Support Vector Regression model - $LSVR$ [104] which employs linear support vector machine to learn relationships between the input time series and the output time series from historical traffic flow and then predict the future traffic.
- Feed-Forward Neural Network - *FNN*, composed of two dense layers with $L2$ normalization.
- Fully Connected LSTM - *FC-LSTM*, an RNN-based sequence model. This architecture conforms to an encoder-decoder framework composed of two LSTM layers in both encoder and decoder side [85].

- Diffusion Convolutional Recurrent Neural Network - *DCRNN*, proposed by Li et al. [77]. As the name has indicated, it designs a diffusion convolutional operator by assuming the traffic follows a diffusion process, and then this graph convolutional operator is fused with GRU to capture spatiotemporal dependencies.

- Spatiotemporal Graph Convolutional Networks - *ST-GCN* [72], which is completely built on spatial and temporal convolutional structures.

For more implementation details of these baselines, please refer to the work by [77]. In order to keep a fair comparison, we do not run most of the baselines but directly take the results from the DCRNN paper. For *ST-GCN* model, we use open-sourced codes provided by [72] to train the models with the datasets we use in this paper.

Implementation Details

To test the performance of different encoding methods, we set the length of a sequence in history – M and the length of the forecasting sequence – H as 12. For the Time Series Segment Method, the number of days for capturing weekly-periodic patterns – w and daily-periodic patterns – d are 1. The hidden dimension is 64 for all the layers in our model. Two encoder and decoder cells are used and one graph convolutional layer is utilized. We use open source code to implement this whole architecture. [1] Additionally, as stated in subsection 3.4.3, the forecasting sequence is unknown at prediction while known at training. As a result, there is a discrepancy between these two process, which causes a quick error accumulation along the yielded sequence [105]. In order to solve this problem, we also adopted scheduled sampling in our model to bridge the difference. Please refer to [105] for more details.

[1] `https://github.com/tensorflow/models/tree/master/official/transformer`

Evaluation Metrics

In order to evaluate and compare the performance of different models, we adopt the following three metrics.

Mean Absolute Errors(MAE): the same as Eq. 6.1.

Root Mean Squared Errors (RMSE):

$$RMSE = \sqrt{\frac{1}{H}\sum_{t=1}^{H}\frac{1}{N}\sum_{j=1}^{N}(\mathbf{X}_j^t - \hat{X}_j^t)^2} \tag{3.18}$$

Mean Absolute Percentage Errors (MAPE):

$$MAPE = \frac{1}{H}\sum_{t=1}^{H}\frac{1}{N}\sum_{j=1}^{N} \mid \frac{\mathbf{X}_j^t - \hat{X}_j^t}{\mathbf{X}_j^t} \mid \tag{3.19}$$

The first two metrics measure absolute prediction errors, while the last metric measures relative prediction errors. For all these metrics, smaller values mean better prediction performance. On both datasets, we exclude missing data when evaluating the performance of models.

3.5.3 Experimental Results

Comparison of Traffic Prediction Performance

Table 3.3 shows the prediction performance of different methods in terms of the 15-min, 30-min and 60-min ahead prediction. Obviously, there are several interesting discoveries: (1) In general, neural network-based models, including FNN, FC-LSTM, ST-GCN, DCRNN, and our model noticeably outperform these weak baselines which typically only focus on modeling temporal features. This indicates that these simple

Table 3.3: Prediction performance of different models on two datasets

					METR-LA				
T	Metric	HA	ARIMAKal	SVR	FNN	FC-LSTM	ST-GCN	DCRNN	Our model
15 mins	MAE	4.16	3.99	3.99	3.99	3.44	3.8	2.77	**2.43**
	RMSE	7.8	8.21	8.45	7.94	6.3	7.9	5.38	**4.73**
	MAPE	13.00%	9.60%	9.30%	9.90%	9.60%	9.48%	7.30%	**6.57%**
30 mins	MAE	4.16	5.15	5.05	4.23	3.77	5.07	3.15	**2.79**
	RMSE	7.8	10.45	10.87	8.17	7.23	10.28	6.45	**5.61**
	MAPE	13.00%	12.70%	12.10%	12.90%	10.90%	12.90%	8.80%	**7.45%**
60 mins	MAE	4.16	6.9	6.72	4.49	4.37	7.16	3.6	**3.28**
	RMSE	7.8	13.23	13.76	8.69	8.69	13.43	7.59	**6.68**
	MAPE	13.00%	17.40%	16.70%	14.00%	13.20%	13.45%	10.50%	**9.08%**
					PEMS-BAY				
T	Metric	HA	ARIMAKal	SVR	FNN	FC-LSTM	ST-GCN	DCRNN	Our model
15 mins	MAE	2.88	1.62	1.85	2.2	2.05	1.46	1.38	**1.22**
	RMSE	5.59	3.3	3.59	4.42	4.19	3.24	2.95	**2.78**
	MAPE	6.80%	3.50%	3.80%	5.19%	4.80%	3.01	2.90%	**2.76%**
30 mins	MAE	2.88	2.33	2.48	2.3	2.2	1.94	1.74	**1.59**
	RMSE	5.59	4.76	5.18	4.63	4.55	4.27	3.97	**3.61**
	MAPE	6.80%	5.40%	5.50%	5.43%	5.20%	4.59%	3.90%	**3.43%**
60 mins	MAE	2.88	3.38	3.28	2.46	2.37	2.52	2.07	**1.77**
	RMSE	5.59	6.5	7.08	4.98	4.96	5.52	4.74	**4.36**
	MAPE	6.80%	8.3	8.00%	5.89%	5.70%	6.05%	4.90%	**4.29%**

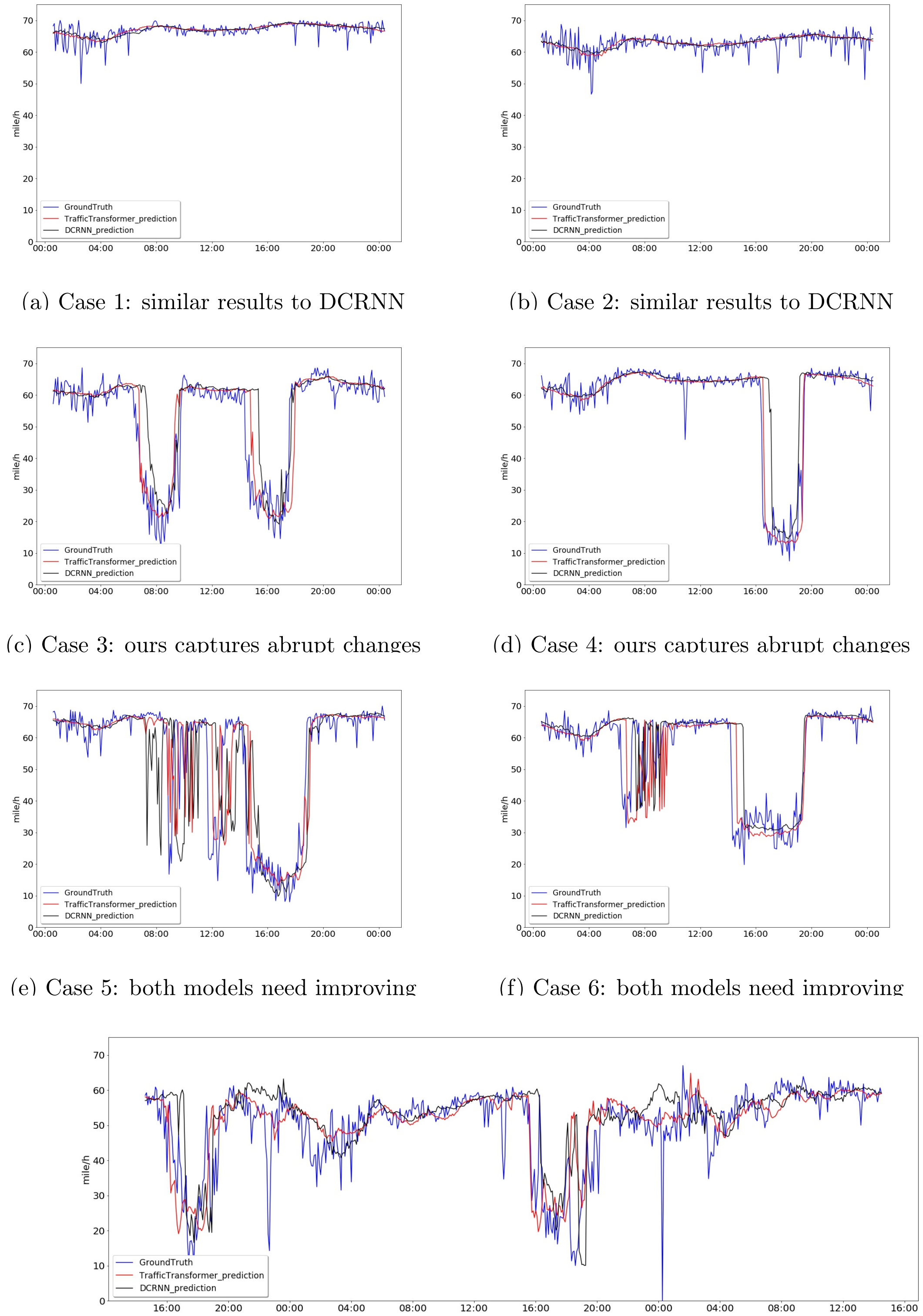
models are unable to capture the complicated spatiotemporal dependency in traffic. (2) The prediction performance of Traffic Transformer is way better than the other models on both datasets in terms of all the evaluation metrics. Especially on the more challenging dataset ***METR-LA***, the average prediction error is reduced by **26.8%**, **12.4%**, **34.5%** in terms of MAE, RMSE, MAPE, respectively. These improvements demonstrate the effectiveness of our architecture in modeling complex spatiotemporal dependencies for traffic prediction.

In addition, to have a better understanding of why our model performs the best, we visualize the forecasting results of our model and the DCRNN model. From Figure 3.3a and 3.3b, we can observe that the DCRNN and our model achieve almost the same performance in the sense that both models are good at dealing with relatively stable traffic conditions. Figure 3.3c and 3.3d illustrate our model has a better ability to accurately capture the abrupt changes in the traffic and the predicted traffic is aligned with the ground truth better than that of DCRNN. However, when the traffic changes very frequently and abruptly, as shown in Figure 3.3e and 3.3f, our model also struggles, although it is still better than DCRNN. More attention should be paid to this extreme circumstance in the future. In addition, we observe that our model and the DCRNN models yield completely different forecasting results between 22:00-2:00 in Figure 3.3g. Apparently, our prediction results are much more accurate. By comparing the similarity of the two forecasting time series at that time interval in this subfigure, we attribute our success to the ability of our model in capturing periodic patterns.

Experimental Comparison of Different Temporal Encoding Methods

With the aim to evaluate the effectiveness of our model in capturing temporal features, we compare the prediction performance of our models under different encoding methods. For most of the encoding methods, we consider two ways of injecting position embeddings to our models, namely addition-based combination (AC) and similarity-based combination (SC). However, we only apply AC to the encoding methods which are centered around the periodic position encoding strategy, since the similarity between the hybrid periodic position embedding and the position embedding of the future time step is meaningless. In addition, we also include the results of using the original position encoding (OPE) strategy in Transformer. This experiment is conducted on METR-LA dataset and RMSE is reported.

Table 3.4 describes the comparison results. We observe that: (1) The encoding methods centered around the periodic position encoding strategy, including RPPE and GPPE, yield the worst results. This is because this idea in fact increases the difficulty of the model learning the commonality from training sequences, as more detailed temporal features in the position embedding reduce the similarity between two sequences. (2) Relative position encoding strategy (RPE) works much better than global position encoding strategy (GPE) when Addition-based combination is applied. This is led by the unseen



(a) Case 1: similar results to DCRNN

(b) Case 2: similar results to DCRNN

(c) Case 3: ours captures abrupt changes

(d) Case 4: ours captures abrupt changes

(e) Case 5: both models need improving

(f) Case 6: both models need improving

(g) Case 7: our model benefits from the modeling of the periodicity of time series

Figure 3.3: One-hour ahead traffic speed prediction on METR- LA dataset

position embeddings of the time steps at testing. In the global position encoding strategy, we generate position indexes for the whole time steps, and thus each position embedding is unique. The position embeddings of the time steps at testing set are never trained during the training process. However, when similar-based combination is adopted, they achieve similar results. This attributes to the characteristics of Eq. 3.5 as the similarity between different time steps is only relevant to their time interval. (3) As expected, OPE has higher prediction errors. Since there is no translation relationship between the source sequence and the target sequence in traffic forecasting, the original position encoding strategy is inapplicable to traffic forecasting. (4) The best performance goes out to the TSE method and the similar-based combination shows a better result. This demonstrates the superiority of this encoding idea and the similar-based combination is more appropriate in dealing with temporal information.

Table 3.4: Performance comparison of different temporal encoding methods

	OPE	RPE		GPE		RPPE		GPPE		TSE	
	AC	AC	SC	AC	SC	AC	SC	AC	SC	AC	SC
15 mins	5.75	5.26	5.07	7.47	5.10	8.47	-	8.42	-	4.91	**4.73**
30 mins	6.82	6.30	6.18	9.54	6.15	11.53	-	11.35	-	5.88	**5.61**
60 mins	8.79	7.45	7.30	12.57	7.32	13.99	-	14.08	-	6.91	**6.68**

Benefits of Modeling Spatial Dependency

Aside from temporal dependencies, here we compare the performance of different graph convolutional operators in modeling spatial dependencies. Three variants of our model-Traffic Transformer (TT for short) are: (1)TT-GCN, which utilizes the well-known GCN to capture spatial dependencies. (2)TT-DCN, which assumes the traffic flow follows a diffusion process. (3)TT-NO, which neglects the role of spatial dependencies.

Figure 3.4 shows the prediction performance of these variants with the same parame-

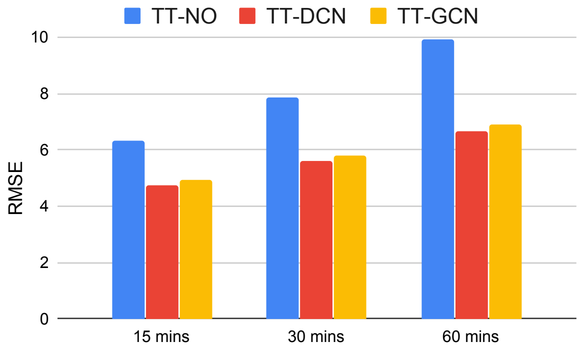


Figure 3.4: Prediction results on METR-LA dataset

ters. Without explicitly modeling the spatial dependency, TT-NO yields the worst results in terms of all the horizons. This effectively demonstrates the necessity of *explicitly* modeling the spatial dependency. Furthermore, TT-DCN consistently outperforms TT-GCN, though, by a small margin. We believe this is because DCN can account for the direction of the traffic flow while GCN fails.

3.6 Conclusion

In this research we introduced the novel non-recurrent architecture *Traffic Transformer*. We successfully used it for traffic forecasting to capture spatiotemporal dependencies. This architecture can be regarded as an extension of Transformer, a well-known sequential model in natural language processing. In order to explore how to capture temporal dependencies in Transformer, we proposed seven different ways of modeling the continuity and the periodicity of time series. The time series segment method achieved the best result. Additionally, we introduced a graph convolutional neural network into Transformer for modeling spatial dependencies of the traffic. By doing so, the dynamic spatiotemporal characteristics of traffic can be captured. Extensive experiments on two benchmark datasets showed that our model is superior to the baselines, demonstrating the effectiveness of our proposed temporal encoding method and our proposed overall architecture. One limitation of this paper is that we only account for temporal attention mechanism, while different upstream roads at different time may contribute differently. In the future work, we will explore how to design a spatiotemporal attention mechanism to address this at a finer scale.

Chapter 4

Time in a Box: Advancing Knowledge Graph Completion with Temporal Scopes

This chapter is related to Research Question 1 and 2. Firstly, given that a time interval can be modeled by a collection of time instants that are within it, I use representations of a collection of time instants to represent both semi-intervals and closed intervals. However, different time intervals may contain various number of instants, which makes subsymbolic methods hard to be optimized. A random sampling method is introduced to address this issue. In the end, for each statement, a unified input can be generated regardless of the type of its temporal information. In addition, a method for linking different types of temporal information to atemporal statements is developed, in which I assume temporal information can be viewed as another relation that may hold between entities. This chapter provides a complete landscape of applying subsymbolic methods for quantitative temporal reasoning.

Peer Reviewed Publication	
Title	Time in a Box: Advancing Knowledge Graph Completion with Temporal Scopes
Author	Ling Cai, Krzysztof Janowicz, Bo Yan, Rui Zhu, and Gengchen Mai
Venue	Proceedings of the 11th on Knowledge Capture Conference
Editors	Anna Lisa Gentile and Rafael Gonçalves
Publisher	Association for Computing Machinery
Pages	121-128
Publication Date	2 December 2021
Copyright	Reprinted with permission from ACM
DOI	https://doi.org/10.1145/3460210.3493566

Abstract Almost all statements in knowledge bases have a temporal scope during which they are valid. Hence, knowledge base completion (KBC) on temporal knowledge bases (TKB), where each statement *may* be associated with a temporal scope, has attracted growing attention. Prior works assume that each statement in a TKB *must* be associated with a temporal scope. This ignores the fact that the scoping information is commonly missing in a KB. Thus prior work is typically incapable of handling generic use cases where a TKB is composed of temporal statements with/without a known temporal scope. In order to address this issue, we establish a new knowledge base embedding framework, called TIME2BOX, that can deal with atemporal and temporal statements of different types simultaneously. Our main insight is that answers to a temporal query always belong to a subset of answers to a time-agnostic counterpart. Put differently, time is a filter that helps pick out answers to be correct during certain periods. We introduce boxes to represent a set of answer entities to a time-agnostic query. The filtering functionality of time is modeled by intersections over these boxes. In addition, we generalize current evaluation protocols on time interval prediction. We describe experiments on two datasets and show that the proposed method outperforms state-of-the-art (SOTA) methods on both link prediction and time prediction.

4.1 Introduction

A knowledge base (KB) such as Wikidata and DBpedia stores statements about the world around us. A KB is typically represented as a set of triples in the form of (s, r, o) – short for *(subject, relation, object)*, encoding the association between entities and relations among them. A statement is often temporally scoped, which indicates during which time period it is valid. Two examples are (*Albert Einstein, educatedAt, ETH Zurich, 1896 - 1900*) and (*Albert Einstein, academicDegree, Doctor of Philosophy in Physics, 1906*). The former specifies the time period during which Albert Einstein studied at ETH, and the latter points out the specific date when he obtained his degree. Graphs that contain a substantial amount of such time-aware statements are often called temporal knowledge base (TKB) in the machine learning literature. Each statement in a TKB is associated with a validity time as (s, r, o, t^*)[1].

Due to the ever-changing state of the world and missing data, TKBs usually contain inaccurate and incomplete information similar to KBs. The sparsity of TKBs necessitates temporal knowledge base completion (TKBC), namely inferring missing statements from known statements. Temporal link prediction task is proposed to evaluate a TKBC model by testing its performance on answering incomplete temporal queries of the form (s, r, $?o$, t^*) or ($?s$, r, o, t^*).

Despite recent success stories on time-agnostic KBC, research on TKBC is still in its early age and is facing new challenges. The validity time period of a statement is often missing in a KB. As a result, it is difficult to distinguish whether statements in a KB are atemporal (e.g., (*Albert Einstein, instanceOf, Human*)) or time-dependent (e.g., (*United States of America, instanceOf, Historical Unrecognized State*[2])). This leads to the question of which statements should be part of a TKB in the first place. Prior

[1] t^* could be a time instant or time interval

[2] According to Wikidata that statement holds true during 1776-1784.

works restrict TKBs to a collection of statements where the validity time period for each statement *must* be available. However, in WIKIDATA114k, a dataset from Wikidata, for instance, 85.1% of all statements are temporal, 56.2% of the temporal statements are missing their validity temporal information and are excluded in previous studies while only 247,393 out of 1,660,824 statements (i.e., 14.9%) are truly atemporal [3]. As the number of temporal statements with missing validity information is substantial, excluding them from a TKB will significantly reduce the amount of information that could be useful in TKB studies.

Retaining these temporally scoped statements leads to several challenges that need to be addressed. For instance, how to design a TKBC model to handle statements with and without known temporal scoping from the data representation perspective and model design perspective? Clearly, the conventional representation in prior TKBC in the form of $(s,\ r,\ o,\ t)$[4] falls short. An ideal TKBC model should be more *flexible* to address cases when the validity information of different types (i.e., point in time, right-open interval (known start time), left-open interval (known end time), closed interval) is presented in a TKB or even no validity information is available for a statement.

The second challenge is how to predict the temporal scope of a statement as it is often missing in TKBs. This task is referred to as time interval prediction, which amounts to answering incomplete queries of the form $(s,\ r,\ o,\ ?I)$. How to generate a predicted time interval and evaluate it require further investigation. This problem has only been addressed very recently by Jain et al. [106]. However, at times their evaluation protocols fail to distinguish one predicted interval from another since they do not consider the gap between the predicted and the gold interval in case of no overlap. For instance, the same

[3] For all the statements, we first categorize predicates into two groups – atemporal predicates and temporal predicates. If a predicate has ever been involved in a statement that has temporal scoping, it belongs to temporal predicates; otherwise, it is an atemporal predicate. Atemporal statements are those associated with atemporal predicates.

[4] t denotes a time point

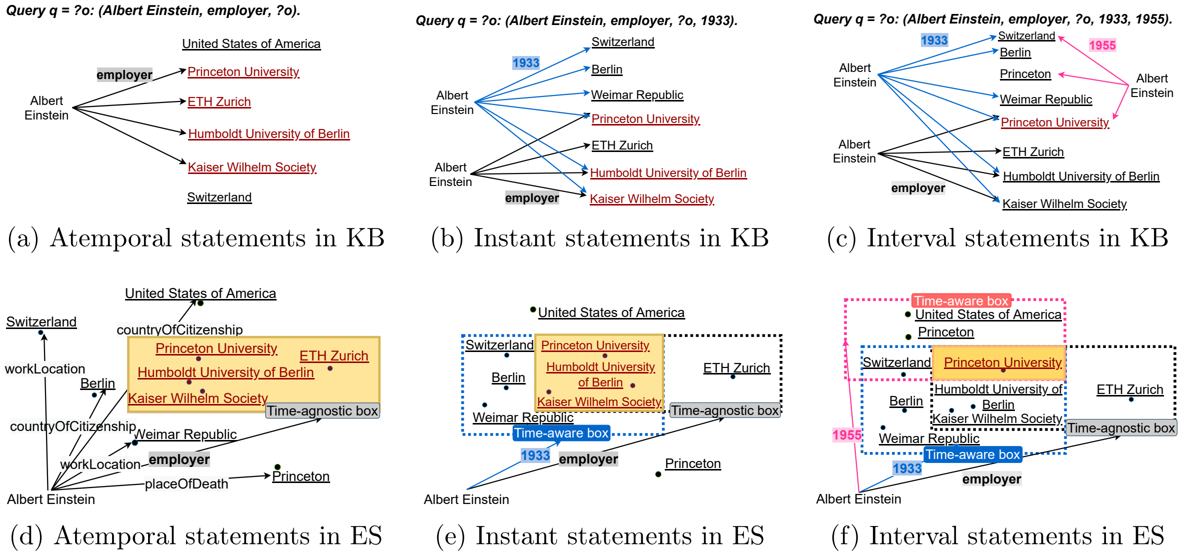
metric scores are assigned to two predictions [1998, 1999] and [1998, 2010] when a gold interval [2011, 2020] is considered.

In this paper, we present a novel TKBC embedding framework, called TIME2BOX, which relies on the intuition that the answer set in a temporal query $(s, r, ?o, t*)$ is always a subset of answers of its time-agnostic counterpart $(s, r, ?o)$. As illustrated in Figure 4.1, there are four correct answer entities to a query (*Albert Einstein, employer, ?o*). However, when temporal information is specified (e.g., *Albert Einstein, employer, ?o, 1933*) as shown in Figure 4.1e, the number of positive answers becomes three. With more temporal information being available (e.g., *Albert Einstein, employer, ?o, [1933, 1955]*), the answer set shrinks further (see Figure 4.1f). Therefore, we propose to model a statement in a TKB by imitating the process of answering its corresponding temporal query$(s, r, ?o, t^*)$, which can be achieved in two steps – finding answer entities to its atemporal counterpart (*s, r, ?o*) by using KBC methods and then picking out entities to be true to the temporal query from preceding answers by including time. We implement this idea by using box embeddings, especially inspired by QUERY2BOX [107], which is originally used for answering conjunctive queries. Boxes, as containers, can naturally model a set of answers they enclose. The filtering functionality of time can be naturally modeled as intersections over boxes similarly to Venn diagrams. Meanwhile, performing an intersection operation over boxes would still result in boxes, thus making it possible to design a unified framework to deal with statements of different types.

Our main research contributions are listed as follows:

- We propose a box-based KG embedding TKBC framework (TIME2BOX) that can represent and model statements with different types of validity information (i.e., unknown temporal scoping, left/right-open intervals and closed intervals).

- We introduce a new evaluation metrics *gaeIOU* for interval evaluation by taking



(a) Atemporal statements in KB (b) Instant statements in KB (c) Interval statements in KB

(d) Atemporal statements in ES (e) Instant statements in ES (f) Interval statements in ES

Figure 4.1: **Illustration of TIME2BOX reasoning process.** In each figure, the upper part shows entities and relations in the KB space and the latter illustrates their correspondences in the embedding space. In all figures, the final boxes are shaded regions in orange and answer entities are in the boxes. Note that we omit the edges between *Albert Einstein* and associated entities in Figure 4.1e and 4.1f for simplicity. Figure 4.1d shows that for an atemporal query, the reasoning process picks out all possible answers from the whole entity space and encloses them into a time-agnostic box. In Figure 4.1e a time-aware box is added to enclose entities that are relevant to *Albert Einstein* in 1933. Then the intersection between time-agnostic and time-aware boxes consists of a new box, which contains entities that satisfy both requirements. When more validity information is available, Figure 4.1f shows that more time-aware boxes can be added and the intersection box that contains correct answers would shrink further. By doing so, TIME2BOX is flexible enough to handle different types of queries.

the gap between a gold interval and a predicted interval into consideration if no overlap exists.

- Extensive experiments on two datasets - WIKIDATA12k and WIKIDATA114k - show that TIME2BOX yields the state-of-the-art (SOTA) results in link prediction and outperforms SOTAs in time prediction by significant margins. TIME2BOX code is available at Github[5].

4.2 Related Work

Knowledge Base Completion The core insight of KBC is to embed entities and relations in a KB into low-dimensional vectors, which can be utilized in downstream tasks, such as link prediction. These methods can be roughly classified into two groups: transformation-based models [62, 65] and semantic matching energy based models [66, 68]. All KBC models ignore the temporal scoping of statements, and thus are unable to address temporal statements. However, these models are the foundations for TKBC.

Temporal Knowledge Base Completion There are two lines of works on temporal link prediction. The first assumes that knowledge in KBs evolves over time and historical statements/events drive the occurrence of new events [108, 109]. The other line is to fill in missing components in TKGs with/without explicitly modeling temporal dependencies between statements. We are on the second line. However, our idea is to deal with cases when time could be an instant, a (left/right-open) interval, closed interval or even missing, while prior works can only handle timestamped statements.

[5] `https://github.com/ling-cai/Time2Box`

4.3 Preliminaries

4.3.1 Temporal Knowledge Bases

Prior TKBC methods typically work on TKBs in which each statement has to be associated with validity information. Thereby, for statements that do not have known temporal scopes, they either exclude them from a TKB in the beginning or assume that these statements hold all the time [110]. However, there are limitations in both ways. As discussed in Section 4.1, excluding them from a TKB will significantly reduce the amount of information that could be beneficial in TKBC studies as the number of such statements is substantial. For the latter, their assumption would be problematic since a lot of them may only hold for a certain time period. For instance, the statement (*Warsaw, country, Russian Empire*) holds during the time interval [1815-07-09, 1916-11-04]. Following the open-world assumption (OWA), we argue that TKBs are an extension to KBs insofar as the lack of temporal scoping for any given statement does not imply it holding indefinitely.

In the following, we use t and $I_{st}^{et} = [st, et]$ to denote a time point and a time interval, respectively. The symbol $-$ will stand for unknown temporal validity. There are five types of statements in such a TKB: (1) (s, r, o) for a statement without a known temporal scope; (2) (s, r, o, t) for a timestamped statement which holds at a point in time t; (3) (s, r, o, I_{st}^{-}) for a right-open interval-based statement, in which only the time when the statement starts to hold is known; (4) (s, r, o, I_{-}^{et}) for a left-open interval-based statement, in which only the time when the statement ceases to hold is known; and (5) (s, r, o, I_{st}^{et}) for a statement which is temporally scoped by a closed interval I_{st}^{et}. Then a TKB is denoted as $\mathcal{G} = \bigcup_{(s,r,o,t^*)}$, namely the union of statements of the five types, where $s, o \in E$ represent entities, $r \in R$ denotes a relation and $t^* \in \{t, I_{st}^{-}, I_{-}^{et}, I_{st}^{et}, None\}$ denotes different types of valid time or no valid time available.

4.3.2 The TKBC Problem

Link prediction and time prediction are two main tasks used to evaluate a TKBC model. Statements in TKBs are split into training, validation, and test sets, used for model training, parameter tuning and model evaluation, respectively.

Link prediction Queries used in this task are of the form (s, r, ?o, t^*). Performance is evaluated on the rank of a given golden, i.e., ground truth, answer in the list of all the entities sorted by scores in a descending order. Then MRR (mean reciprocal rank), MR (mean rank), HITS@1, HITS@3 and HITS@10 are computed from the ranks over all queries in the test set. However a query may be satisfied by multiple answer entities. Thus another correct answer may be ranked over the given golden answer. In such cases, a KBC/TKBC model should not be penalized. A traditional strategy used in KBC is to filter out those correct answers that are already in the training and validation sets before calculating metrics. This strategy can be directly applied to queries of the form (s, r, ?o) or (s, r, ?o, t). However, it may not be sufficient for queries of the form (*s, r, ?o, I*), as there may exist other answers that are true during a time period within the interval I. For example, suppose two statements – *(Albert Einstein, employer, Princeton University, [1933, 1955])* and *(Albert Einstein, employer, Leiden University, [1920, 1946])*, both Princeton University and Leiden University are correct answers during the period [1933, 1946]. One naive way to solve this problem is to discretize the interval I to a sequence of time points ts and then to convert (*s, r, ?o, I*) into timestamped queries of the form (*s, r, ?o, t*) so that the same filtering process can be performed on each timestamped query. Finally, the ranks over them are averaged to be the rank for a time interval-based query. This idea is well-aligned with the proposal by Jain et al. [106].

Time prediction Time prediction queries in TKBs are of the form (s, r, o, $?I$). Despite the fact that the validity information could be a point in time or a time interval, a point in time can be viewed as a special time interval, in which start time and end time coincide.

Thus, time prediction boils down to time interval prediction. Its performance is evaluated by the overlap between a gold interval and a predicted interval or the closeness between those in case of no overlap. We describe the existing evaluation protocols and propose a generalized evaluation metric in Section 4.5.

4.4 Method

The key insight of TIME2BOX lies in an intuition that the answer set of a temporal query (s, r, $?o$, t^*) is always a subset of answers of its time-agnostic counterpart (s, r, $?o$) and set size decreases by adding more temporal constraints.

As illustrated in Figure 4.1d, four object entities satisfy the atemporal query (*Albert Einstein, employer, ?o*) while three entities are the correct answers when the query is restricted to the year of 1993 (see Figure 4.1e) and only one entity is correct when another temporal information is further added in the statement, shown in Figure 4.1f. Inspired by this observation, we propose to model a temporal statement (s, r, o, t^*) by imitating the process of answering its corresponding temporal query (s, r, $?o$, t^*), which can be achieved through two steps: 1) finding a set of answer entities that are true for the corresponding atemporal query by using any KBC model and 2) imposing a filtering operation enforced by time to restrict answers afterwards. In following sections, we take a time instant-based statement as example to formalize our idea in a KB space and a vector space, respectively.

4.4.1 Formalization in a KB Space

For a statement (*s, r, o, t*) in a TKB, the first step of TIME2BOX, as shown in Figure 4.1d, is to project the subject s to a set of object entities that are true to its corresponding atemporal query in the form of (*s, r, ?o*) enforced by the relation r. This is a prerequisite for any statement and can theoretically be addressed by any KBC method. Formally, the relation projector is defined as:

Relation Projector – $\mathbf{OP_r}$: Given the subject entity s and the relation r, this operator obtains: $S_r = \{o' \mid (s, r, o') \in \mathcal{G}^N\}$. $\mathcal{G}^N$ is the time-agnostic counterpart of $\mathcal{G}$.

Then temporal information is used to filter out entities that are incorrect during the time of interest from the answer set S_r. This can be achieved by first *projecting* the subject s to a set of object entities that co-occur with s in statements at a given time point (as shown in blue edges in Figure 4.1e) and then finding the *intersection* over them and S_r (see the three entities in red in Figure 4.1e). Accordingly, the two involved steps are defined as:

Time Projector – $\mathbf{OP_t}$: Given the subject s and the timestamp t, this operator obtains: $S_t = S_t = \{o' \mid o' \in E \textit{ and } (s, r', o', t) \in \mathcal{G} \textit{ and } r' \in R\}$.

Intersection Operator – OI: Given S_r and S_t, this operator obtains the intersection $S_{inter} = \{o \mid o \in (S_r \textit{ and } S_t)\}$.

In fact, such a modeling process also fits to left/right-open interval-based statements directly. For a left/right-open interval-based statement, we only consider the known endpoint time in such an interval as we follow the open-world assumption. However, for an atemporal statement, we only need one relation projector to obtain S_r, which is the final set consisting of correct answer entities to its query form. For a closed interval-based query, one commonly used approach is to randomly pick one timestamp within the interval and to associate it with (*s, r, o*). Then it can be modeled the same way as an instant-based statement. At training, a timestamp is always randomly picked from the interval to ensure that all the timestamps in the interval are used. In addition to the common strategy, TIME2BOX allows sampling of a sub-time interval within the given interval so that two temporal constraints (i.e., start time and end time) can be imposed by using two temporal projectors, as shown in Figure 4.1f[6].

4.4.2 Implementation in a Vector Space

In order to implement this idea in a vector space, two key points are 1) how to model a set of answers returned by a KBC model and 2) how to instantiate two projectors and one intersection operator.

Prior KBC models are incapable of directly representing a set of answer entities in a vector space. Instead, they usually represent entities and relations as single points in the vector space and model point-to-point projections, e.g., TransE.

Inspired by QUERY2BOX [107], which is used to deal with complex queries that involve conjunctions, existential quantifiers, and disjunctions, we introduce the idea of boxes in the vector space and thus name the proposed framework TIME2BOX. The reasons for adopting boxes are three-fold. First, boxes are containers that can naturally model a set of answer entities they enclose. Second, finding the intersection set among sets of entities amounts to finding the intersected area over boxes similar to the concept of Venn diagram. Third, the result of performing an intersection operation over boxes is still a box, which makes it possible to deal with statements of different types in a unified framework.

[6] Alternatively, one could also enumerate all the timestamps within the interval and use different $\mathbf{OP_t}$ to project the subject to multiple sets of entities, each of which is specific for one timestamp. Subsequently, an intersection operator again is performed over all the sets of entities obtained from $\mathbf{OP_r}$ and $\mathbf{OP_t}$ in the previous step. However, in spite of its efficiency, this practice is hard to implement in mini-batch training manner since time intervals in different statements usually have varying duration and thus contain different number of timestamps.

In TIME2BOX, each entity $e \in E$, relation $r \in R$, and timestamp $t \in T$ (T is the set of all discrete timestamps in a TKB) are initialized as vector embeddings $\mathbf{e} \in \mathbb{R}^d$, $\mathbf{r} \in \mathbb{R}^d$, and $\mathbf{t} \in \mathbb{R}^d$. S_r, S_t, and S_{inter} refer to sets of entities and thus are modeled by boxes, represented as box embeddings in the vector space. In the following, we first introduce the definition of box embeddings and then introduce main components of modeling and reasoning.

Box Construction and Reasoning

Box embeddings: Mathematically, they are axis-aligned hyper-rectangle in a vector space, which can be determined by the position of the box (i.e., a center point) and its length (i.e., offsets). Formally, in a vector space $\mathbb{R}^d$, a box can be represented by $\mathbf{b}$=(Cen($\mathbf{b}$), Off($\mathbf{b}$)), where Cen($\mathbf{b}$) $\in \mathbb{R}^d$ is its center point and Off($\mathbf{b}$) $\in \mathbb{R}^d_{\geq 0}$ specifies the length/2 of the box in each dimension. If an entity belongs to a set, its entity embedding is modeled as a point inside the box of the set. The interior of a box in the vector space can be specified by points inside it:

$$box_{\mathbf{b}} = \{\mathbf{e} \in \mathbb{R}^d : \mathrm{Cen}(\mathbf{b}) - \mathrm{Off}(\mathbf{b}) \preceq \mathbf{e} \preceq \mathrm{Cen}(\mathbf{b}) + \mathrm{Off}(\mathbf{b}))\} \tag{4.1}$$

where $\preceq$ denotes element-wise inequality.

Projection operators in a vector space In previous work, relations are commonly assumed to be projectors that transform a subject embedding to an object embedding in terms of *points* in a vector space, e.g., TransE [62] and RotatE [65]. Here we adopt a similar idea but take both relations and timestamps as projectors ($\mathbf{OP_r}$ and $\mathbf{OP_t}$) to project a subject to a set of entities in S_r – represented as a time-agnostic *box* $\mathbf{b}_{S_r}$ and to a set of entities in S_t – represented as a time-aware *box* $\mathbf{b}_{S_t}$, respectively, which are illustrated in Figure 4.1.

The center of a box can be defined as the resulting embedding after applying a projection operator ($\mathbf{OP_r}$ or $\mathbf{OP_t}$) on the subject embedding. The centers of $\mathbf{b}_{S_r}$ and $\mathbf{b}_{S_t}$ can be formulated as below:

$$Cen(\mathbf{b}_{S_r}) = \mathbf{e} \odot \mathbf{r}; \quad Cen(\mathbf{b}_{S_t}) = \mathbf{e} \otimes \mathbf{t} \tag{4.2}$$

where $\odot\mathbf{r}$ and $\otimes\mathbf{t}$ are projectors $\mathbf{OP_r}$ and $\mathbf{OP_t}$, respectively. Theoretically, projection operators could be instantiated by any projector in existing KBC models, such as element-wise addition in TransE [62] , element-wise product in DistMult [66] , and Hadamard product in RotatE [65]. Even though $\mathbf{OP_r}$ and $\mathbf{OP_t}$ can be different, we choose the same projector for both and implement two TIME2BOX models by taking element-wise addition and element-wise product as operators by following TransE and DistMult, respectively. Accordingly, these two models are named as TIME2BOX-TE and TIME2BOX-DM.

Ideally, the size of the box $\mathbf{b}_{S_r}$ should be determined by both the subject entity and the relation, since the box contains all object entities that satisfy a query in the form of $(s,\ r,\ ?o)$. The same applies to $\mathbf{b}_{S_r}$. However, as the entity space is usually large in a KB, introducing entity-specific parameters would result in high computational cost. Therefore, in practice, $\text{Off}(\mathbf{b}_{S_r})$ and $\text{Off}(\mathbf{b}_{S_t})$ are only determined by the relation $r \in R$ and the timestamp $t \in T$, respectively. Put differently, the size of $\mathbf{b}_{S_r}$ and $\mathbf{b}_{S_t}$ are initialized based on r and t, which are learned through training.

Intersection Operators in a vector space An intersection operator aims to find the intersection box $\mathbf{b}_{inter} = (\text{Cen}(\mathbf{b}_{inter}), \text{Off}(\mathbf{b}_{inter}))$ of a set of box embeddings $\mathbf{B} = \{\mathbf{b}_{S_r}, \mathbf{b}_{S_t1}, ..., \mathbf{b}_{S_tn}\}$ obtained from the previous step. The intersection operator should be able to deal with $\mathbf{B}$ of different sizes, as required in Figure 4.1. Thus, both $\text{Cen}(\mathbf{b}_{inter})$ and $\text{Off}(\mathbf{b}_{inter})$ are implemented by using attention mechanisms. Following the idea in [111], the center point $\text{Cen}(\mathbf{b}_{inter})$ is calculated by performing element-wise attention over the centers of boxes in $\mathbf{B}$. This can be formulated as follows:

$$\text{Cen}(\mathbf{b}_{inter}) = \sum_i \text{softmax}(\text{NN}(\text{Cen}(\mathbf{b}_i)) \odot \text{Cen}(\mathbf{b}_i) \tag{4.3}$$

where NN is a one-layer neural network and $\mathbf{b}_i \in \mathbf{B}$.

Since the intersection box $\mathbf{b}_{inter}$ must be smaller than any of the box in $\mathbf{B}$, we use element-wise min-pooling to make sure the new box must be shrunk and perform DeepSets [112] over all the $\text{Off}(\mathbf{b}_i)$ ($\mathbf{b}_i \in \mathbf{B}$) to downscale $\mathbf{b}_{inter}$ [107]. This can be written as below:

$$\text{Off}(\mathbf{b}_{inter}) = \text{Min}(\mathbf{Off}) \odot \sigma(\text{DeepSets}(\mathbf{Off})) \tag{4.4}$$

where $\text{DeepSets}(\{\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_n}\}) = \text{MLP}(1/n) \cdot \sum_i^n \text{MLP}(\mathbf{x_i})$, σ denotes the sigmoid function, and $\mathbf{Off} = \{\text{Off}(\mathbf{b}_i) : \mathbf{b}_i \in \mathbf{B}\}$.

4.4.3 Optimization Objective

For a query, TIME2BOX aims to pull correct entity embedding into the final box $\mathbf{b}_{inter}$ while pushing incorrect entity embedding far away from it. The distance-based loss proposed by Sun et al. [65] satisfies this need :

$$Loss = -log\ \sigma(\gamma - D(\mathbf{o}, \mathbf{b}_{inter})) - \frac{1}{k}\sum_{i=1}^{k} log\ \sigma(D(\mathbf{o}', \mathbf{b}_{inter}) - \gamma) \tag{4.5}$$

where σ is the sigmoid function, γ is a fixed margin, $\mathbf{o}$ is the embedding of a positive entity to the given query, and k is the number of negative samples $\mathbf{o}'$. $D(\mathbf{o}, \mathbf{b}_{inter})$ measures the distance between entity $\mathbf{o}$ and the final box $\mathbf{b_{inter}}$. With the size of a box being considered, the distance is divided into two parts: outside distance $D_{outside}(\mathbf{o}, \mathbf{b}_{inter})$ and inside distance $D_{inside}(\mathbf{o}, \mathbf{b}_{inter})$. For cases when $\mathbf{o}$ is outside of $\mathbf{b}_{inter}$, the former refers to the distance of an entity embedding $\mathbf{o}$ to the boundary of the box $\mathbf{b}_{inter}$, and the latter calculates the distance between the box's center $Cen(\mathbf{b}_{inter})$ and its boundary. This can be formalized as below:

$$D(\mathbf{o}, \mathbf{b}_{inter}) = \alpha \cdot D_{inside}(\mathbf{o}, \mathbf{b}_{inter}) + D_{outside}(\mathbf{o}, \mathbf{b}_{inter}) \tag{4.6}$$

where $\alpha \in [0, 1]$. When $\alpha = 0$, it means that a positive entity is required to be in a $\mathbf{b}_{inter}$, but its distance to the center is not as important. $D_{inside}(\mathbf{o}, \mathbf{b}_{inter})$ and $D_{outside}(\mathbf{o}, \mathbf{b}_{inter})$ are written as:

$$D_{inside}(\mathbf{o}, \mathbf{b}_{inter}) = \|\mathrm{Cen}(\mathbf{b}_{inter}) - \mathrm{Min}(\mathbf{b}_{max}, \mathrm{Max}(\mathbf{b}_{min}, \mathbf{o}))\|_1$$
$$D_{outside}(\mathbf{o}, \mathbf{b}_{inter}) = \|\mathrm{Max}(\mathbf{o} - \mathbf{b}_{max}, \mathbf{0}) + \mathrm{Max}(\mathbf{b}_{min} - \mathbf{o}, \mathbf{0})\|_1$$

where $\mathbf{b}_{min} = \mathrm{Cen}(\mathbf{b}_{inter}) - \mathrm{Off}(\mathbf{b}_{inter})$ and $\mathbf{b}_{max} = \mathrm{Cen}(\mathbf{b}_{inter}) + \mathrm{Off}(\mathbf{b}_{inter})$ are embeddings of the bottom left corner and the top right corner of $\mathbf{b}_{inter}$, respectively.

Compared to answering atemporal queries, finding correct answers to temporal ones is more challenging. Therefore, the loss function should reward more in the optimization direction that is capable of correctly answering temporal queries. For a given query q_i, we use $\frac{1}{n_{q_i}}$, where n_{q_i} is the number of correct answers to q_i that appear in training as a weight to adjust the loss. The core idea here is that time-aware queries often are satisfied with fewer answers, and, thus, are harder to answer compared to atemporal queries.

4.4.4 Time Negative Sampling

Entity negative sampling is widely used in KBC. For a positive sample (s, r, o), negative samples are constructed by replacing o with other entities o', ensuring that (s, r, o') must not appear in training set. In this paper, we adopt this strategy so that the model is able to learn the association between entities, relations, and time occurring in a positive sample by distinguishing the correct answers from the negative samples. Moreover, for time-aware statements, we perform temporal negative sampling, which corrupts a statement (s, r, o, t) by replacing t with a number of timestamps t'. This is important for statements where only start time or end time is available. As shown in Figure 4.1, the proposed architecture cannot distinguish those statements from time instant-based statements. But temporal negative sampling can mitigate this issue to some degree. The following is used for temporal negative sampling concerning different types of statements (st and et are short for start and end time):

$$T' = \begin{cases} \{t' \in T : (s, r, o, t') \notin \mathcal{G}\} & (s, r, o, t) \\ \{t' \in T : (s, r, o, t') \notin \mathcal{G}, t' < st\} & (s, r, o, I_{st}^{-}) \\ \{t' \in T : (s, r, o, t') \notin \mathcal{G}, t' > et\} & (s, r, o, I_{-}^{et}) \\ \{t' \in T : (s, r, o, t') \notin \mathcal{G}, t' \notin T_{st}^{et}\} & (s, r, o, I_{st}^{et}) \end{cases} \tag{4.7}$$

where T_{st}^{et} denotes a set of time points within the interval I_{st}^{et}.

4.4.5 Time Smoothness Regularizer

Time is continuous. We may expect that neighboring timestamps would have similar representations in the vector space. Following Lacroix et al. [110], we penalize time difference between embeddings of two consecutive timestamps by using L_2:

$$\Lambda(T) = \frac{1}{|T|-1} \sum_{i=1}^{|T|-1} \|\mathbf{t}_{i+1} - \mathbf{t}_i\|_2^2 \tag{4.8}$$

During the training step, for batches with temporal statements, we add this regularizer with a weight scalar β to the loss function in Eq. 6.1, where β specifies the degree of penalization.

4.5 Evaluation Metrics in Time Prediction

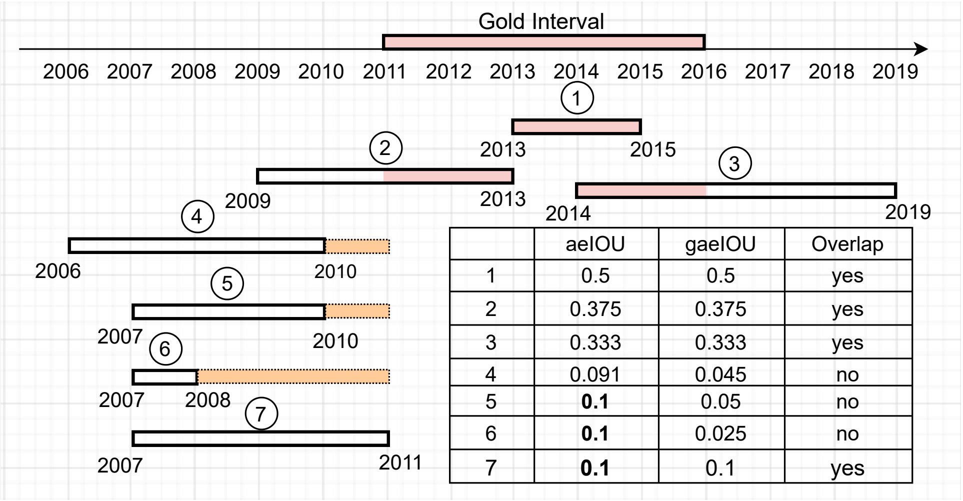
Time Interval Evaluation *gIOU* [113] and *aeIOU* [106] are two evaluation metrics recently adopted in time interval prediction. Both are built on *Intersection Over Union* that is commonly used for bounding box evaluation in Computer Vision.

The idea of *gIOU* is to compare the intersection between a predicted interval and a gold interval against the maximal extent that the two intervals may expand. It can be formulated as below:

$$\begin{gathered} gIOU(I^{gold}, I^{pred}) = \frac{D(I^{gold} \bigcap I^{pred})}{D(I^{gold} \bigcup I^{pred})} - \\ \frac{D(I^{gold} \biguplus I^{pred} \setminus (I^{gold} \bigcup I^{pred}))}{D(I^{gold} \biguplus I^{pred})} \in (-1, 1] \end{gathered} \tag{4.9}$$

where $I^{gold} \bigcap I^{pred}$ is the overlapping part of two intervals, $I^{gold} \biguplus I^{pred}$ denotes the shortest contiguous interval (hull) that contains both I^{gold} and I^{pred}. As shown in Figure 4.2, if $I^{gold} = [2011, 2016]$ and $I^{pred} = [2009, 2013]$, then $I^{gold} \bigcap I^{pred} = [2011, 2013]$ and $I^{gold} \biguplus I^{pred} = [2009, 2016]$. $D(I) = I_{max} - I_{min} + 1$ is the number of time points at a certain granularity (e.g., year in this paper) during the time interval I.

Compared to *gIOU*, affinity enhanced IOU, denoted as *aeIOU*, provides a better



	aeIOU	gaeIOU	Overlap
1	0.5	0.5	yes
2	0.375	0.375	yes
3	0.333	0.333	yes
4	0.091	0.045	no
5	**0.1**	0.05	no
6	**0.1**	0.025	no
7	**0.1**	0.1	yes

Figure 4.2: Evaluation Comparison between aeIOU and gaeIOU on different predicted intervals. Suppose a gold interval is [2011, 2016], seven possible predicted intervals are represented as rectangles in black. Intersections between the predicted and the gold are in pink and gaps are in orange if no overlap exists. Notably, gaeIOU is able to distinguish these predictions while aeIOU fails to do so.

evaluation in case of non-overlapping intervals and outputs scores in $[0, 1]$. It can be written as follow:

$$aeIOU(I^{gold}, I^{pred}) = \begin{cases} \frac{D(I^{gold} \bigcap I^{pred})}{D(I^{gold} \biguplus I^{pred})} & D(I^{gold} \bigcap I^{pred}) > 0 \\ \frac{1}{D(I^{gold} \biguplus I^{pred})} & otherwise \end{cases} \tag{4.10}$$

However, we notice that *aeIOU* cannot tell some cases apart. As illustrated in Figure 4.2, *aeIOU* results in the same scores for ⑤, ⑥, and ⑦ when compared to the gold interval$[2011, 2016]$. Intuitively one would assume that ⑦ is better than the others and ⑥ is the least desirable. The former has a one-year intersection between ⑦ and the gold. For the latter, the gap between ⑤ and the gold is smaller than that between ⑥ and the gold, despite the fact that neither ⑤ and ⑥ overlaps with the gold. Its failure lies in that it does not consider the gap between the gold and the predicted interval in case of no overlap.

In the following, we take both the hull and the intersection/gap between a gold interval and a predicted interval into the design of the metric. The intuition is that when the size of the hull remains the same, the metric score of a predicted interval *decreases* with a larger gap to the gold in case of no overlap and *increases* with a larger intersection. $aeIOU$ is therefore generalized to $gaeIOU$ as below:

$$gaeIOU(I^{gold}, I^{pred}) = \begin{cases} \frac{D(I^{gold} \bigcap I^{pred})}{D(I^{gold} \biguplus I^{pred})} & D(I^{gold} \bigcap I^{pred}) > 0 \\ \\ \frac{D'(I^{gold},\ I^{pred})^{-1}}{D(I^{gold} \biguplus I^{pred})} & otherwise \end{cases} \tag{4.11}$$

where $D'(I^{gold},\ I^{pred}) = max(I^{gold}_{min}, I^{pred}_{min}) - min(I^{gold}_{max}, I^{pred}_{max}) + 1$ is the length of the gap.

Accordingly, the Property (P) that a good evaluation metric must satisfy can be rewritten as: if predicted intervals (partially) overlap with the gold interval with the same size, then the prediction having a smaller hull with the gold interval should be awarded more by M; if there is no overlap, the prediction that has a smaller hull and a narrower gap with the gold should be scored higher by M. It can be formalized as below:

Property P: In case of $D(I^{gold} \bigcap I^{pred1}) = D(I^{gold} \bigcap I^{pred2}) \neq 0$, $M(I^{gold}, I^{pred1}) > M(I^{gold}, I^{pred2})$ if and only if $D(I^{gold} \bigcup I^{pred1}) < D(I^{gold} \bigcup I^{pred2})$.
In case of non-overlapping, $M(I^{gold}, I^{pred1}) > M(I^{gold}, I^{pred2})$ if and only if $D(I^{gold} \bigcup I^{pred1}) \cdot D'(I^{gold} \bigcap I^{pred1}) > D(I^{gold} \bigcup I^{pred2}) \cdot D'(I^{gold} \bigcap I^{pred2}))$.

It follows that $gaeIOU$ satisfies Property P, whereas $aeIOU$ does not satisfy it; see Figure 4.2.

4.6 Experiment

Our goal here is to evaluate TIME2BOX in both link prediction and time prediction tasks. For a test sample $(s, r, ?o, t^*)$, we replace $?o$ with each entity $o' \in E$ and use

$log\sigma(\gamma - D(\mathbf{o}', \mathbf{b}_{inter}))$, a variation of the inverse distance used in Eq. 6.1, as scores for link prediction. Entities that have higher scores are more likely to form new links. Likewise, in terms of time prediction, for a query $(s, r, o, ?I)$, we first replace I with each timestamp $t \in T$ and calculate its score. Then we use the greedily coalescing method proposed in [106] to generate time intervals as predictions.

4.6.1 Datasets

We report experimental results using two TKBC datasets, which both are rooted in Wikidata. WIKIDATA12k is a widely used benchmark dataset in TKBC where each statement is associated with a time "interval" [114]. Such an "interval", in fact, could be a time instant, where start time and end time are the same, a left/right-open interval, or a closed interval. Note that this dataset excludes statements that do not have known temporal scopes in Wikidata, although they may be time-dependent and useful in TKBC, as discussed in Section 4.1. The other dataset is a subset of WIKIDATA432k proposed by [110], which is the only TKB dataset where the start time, end time, or both of a statement can remain unspecified. Although this dataset is more appropriate for a TKBC problem, there are two limitations. First, it poses a computational burden as it contains 432k entities and 407 relations, consisting of 7M tuples in the training set. Second, there are several mistakes in the temporal information. For instance, 2014 was written as 2401. We extract a subgraph, named as WIKIDATA114k, and correct temporal information by checking it against Wikidata. More details about data pre-processing and statistics are in Appendix A.1 (All Appendices are available online[7]). Since our focus is on generic knowledge bases, we do not consider event-based datasets, such as ICEWS14 and ICEWS05-15, in which each statement is associated with a timestamp.

[7] Link to online Appendices.

4.6.2 Baselines and Model Variants

In the following experiments, we regard TIME2BOX-TE as our main model, in which both the relation operator and the time projector are instantiated as an element-wise addition. It is denoted as TIME2BOX in resulting tables. We compare it against two SOTAs in TKBC: TNT-Complex and TIMEPLEX base model by using the implementation in [106], both of which are based on the time-agnostic KBC model: ComplEx [68].

In addition to comparison with existing SOTAs, we also conduct an ablation study, in which several variants of the proposed model are compared: (1) TIME2BOX-SI, short for Sample Interval: for a closed interval-based statement, this variant randomly samples a sub time interval from a given interval at each training step and train it as shown in Figure 4.1f. (2) TIME2BOX-TR: previous works in TKBC often explicitly fused relations with temporal information to obtain time-aware relations and empirically demonstrated its effectiveness [106, 110, 115]. We also explicitly model the association between relations and time as a new point $p_{rt} = \mathbf{r} + \mathbf{t}$ in the vector space and incorporate it into Eq. 4.3 to help locate the intersection box.

(3)TIME2BOX-DM: this variant implements the relation and temporal projectors as an element-wise product in real space as DistMult does.

(4) TIME2BOX-TNS: this variant is used to test the effect of temporal negative samples, in which we replace a number of entity negative samples with temporal negative samples, as introduced in Section 4.4.4.

All these models are trained on statements in training set and evaluated by answering queries where either the object or the temporal information is missing. Hyper-parameter settings are introduced in Appendix A.2 and comparison of parameters used in different models is summarized in Table 11 in Appendix A.6. Moreover, we notice there are several limitations in current experimental setups of SOTAs and we detail them in Appendix A.3.

4.6.3 Main Results

Datasets	WIKIDATA12k				WIKIDATA114k			
Metrics	MRR	MR	HITS@1	HITS@10	MRR	MR	HITS@1	HITS@10
TNT-Complex	31.77	415	19.24	51.74	49.25	638	41.02	66.99
TIMEPLEX base	34.55	302	21.91	53.25	49.99	337	41.25	66.10
TIME2BOX-TR	34.99	102	24.79	56.32	50.25	85	41.73	67.13
TIME2BOX-DM	35.90	139	25.52	56.74	48.84	284	41.09	64.33
TIME2BOX-SI	36.79	**100**	27.16	56.43	50.42	**139**	41.65	67.58
TIME2BOX-TNS	37.25	**100**	**27.41**	57.31	**50.55**	185	**41.77**	67.78
TIME2BOX	**37.30**	101	27.38	**57.36**	50.49	168	41.69	**67.91**

Table 4.1: Link prediction evaluation across two datasets.

Link Prediction Task We report main results of link prediction in Table 4.1. TIME2BOX and all its variants consistently outperform or are on a par with the performance on SOTAs in terms of MRR, MR, HITS@1 and HITS@10. On WIKIDATA12k, TIME2BOX outperforms TIMEPLEX base by around 3 points in terms of MRR and over 5 points in HITS@1. On WIKIDATA114k, TIME2BOX is slightly better than two SOTAs in general for MRR, HITS@1 and HITS@10. In addition, we notice that TIME2BOX beats SOTAs by large margins in time interval-based link prediction, as shown in Table 8 in Appendix A.4. Our method improves around 20 and 7 HITS@1 points in terms of half-open interval-based link prediction and closed interval-based link prediction on WIKIDATA12k, respectively. On WIKIDATA114k TIME2BOX improves around 6 and 4 HITS@1 points, respectively.

Another critical observation in Table 4.1 is the substantial improvements of using TIME2BOX in terms of MR on both datasets. TIME2BOX returns an MR of 100 and 139 on WIKIDATA12k and WIKIDATA114k, respectively and TIMEPLEX base obtains 302 and 337 for MR on both datasets. It indicates that TIME2BOX is capable of giving a fair rank for a gold answer to any test query on average. This is likely because of the idea of using boxes to constraint the potential answer set. As a time-agnostic box is optimized towards embracing entities satisfying atemporal queries of the form $(s, r, ?o)$ in the learning process, boxes implicitly manage to learn common characteristics of the satisfied $?o$. Therefore, TIME2BOX is less likely to output extremely bad predictions. Examples in Section 4.6.4 exemplify this hypothesis.

Time Prediction Task Table 4.2 and Table 4.3 summarizes the results for two datasets. On both datasets, TIME2BOX and its variants consistently outperform SOTAs by significant margins. Specifically, TIME2BOX improves over TIMEPLE by about 5.56, 7.25, and 4.87 points with respect to gIOU@1, aeIOU@1, and gaeIOU@1, respectively, on WIKIDATA12k. As for WIKIDATA114k, despite subtle improvements in link prediction, the advancement of TIME2BOX is more pronounced in time prediction, which shows that it gains 8.7, 5.87, and 4.66 points on gIOU@1, aeIOU@1, and gaeIOU@1, respectively. Furthermore, the improvements on gaeIOU@10 are much more notable with gains of 15.81 and 11.07 points on the two datasets, respectively.

Datasets	WIKIDATA12k					
Metrics	gIOU@1	gIOU@10	aeIOU@1	aeIOU@10	gaeIOU@1	gaeIOU@10
TNT-Complex	31.44	55.18	18.86	40.94	11..01	29.51
TIMEPLEX base	35.63	60.86	18.60	37.75	12.61	32.63
TIME2BOX-TR	39.63	67.83	23.47	44.64	15.87	41.53
TIME2BOX-DM	38.78	62.44	21.91	41.55	14.94	37.14
TIME2BOX-SI	39.68	65.30	23.66	42.16	16.09	38.54
TIME2BOX-TNS	**42.30**	**70.16**	**25.78**	**50.04**	**17.41**	**47.54**
TIME2BOX	41.20	68.53	24.70	46.05	16.98	43.08

Table 4.2: Time prediction evaluation on WIKIDATA12k.

Datasets	WIKIDATA114k					
Metrics	gIOU@1	gIOU@10	aeIOU@1	aeIOU@10	gaeIOU@1	gaeIOU@10
TNT-Complex	27.94	48.31	16.18	35.32	7.31	23.68
TIMEPLEX base	29.31	57.68	18.56	36.70	12.53	32.47
TIME2BOX-TR	37.49	67.95	25.05	49.02	15.41	45.72
TIME2BOX-DM	35.88	66.62	24.33	48.03	14.89	44.48
TIME2BOX-SI	34.02	62.89	23.10	44.74	14.07	40.05
TIME2BOX-TNS	37.31	66.91	25.07	48.18	15.57	44.66
TIME2BOX	**38.01**	**71.29**	**24.42**	**50.07**	**15.88**	**47.77**

Table 4.3: Time prediction evaluation on WIKIDATA114k.

4.6.4 Qualitative Study

Table 4.4 showcases examples of timestamp-based link prediction on WIKIDATA12k. The comparison between TIMEPLEX base and TIME2BOX reveals that TIME2BOX is able to learn common characteristics of entities by adopting boxes. For instance, the predicted top 10 returned by TIME2BOX are possible affiliations (e.g., institutes, colleges, universities) in the first query and are countries in the second query. By contrast, TIMEPLEX base returns a mixture of entities with distinct classes for both queries. Furthermore, Table 4.5 shows an example of time interval-based link prediction, in which TIME2BOX is able to consistently output correct predictions across time and precisely discern the changes of objects over time (i.e., the correct answer shifts from Russian Empire to Ukrainian People's Republic in 1916), while TIMEPLEX base fails. This can be attributed to the ability of TIME2BOX to capture the order of timestamps and the idea of temporal boxes as a constraint over potential answer entities. Hence, answer entities that are true in two consecutive years can be enclosed in the intersection of temporal boxes.

Query Example 1: (Yury Vasilyevich Malyshev, educatedAt, ?o, 1977)	
TIMEPLEX base	TIME2BOX
1. Bauman Moscow State Technical University,	1. Bauman Moscow State Technical University,
2. Gold Star,	**2. Gagarin Air Force Academy,**
3. Communist Party of the Soviet Union,	3. S.P. Korolev Rocket and Space Corporation Energia,
4. Order of Lenin,	4. Saint Petersburg State Polytechnical University,
5. S.P. Korolev Rocket and Space Corporation Energia,	5. University of Oxford,
6. Hero of the Soviet Union,	6. Saint Petersburg State University,
7. **Gagarin Air Force Academy,**	7. Steklov Institute of Mathematics,
8. Balashov Higher Military Aviation School of Pilots,	8. Leipzig University,
9. Ashok Chakra,	9. Heidelberg University,
10. Heidelberg University	10. Moscow Conservatory
Query Example 2: (Pedro Pablo Kuczynski, countryOfCitizenship,?o, 2015)	
TIMEPLEX base	TIME2BOX
1. doctor honoris causa,	1. France,
2. President of Peru,	2. Germany,
3. Minister of Economy and Finance of Peru,	**3. United States of America,**
4. Grand Cross of the Order of the Sun of Peru,	4. Austria,
5. President of the Council of Ministers of Peru,	5. Romania,
6. World Bank,	6. United Kingdom,
7. Serbia,	7. Poland,
8. Royal Spanish Academy,	8. Kingdom of Italy,
9. Meurthe-et-Moselle,	9. Russian Soviet Federative Socialist Republic,
10. Norwegian Sportsperson of the Year	10. Russian Empire

Table 4.4: Examples of timestamp-based link prediction on WIKIDATA12k. Top 10 entities predicted by TIMEPLEX base and TIME2BOX are numbered, where 1 denotes Top One. Correct answers are in bold.

	Query Example: (Kyiv, country, ?o, [1905, 1919])	
	Gold Answers: (1)[1905, 1916]->Russian Empire; (2)[1917, 1919]->Ukrainian People's Republic	
year	Timplex	TIME2BOX
1905	Soviet Union	Russian Empire
1906	Soviet Union	Russian Empire
1907	Russian Empire	Russian Empire
1908	Soviet Union	Russian Empire
1909	Ukrainian Soviet Socialist Republic	Russian Empire
1910	Soviet Union	Russian Empire
1911	Russian Empire	Russian Empire
1912	Ukrainian Soviet Socialist Republic	Russian Empire
1913	Soviet Union	Russian Empire
1914	Ukrainian Soviet Socialist Republic	Russian Empire
1915	Soviet Union	Russian Empire
1916	Ukrainian People's Republic	Russian Empire
1917	Ukrainian People's Republic	Ukrainian People's Republic
1918	Ukrainian People's Republic	Ukrainian People's Republic
1919	Ukrainian People's Republic	Ukrainian People's Republic

Table 4.5: An example of interval-based link prediction on WIKIDATA12k. For time interval-based link prediction, the current strategy is to discretize intervals to timestamps and average ranks for each timestamp-based prediction result as the final evaluation. Only top 1 predictions are shown here.

4.6.5 Model Variation Study

In this section, we report on observations of results about different model variations, which are shown in Table 4.1, 4.2 and 4.3. Compared to TIME2BOX-DM, which adopts element-wise product as operators, element-wise addition projectors (TIME2BOX) perform better in link prediction and time prediction on both datasets. Moreover, we observe that explicitly modeling association between time and relation (i.e., TIME2BOX-TR) does not significantly improve the performance of TIME2BOX framework, although it speeds up the convergence at training, indicating that intersection operators are good enough to learn the association between time and relations implicitly. As for different time-aware strategies incorporated in TIME2BOX-SI and TIME2BOX-TNS, we find that on both datasets TIME2BOX-SI does not outperform TIME2BOX, indicating that using one sample strategy (i.e., Figure 4.1e) is better at modeling time interval-based statements in TIME2BOX. In addition, we find that by incorporating time negative samples, the performance on time prediction can be further improved on WIKIDATA12k, although TIME2BOX-TNS is not superior to TIME2BOX in link prediction.

4.7 Conclusion

In this work, we presented a box-based temporal knowledge graph (TKBC) completion framework (called TIME2BOX) to represent and model statements with different types of validity information (i.e., no time, known start time, known end time, instant, both start and end time) in a vector space. We argued that a TKBC problem can be solved in two steps. First by solving an atemporal KBC problem and then narrowing down the correct answer sets that are only true at the time of interest. Therefore, we introduced time-agnostic boxes to model sets of answers obtained from KBC models. Time-aware boxes are used as a filter to pick out time-dependent answers. TIME2BOX outperforms existing TKBC methods in both link prediction and time prediction on two datasets - WIKIDATA12k and WIKIDATA114K. By investigating the model performance on statements with different types of validity information, we found that the improvement of TIME2BOX largely attributes to its better ability to handle statements with interval-based validity information. In the future, we will explore how to incorporate spatial scopes of statements into KGE models, such that KBC can benefit from both spatial and temporal scopes of statements.

Chapter 5

A Hyperbolic Embedding Model for Qualitative Spatial and Temporal Reasoning

This chapter concentrates on qualitative temporal reasoning rather than quantitative temporal reasoning. It presents a generic method for qualitative temporal reasoning by drawing insights from theories of time and properties of temporal relations. The proposed method is designed to automatically discover rule chains among relations as disclosed in composition/transitivity tables. Meanwhile, the method is capable of modeling inverse relations, transitivity, symmetricity and asymmetricity of temporal relations. The method is also tested on qualitative spatial reasoning. The evaluation results have show its superiority over conventional symbolic reasoning methods on both tasks of qualitative spatial and temporal reasoning. The chapter demonstrates the effectiveness of subsymbolic methods in qualitative reasoning.

Peer Reviewed Publication	
Title	HyperQuaternionE: A hyperbolic embedding model for qualitative spatial and temporal reasoning
Author	Ling Cai, Krzysztof Janowicz, Rui Zhu, Gengchen Mai, Bo Yan, and Zhangyu Wang
Venue	GeoInformatica
Editors	Shashi Shekhar, Elisa Bertino, and Chris Brunsdon
Publisher	Springer Nature
Pages	In Press
Publication Date	05 September 2022
Copyright	@All Authors 2022
DOI	https://doi.org/10.1007/s10707-022-00469-y

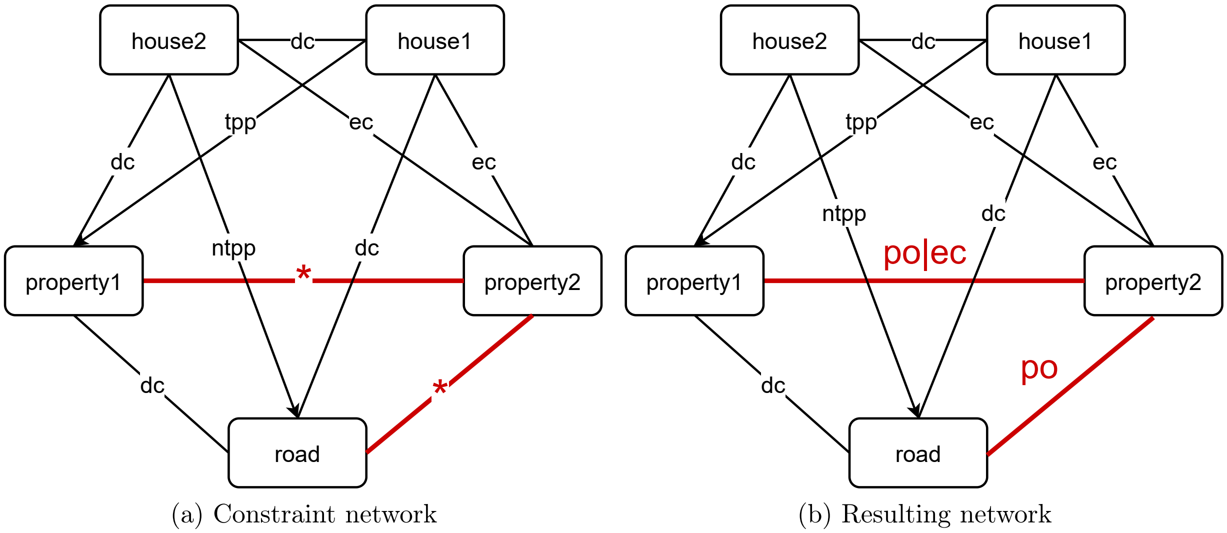
Abstract Qualitative spatial/temporal reasoning (QSR/QTR) plays a key role in research on human cognition, e.g., as it relates to navigation, as well as in work on robotics and artificial intelligence. Although previous work has mainly focused on various spatial and temporal calculi, more recently representation learning techniques such as embedding have been applied to reasoning and inference tasks such as query answering and knowledge base completion. These subsymbolic and learnable representations are well suited for handling noise and efficiency problems that plagued prior work. However, applying embedding techniques to spatial and temporal reasoning has received little attention to date. In this paper, we explore two research questions: (1) How do embedding-based methods perform empirically compared to traditional reasoning methods on QSR/QTR problems? (2) If the embedding-based methods are better, what causes this superiority? In order to answer these questions, we first propose a hyperbolic embedding model, called HyperQuaternionE, to capture varying properties of relations (such as symmetry and anti-symmetry), to learn inversion relations and relation compositions (i.e., composition tables), and to model hierarchical structures over entities induced by transitive relations. We conduct various experiments on two synthetic datasets to demonstrate the advantages of our proposed embedding-based method against existing embedding models

as well as traditional reasoners with respect to entity inference and relation inference. Additionally, our qualitative analysis reveals that our method is able to learn conceptual neighborhoods implicitly. We conclude that the success of our method is attributed to its ability to model composition tables and learn conceptual neighbors, which are among the core building blocks of QSR/QTR.

5.1 Introduction

In our daily life, we humans usually use qualitative expressions, such as *left*, *north*, *after* and *during*, to describe and infer spatial/temporal relations between two objects. The field that studies how to enable machines/artificial intelligence (AI) agents to represent qualitative spatial and temporal expressions, and to draw inferences on top of these representations, namely qualitative spatial/temporal reasoning (QSR/QTR), is an active research topic in AI. In the past years, it has fostered a variety of research across various applications such as cognitive robotics [?], visual sensemaking [?], semantic question answering [?], spatio-temporal data mining [?] and (spatial) cognition and navigation [40, 116].

Since the late 1980s, a plethora of theoretical research have been dedicated to computational QSR/QTR [29, 42, 117, 118, 119, 120, 121, 122, 123, 124]. Among them two best studied fundamental problems in qualitative reasoning (QR) are qualitative knowledge *representation* and *reasoning*. In the past, a lot of work has focused on the knowledge representation aspect. For instance, non-null regions in an n-dimensional embedding space $\mathbb{R}^n$ [125] are taken as ontological primitives, and binary topological relations, i.e., Region Connection Calculus (RCC)-8 relations [42, 126], and Allen's temporal relations [29] as primitive relations between two regions/time intervals. Reasoning, however, remains to be a challenge. Composition tables (CT) and conceptual neighborhood structures (CNS) are among the major reasoning techniques, jointly supporting inferences about spatial and temporal relations between geospatial entities or events [40, 127, 42, 128]. For instance, one can use CT as constraints to reason over spatial relations. Simply put, such a method regards known binary relations as constraints between regions. Then the reasoning task boils down to a consistency satisfactory problem (CSP), i.e., to determine whether the available information is consistent or not, given the CT. For example, as shown in Figure 5.1, the possible topological relation between *property1* and *property2* is either *partially overlap* or *externally connected* after path-consistency checking built up on RCC-8's composition table [129, 130].



(a) Constraint network (b) Resulting network

Figure 5.1: Constraint network-based reasoning. The symbol $*$ in red denotes all RCC-8 relations. Full names of relations are described in Table 5.1. Figure 5.1a illustrates the initial constraints between entities imposed by the relations on edges, and Figure 5.1b shows the resulting relations after path-consistency checking.

Despite those success stories of traditional QR approaches, several limitations remain. First, constraint-based methods are prone to erroneous information, e.g, introduced by noise. Errors may occur at any stage during information collection, and, thus, are inevitable in reality, which may break down the traditional reasoning capabilities. For instance, if the relation between *house2* and *property2* is wrongly recorded as *dc* instead of *ec*, inferring unknown relations based on CT will fail. Second, traditional QR approaches are only applicable to a limited number of reasoning tasks, such as deducing new knowledge, checking consistency, updating existing knowledge, and discovering a minimal set of useful representations. Albeit seemingly different, all these tasks are in fact mutably transformable and can be solved essentially in a similar fashion [131]. Such a shortage of applications is partially attributed to the symbolic knowledge representation used in traditional QR, which prohibits it from being beneficial to other tasks which purely rely on numeric computations. Meanwhile, the symbolic representation of knowledge is usually in the form of triples (i.e., $\langle subject, relation, object\rangle$). Traditional QR approaches only make full use of pairwise constraints between entities while failing to benefit from higher-order interactions. Third, reasoning over spatial/temporal calculi is NP-complete [129], which makes traditional QR methods difficult to scale. Extra efforts (e.g., identifying maximal tractable subsets containing all basic relations and different optimizing strategies) are needed to improve the efficiency, which becomes even more problematic with an increasing number of relations. These limitations, consequently, necessitate more robust spatial/temporal reasoners.

The past decade has witnessed great breakthroughs in Machine Learning (ML). Embedding/sub-symbolic techniques, in particular, have been applied to tackle various reasoning tasks. Examples include word/sentence similarity measuring [132, 133, 81], question/query answering [134, 135, 136, 137], dynamic interaction inference [138], as well as knowledge graph completion and reasoning [14, 15, 16, 17]. Generally speaking, their success can be attributed to learnable sub-symbolic representations (i.e., embeddings) in contrast to symbolic representations. At training, an embedding method is trained to draw patterns of and interactions between entities from data and sub-symbolic representations of entities are learned accordingly. This training process is analogous to knowledge abstraction, which preserves the core essentials of entities but ignores subtle details. Moreover, such a process of automatic abstraction makes embedding models less prone to local errors and data incompleteness, and improve their generability [?, ?].

Despite their appealing characteristics, the adoption of sub-symbolic approaches to QSR/QTR remains mostly unexplored. To fill in this gap, we propose a hyperbolic embedding model, called HyperQuaternionE, as an implicit reasoner for spatial and temporal reasoning. In the model design, we consider the following two prominent characteristics of spatial/temporal reasoning. First, composition tables, which specify role chains of relations, have been the backbone of most qualitative reasoning methods. In order to enable embedding models to automatically find and take use of such role chains, we introduce quaternions, an extension of complex numbers, in the embedding space. Quaternion mutiplication follows the non-commutative law and thus is well suited for modeling relation composition. Additionally, quaternions can be used to model other properties of relations (e.g., symmetric and anti-symmetric) and inverse relations. Second, hierarchical structures over entities must be considered. Certain spatial and temporal relations, such as *non-tangentially proper par* and *before*, are transitive, thus inducing hierarchical structures over entities (e.g., regions or temporal intervals). This suggests that a hyperbolic embedding space, which can embed trees with arbitrarily low distortion [139], would be more appropriate than Euclidean space. Therefore, we adopt hyperbolic space as our embedding space and transfer quaternions to this space to preserve the properties mentioned above. We evaluate our method on two tasks, namely entity inference and relation inference, which are to identify entities that have a given (spatial/temporal) relation (e.g., *partially overlapping*) to a target entity, and to infer the relation held between two given entities, respectively. Finally, we conduct a qualitative analysis over the trained models in order to uncover the reasoning mechanisms behind our model.

The remainder of this paper is structured as follows. Section 5.2 introduces important concepts and terms applied in the proposed method. Section 5.3 summarizes related work on spatial and temporal reasoning, knowledge graph embedding models, and their applications in geospatial knowledge graphs. Section 6.3 elaborates on the motivation of our proposed embedding model and its formulation. To compare the reasoning ability of different models, Section 6.4 presents the datasets, baseline methods, as well as evaluation metrics used in the study, followed by an experimental summary of key findings. Section 6.5 concludes our work and points out future research directions.

5.2 Background

Before reviewing related work, we first introduce concepts and terms used in the literature.

5.2.1 Basic Definitions

Definition 2 (Spatial and Temporal Relations) *In this paper, we focus on the eight topological relations of RCC-8 [42], and the thirteen temporal relations developed by Allen [29]. Table 5.1 and 5.2 list those relations together with their inherent properties (i.e., transitive and symmetric).*

Name (abbrev.)	Transitive	Symmetric
disconnected (dc)	✗	✓
externally connected (ec)	✗	✓
partially overlapping (po)	✗	✓
tangentially proper part (tpp)	✗	✗
tangentially proper part inverse (tppi)	✗	✗
non-tangentially proper part (ntpp)	✓	✗
non-tangentially proper part inverse (ntppi)	✓	✗
equal (eq)	✓	✓

Table 5.1: List of spatial relations

Name (abbrev.)	Transitive	Symmetric	Name (abbrev.)	Transitive	Symmetric
before (<)	✓	✗	after (>)	✓	✗
meets (m)	✗	✗	met-by (mi)	✗	✗
overlaps (o)	✗	✗	overlapped-by (oi)	✗	✗
during (d)	✓	✗	contains (di)	✓	✗
starts (s)	✓	✗	started-by (si)	✓	✗
finishes (f)	✓	✗	finished-by (fi)	✓	✗
equal (=)	✓	✓			

Table 5.2: List of temporal relations

Definition 3 (Knowledge Graphs) *Formally, a Knowledge Graph (KG) can be represented as $G = (V, E)$, where V is the set of nodes/entities and E is the set of edges with labels, denoting relations held between two entities. A statement then consists of a head entity, a relation, and a tail entity, written as $\langle h, r, trlangle$, where $h, t \in V$ and $r = \sigma(e), e \in E$. σ is a mapping function from an edge to its label. One way to represent such a type of knowledge is known as the RDF (Resource Description Framework), a standard mostly used in the Semantic Web literature. We use the term Knowledge Graph here to denote such a set of RDF statements. Naturally, a statement claiming a spatial or temporal relation between two entities (i.e., geometries or temporal intervals) can be represented as a triple. For instance, a statement that geometry A is disconnected to geometry B can be represented as triple (A, dc, B). Note that we use a unified name – spatial KGs (SKGs) – to refer to KGs involving only spatial relations or/and temporal relations.*

Definition 4 (Knowledge Graph Embedding) *Given their symbolic nature, it is difficult to apply RDF-based knowledge graphs directly to applications that require notions such as quantitative measurements of similarity. For instance, most recommender systems are built upon sub-symbolic approaches and it is hard for symbolic KGs to contribute directly. In order to address this limitation, knowledge graph embeddings (KGE) were proposed, which aim at encoding entities and relations of a KG into a high-dimensional* *continuous vector space while preserving the underlying structures. Specifically, a KGE model projects symbolic representations of a head entity and a tail entity – h and t, to points in a continuous vector space – their numeric vector representations, $\mathbf{h}$ and $\mathbf{t}$, respectively. Additionally, it assumes the relation r acts as a transformation operator, transforming $\mathbf{h}$ to $\mathbf{t}$ in this continuous space, such as translation, rotation, etc. Note that we use plain symbols (e.g., h) to denote symbolic representations and the bold format (e.g., $\mathbf{h}$) to denote numeric vector representations.*

Mathematically, the embedding of an entity, or a relation, is mostly formalized as $\mathbf{v} \in \mathbb{R}^d$, or $\mathbf{r} \in \mathbb{R}^d$, in Euclidean space. Trained on symbolic representations of statements presented in KGs, a KGE model is optimized towards minimizing the loss of reproducing those presented statements. More details on embedding models will be reviewed in Section 5.3.

Definition 5 (Entity Inference) *Entity Inference refers to answering queries in which one of the entity in a statement is missing, usually expressed as either $\langle ?h, r, t\rangle$ or $\langle h, r, ?t\rangle$, corresponding to missing head or missing tail entities. A plain text example would be which city is located in California?, or which event occurred during the COVID-19 pandemic?*

Definition 6 (Relation Inference) *Relation Inference refers to inferring the relation between two entities, usually in the form of $\langle h, ?r, t\rangle$. Example queries include: what is the topological relation between Los Angeles to California? and which temporal relation holds between the Bronze Age and Stone Age?*

Definition 7 (Quaternion) *A quaternion q has the form of $q = a + bi + cj + dk$, where $a, b, c, d \in \mathbb{R}$ and a is the real part and bi, cj, dk are three imagery parts. Alternatively, we can express a quaternion as $[a, \mathbf{u}]$, where $\mathbf{u} \in \mathbb{R}^3$, consisting of three imagery components. q is a pure quaternion when $a = 0$.*

It was first introduced in 1843 by Irish mathematician William Rowan Hamilton and applied to mechanics in 3D space. We can view it as a generalization of complex numbers (i.e., $a + bi$) but it contains two more imagery parts. Similar to multiplication over complex numbers, there is a rule for the three imagery units i, j, k: $i^2 = j^2 = k^2 = ijk = -1$. According to polynomial multiplication, the multiplication of two quatertions $q_x = a + bi + cj + dk$ and $q_y = e + fi + gj + hk$ can be calculated as below:

$$q_x q_y = (a + bi + cj + dk) * (e + fi + gj + hk) = \begin{bmatrix} a & -b & -c & -d \\ b & a & -d & c \\ c & d & a & -b \\ d & -c & b & a \end{bmatrix} \begin{bmatrix} e \\ f \\ g \\ h \end{bmatrix} \tag{5.1}$$

According to Eq.5.1, we can easily derive that $q_x q_y \neq q_y q_x$, meaning that quaternion multiplication does not conform to the commutative law. This lays the foundation of modeling asymmetric composition tables for qualitative spatial and temporal reasoning, which will be discussed in Section 6.3.

Important properties and definitions of quaternions are given as below:

1. *Inversion of a quaternion*: $qq^{-1} = q^{-1}q = 1$ $(q \neq 0)$.

2. *Conjugate of a quaternion*: $q^* = a - bi - cj - dk = a - \mathbf{u}$. In addition, $(pq)^* = q^*p^*$.

3. *Norm of a quaternion*: $\|q\| := \sqrt{qq^*} = \sqrt{q^*q} = \sqrt{a^2 + b^2 + c^2 + d^2} = \sqrt{a^2 + \|\mathbf{u}\|^2}$. When $\|q\| = 1$, we call q a unitary quaternion, denoted as q_u.

Because $qq^* = q^*q = \|q\|^2$, one way of deriving quaternion inverse is $q^{-1} = \frac{q*}{\|q\|^2}$. In particular, when q is a unitary quaternion, $q^{-1} = q^*$.

Definition 8 (Hyperbolic Space) *Hyperbolic space is a homogeneous space which exhibits hyperbolic geometry with a constant negative sectional curvature.*

There are different hyperbolic models to describe hyperbolic space mathematically, such as the Poincaré plane model [140] and the hyperboloid model (the Lorentz model) [141]. Here, we introduce the Poincaré ball model, which is the generalization of the Poincaré plane model. Mathematically, a d-dimensional Poincaré ball of radius $\frac{1}{\sqrt{c}}$ $(c > 0)$ can be expressed as $\mathbb{B}_c^d = \{\mathbf{x} \in \mathbb{R}^d : c\|\mathbf{x}\|^2 < 1\}$, where $\| \cdot \|$ is the Euclidean norm. Such a ball has a negative curvature $-c$, and with a larger c, the space is more curved. Note that Euclidean space has a curvature of zero, corresponding to $c = 0$, and spherical space has a constant positive curvature. When $c = 1$, the distance between two points in the hyperbolic space is given by:

$$d_H(\mathbf{x}, \mathbf{y}) = arcosh(1 + 2\frac{\|\mathbf{x} - \mathbf{y}\|^2}{(1 - \|\mathbf{x}\|^2)(1 - \|\mathbf{y}\|^2)}) \tag{5.2}$$

where $\| \cdot \|$ is the Euclidean norm.

This formula provides a desirable property that allows hyperbolic space to embed trees/hierarchical data. According to this formula, we can observe that when a point is close to the origin (i.e., $\|\mathbf{x}\| \approx 0$), the distance between it and any other point will be smaller. Conversely, as points move towards the boundary of the ball (e.g., $\|\mathbf{x}\| \approx 1$), the distance will be larger and the distance $d_H(\mathbf{x}, \mathbf{y})$ between two points approaches $d_H(\mathbf{x}, 0) + d_H(0, \mathbf{y})$. Also, as points move away from the root/origin, more "space" is available to separate points (e.g., nodes in a tree) in hyperbolic space. This is analogous to the shortest distance between two sibling nodes in a tree, which is equal to the length of the path through their parent. This means hyperbolic distance exhibits a desirable resemblance to tree metrics. Figure 5.3 illustrates how a tree-like 2D embedding space looks like.

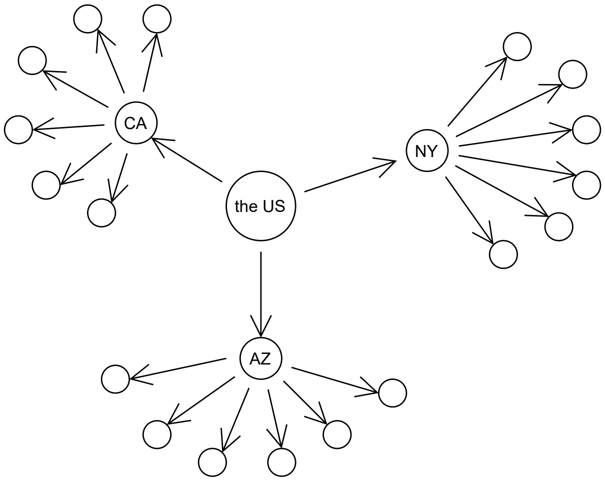


Figure 5.2: Example of a hierarchical tree. This tree is induced by ntppi (non-tangentially proper part inverse) relation, which means a preceding entity has a nttpi relation to its succeeding entities in this tree.

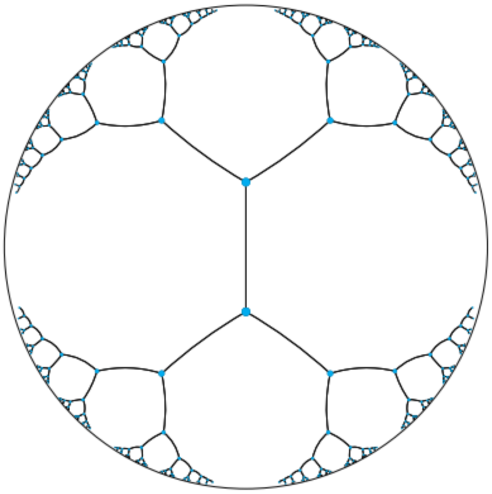


Figure 5.3: Illustration of embedding a hierarchical tree (with two being the branching factor) into a 2D hyperbolic plane. Distances between any two directly connected points (in blue) are equal and distances grow exponentially when approaching to the edge of the plane. (source from [142])

5.3 Related Work

A plethora of Knowledge Graph Embedding (KGE) models have been developed in the past decade. Relations in KGs have different properties, such as symmetry, antisymmetry, inversion, and transitivity [65]. Different models preserve varying properties due to distinct ways of manipulating relations. Accordingly, we roughly divide them into four groups – translation, rotation, mixed manner, and others. Particularly, this group focuses on which properties of relations (e.g., symmetric and inverse) are preserved and whether the model is able to encode relation composition by design. Last but not least, we review related work on hyperbolic embeddings, which sheds lights on modeling hierarchical relations.

Relations as Translation The most representative KGE model is TransE [62]. It assumes that for a statement $\langle h, r, t \rangle$, $\mathbf{t}$ is resulted from $\mathbf{h}$ being translated by $\mathbf{r}$ in a vector space. Translation operation in a *real* vector space can be easily achieved by vector addition, and thus the idea of TransE is formalized as $\mathbf{h} + \mathbf{r} = \mathbf{t}$. A number of variants were proposed subsequently to address issues with the original TransE. For example, TransH argued that TransE cannot deal with other types of relations except for 1-to-1 relation type, and, thus, introduced relation-aware hyperplanes [11]. TranSparse introduced adaptive sparse matrices to address the heterogeneity and imbalance issues of entities and relations in KGs [143]. This group of methods is simple yet very effective, and lays the foundation of most KGE methods. However, they fail to encode simple properties of relations and logic patterns. For instance, they cannot model symmetric property of relations. If relation r is symmetric, both $\mathbf{h}+\mathbf{r} = \mathbf{t}$ and $\mathbf{t}+\mathbf{r} = \mathbf{h}$ should hold according to TransE, which leads $\mathbf{r}$ to be close to $\mathbf{0}$. Additionally, although TransE is able to achieve relation composition, the order of relations is not considered. Namely, it presumes that $r1 \circ r2 = r2 \circ r1$. Therefore, TransE ignores the **non-commutativity** law in relation composition, which causes issues in modeling role chains in composition tables for spatial and temporal reasoning. Moreover, TransE cannot deal with **hierarchical relations** either.

Relations as Rotation One seminal example in this group is RotatE, which assumes that a relation acts as a rotation in 2D space and encodes a relation as a unit complex vector [65]. Similar to TransE in the *real* space, RotatE can be formalized as $\|\mathbf{h} \otimes \mathbf{r} - \mathbf{t}\| = 0$, where $\otimes$ is the vector multiplication in the *complex* space instead. RotatE by design succeeds in modeling multiple logic patterns, such as symmetry, anti-symmetry, inversion, and relation composition. However, it is incapable of dealing with the order of relations in composition, either. Recently, due to the non-commutative law of quaternion multiplication, quaternions, which have two more imaginary elements than complex numbers, have been introduced to address this issue. RotatE3D assumes that a tail entity is resulted from a head entity being rotated by $\mathbf{r} in 3D$ [144]. Despite its effectiveness in capturing various logic patterns, it falls short of modeling hierarchical relations from transitive relations. Such relations are in fact prominent in spatial and temporal reasoning since most spatial/temporal relations are transitive. In this paper, we also make use of quaternions to capture additonal logic patterns and extend it to hyperbolic space in order to encode hierarchical structures.

Relations as Mixed Operators Recently, [145] argue that existing work considers the relation to be either a translation or rotation operator but not both, thus limiting the representational and inferring ability of sub-symbolic models. Hence, they introduce dual quaternions to represent relations, which embrace the properties of translation and rotation simultaneously. Despite its intuitive physical and geometric interpretations, the unified framework do not improve significantly on data sets that encode hierarchical hypernym relations, such as *specific type of*.

Other Methods Another track of studies are based on tensor factorization, such as DistMult [66] and RESCAL [67] in *real* space and ComplEx [146] and TNTComplEx [147] in *complex* space. This type of methods measures the compatibility score of two entities and a relation in a statement. For example, DistMult defines the score as the result of $\mathbf{h} \odot \mathbf{r} * \mathbf{t}^T$, where $\odot$ is the element-wise vector multiplication and $*$ the dot product. Such methods do not have intuitive geometric interpretations and often fail to capture logic patterns as well as properties of spatial/temporal relations.

Hyperbolic Embeddings All the aforementioned methods are not effective in modeling hierarchical data, since their embeddings are built in Euclidean space. Recent embedding methods based on hyperbolic geometry exhibit promising results when modeling parsimonious and taxonomic patterns in data, since hyperbolic geometry is natural to model tree-like structures with low distortion [139, 148, 149, 150, 151]. Specifically, as a counterpart to TransE in the hyperbolic space, MuRP, was proposed by [150] to handle hierarchical data in KGs. It achieves remarkable performance with fewer parameters than TransE. However, MuRP faces the same issues as TransE does since they both conform to the translation assumption. In order to encode various logic patterns and to preserve other properties of relations, [152] proposed to combine hyperbolic rotation and reflection with attention. While substantial improvements are observed, this method mainly focuses on anti-symmetric and symmetric relations. On the contrary, our paper aims at taking a broader range of relation properties (e.g., symmetric and anti-symmetric), inverse relation, and relation composition (i.e., role chains in composition tables) into account when designing an embedding model for QSR/QTR.

5.4 HyperQuaternionE

In this section, we first introduce the motivation of the proposed embedding model and then formulate the idea mathematically.

5.4.1 Motivation

Composition tables, which specify role chains of relations[1], have been widely used in traditional qualitative spatial and temporal reasoning methods, and are identified as one of the key reasoning techniques [40, 127, 42]. *An embedding method should also be able to automatically find and take full use of such role chains in its inference and reasoning.* One

[1] For instance, if entity A is non-tangential proper part of entity B and entity B is externally connected to entity C, then entity A must be disconnected to entity C.

core requirement for such an embedding method is to model *asymmetric* role chains in composition tables; namely $r1 \circ r2 \neq r2 \circ r1$, where $\circ$ denotes the composition operation. For example, if we know geographic entity A is *disconnected* to geographic entity B and B is *tangential proper part* of geographic entity C, the relation of A to C will fall into one of five possible relations, i.e., *dc*, *ec*, *po*, *tpp* or *ntpp* according to the composition table. By contrast, if we first know A is *tangential proper part* of B and B is *disconnected* to C, then the relation of A to C must be *disconnected.* This means the order of relations in role chains matters. In order to take this into account, we use quaternions, an extension of complex numbers, to automatically capture role chains from training data, thanks to the non-commutative law of quaternion multiplication. Additionally, quaternions can be readily used to model varying properties of relations (e.g., symmetric and anti-symmetric relations) and inverse relations, which further contributes to inference and reasoning over spatial and temporal information.

In addition to the need of capturing role chains in composition tables, we notice that 3/8 spatial relations in RCC8, and 9/13 temporal relations in Allen's temporal intervals [29] are transitive (see Table 5.1 and 5.2). Geometrically, transitive relations usually induce tree-like structures over entities, in which as the depth of a tree increases, the number of child nodes grows exponentially. As shown in Figure 5.2, as the root – the US, branches out, more and more child nodes emerge. Also, although some relations (such as *tpp* and *tppi*) are not transitive, they may still induce a tree-like structure over entities to some degree. *Thereby, an embedding method for spatial and temporal reasoning should be built on a suitable embedding space, which is able to encode non-Euclidean structures exhibited in data (e.g., hierarchies).* Past works have demonstrated that hyperbolic embeddings are more suitable for data exhibiting non-Euclidean geometric properties, such as hierarchy [139]. This is because hyperbolic space can be naturally viewed as a continuous analogy to hierarchical trees in discrete space and it grows exponentially with an increasing radius, which corresponds to an exponential increase in the number of child nodes with increasing tree depth [150]. Therefore, given the abundance of transitive spatial/temporal relations, we embed entities and relations in hyperbolic space rather than Euclidean space.

Despite the aforementioned advantages of quaternions and hyperbolic space, the technical bottleneck of the model design rests on how to harmonize quaternions and hyperbolic space while preserving their respective properties. The transformation of quaternions, which are originally defined in Euclidean space, into a hyperbolic space is not trivial, since quaternion-related vector operations (e.g., vector addition, matrix-vector multiplication, and vector inner product over quaternions) and geometric metrics (e.g., the closed form of distance) in Euclidean space is hard to be generalized to hyperbolic space.

In this paper, we propose a hyperbolic embedding model, called HyperQuaternionE, in which this challenge is tackled. In the following, we will first introduce preliminary concepts and notations, then propose our model, and finally analyze which relation properties and composition patterns our model can preserve.

5.4.2 Preliminaries

Quaternion Multiplication and 3D Rotation As mentioned above, one significant advantage of using quaternions in KGE models lies in the ability of quaternions to model asymmetric role chains in composition tables; namely $r1 \circ r2 \neq r2 \circ r1$. This is guaranteed by the non-commutative law of quaternion multiplication (Definition 7). Here, we give a geometrical interpretation by contrasting the role of complex numbers in 2D rotation and that of quaternions in 3D rotation. In 2D space (see RotatE [65]), a 2D rotation can be achieved by the multiplication of a complex number (i.e., a 2D vector to be rotated) and a unitary complex number (i.e., the rotating angle). The rotation direction is either clockwise or counter-clockwise, and the rotation is around the origin. Thus, the order of two consecutive rotations does not make a difference to the resulting vector. That is, the result of rotating a vector by θ_1 first and then by θ_2 is the same as that of rotating the same vector by θ_2 first and then by θ_1; both equal to rotating a vector by an angle of $\theta_1 + \theta_2$ at the end. By contrast, quaternions are related to rotations in 3D space, which are originally used in computer graphics [153, 154]. Any point in 3D space in the form of vectors can be expressed as a pure quaternion, and 3D rotation as quaternion multiplication over a pure quaternion (i.e., the point to be rotated) and a unitary quaternion (i.e., the rotation). Unlike rotations in 2D space, where a vector is always rotated around the origin, each 3D rotation specifies a distinct rotating axis and a rotating angle. That is, rotating results are determined by both rotation axes and angles. As such, the result of performing several 3D rotations over a vector consecutively differs from that of performing the same 3D rotations in another order.

Mathematically, 3D rotations can be formalized as Eq.1. We denote the 3D point ($\mathbf{v} \in \mathbb{R}^3$) to be rotated as a pure quaternion $v = [0, \mathbf{v}]$, a unitary quaternion $q_u = [cos(\theta), sin(\theta)\mathbf{u}]$ (θ is the rotating angle and $\mathbf{u}$ is the rotating axis) as the rotating vector and the resulting point as $v' = [0, \mathbf{v}']$ ($\mathbf{v}' \in \mathbb{R}^3$).

Theorem 1 (Euler-Rodrigues-Hamilton Formula [155]) *Any rotation in 3D space can be derived by quaternion multiplication. The result of rotating a 3D point* $\mathbf{v}$ *by an angle of* θ *around a unit axis* $\mathbf{u}$ *(i.e.,* q_u*) can be expressed as follows:*

$$v' = v_{\|} + q_u v_{\perp} = p_u v p_u^{-1} = p_u v p_u^{*} \tag{5.3}$$

where $v_{\|}$ is the component of $\mathbf{v}$ parallel to $\mathbf{u}$ and $v_{\perp}$ the component of $\mathbf{v}$ perpendicular to

$\mathbf{u}$. $q_u = p_u^2$ and $p_u = [cos(\frac{\theta}{2}), sin(\frac{\theta}{2})\mathbf{u}]$. This theorem can be interpreted as the component of $\mathbf{v}$ perpendicular to $\mathbf{u}$ is rotated twice by $\frac{\theta}{2}$ around $\mathbf{u}$. Proofs to this theorem can be found in [156, 155].

Theorem 2 *Product of two unit quaternions is still a unit quaternion.*

Proof. Let p and q be two arbitrary quaternions. According to Property 2 in Definition 7, $\|pq\| = \sqrt{pq(pq)^*} = \sqrt{pqq^*p^*} = \sqrt{p(qq^*)p^*} = \sqrt{pp^*}\sqrt{qq^*} = \|p\|\|q\|$. Thus when p and q are unit quaternions; namely $\|p\| = \|q\| = 1$, $\|pq\| = 1$, i.e., pq is a unitary quaternion. This property ensures that a number of consecutive rotations can be replaced by a single rotation, which is fundamental to the modeling of relation composition.

Poincaré Ball Model. Similar to [150] and [152], this work uses a d-dimensional Poincaré ball model to form the hyperbolic embedding space for embedding tree-like structures (Definition 5.2.1). Reasons for choosing such a model are two-fold. It provides convenient communication between hyperbolic space and Euclidean space via exponential and logarithmic maps [148], thus making it relatively easy to incorporate quaternions rooted in Euclidean space to hyperbolic space. Moreover, it is well-suited for gradient-based optimization methods (see Section 5.4.2).

When c is considered, the hyperbolic distance of two points $\mathbf{x}, \mathbf{y} \in \mathbb{B}_c^d$ is defined as its geodesic distance in the space, which has the desirable property of forming a tree-like embedding space (see Figure 5.3). It is formulated as follows:

$$d^c(\mathbf{x}, \mathbf{y}) = \frac{2}{\sqrt{c}} arctanh(\sqrt{c}\ \|(-\mathbf{x}) \oplus_c \mathbf{y}\|) \tag{5.4}$$

where $arctanh(\cdot)$ denotes the inverse hyperbolic tangent. The *Möbius addition* (i.e., $\oplus_c$)

of two points $\mathbf{x}, \mathbf{y} \in \mathbb{B}_c^d$ can be expressed as below:

$$\mathbf{x} \oplus_c \mathbf{y} = \frac{(1 + 2c\mathbf{x}^T\mathbf{y} + c\|\mathbf{y}\|^2)\mathbf{x} + (1 - c\|\mathbf{x}\|^2)\mathbf{y}}{1 + 2c\mathbf{x}^T\mathbf{y} + c^2\|\mathbf{x}\|^2\|\mathbf{y}\|^2} \tag{5.5}$$

where $\|\cdot\|$ is the Euclidean norm. We can obtain that $\mathbf{x} \oplus_c (-\mathbf{x}) = (-\mathbf{x}) \oplus_c \mathbf{x} = \mathbf{0}$. This property helps model inverse relations in the embedding space.

Bridging Quaternion and Hyperbolic Space

Exponential Map and Logarithmic Map. As mentioned in Section 5.4.1, the difficulty of model design lies in how to simultaneously preserve inherent properties from both hyperbolic space and quaternions that are well-studied in Euclidean space. In this paper, instead of directly generalizing möbius transformation as well as Poincaré distance with quaternion entries [157], we adopt a simple strategy by introducing exponential and logarithmic maps [148], which bridges between tangent space (which sits in Euclidean space) and hyperbolic space. By doing so, we can perform quaternion operations in tangent space while measuring hyperbolic distance in hyperbolic space.

For a point $\mathbf{x} \in \mathbb{B}_c^d$, its tangent space representation $(\mathbf{x}^E)$ is defined as a d-dimensional vector, which approximates the hyperbolic space $\mathbb{B}_c^d$ around $\mathbf{x}$ (origin). The two mappings $(\exp_0^c(\cdot)$ and $\log_0^c(\cdot))$ at the origin have the following closed-form expressions:

$$\exp_0^c(\mathbf{x}^E) = \tanh(\sqrt{c}\|\mathbf{x}^E\|)\frac{\mathbf{x}^E}{\sqrt{c}\|\mathbf{x}^E\|} = \mathbf{x}^H \tag{5.6}$$

$$\log_0^c(\mathbf{x}^H) = \operatorname{arctanh}(\sqrt{c}\|\mathbf{x}^H\|)\frac{\mathbf{x}^H}{\sqrt{c}\|\mathbf{x}^H\|} = \mathbf{x}^E \tag{5.7}$$

where $\exp_0^c(\cdot)$ maps $\mathbf{x}^E$ in the tangent space to $\mathbb{B}_c^d$ and conversely, $\log_0^c(\cdot)$ maps $\mathbf{x}^H$ in $\mathbb{B}_c^d$ to the tangent space. Note that we use $\mathbf{x}^H$ to denote $\mathbf{x}$ in the hyperbolic space while $\mathbf{x}^E$

being in Euclidean space.

5.4.3 Model Formulation

The core idea behind the proposed HyperQuaternionE is to encode relations as 3D rotations, and assumes that for a triple $\langle h, r, t\rangle$, the tail entity t is the result of the head entity h being rotated by relation r. This indicates two key steps in our method: rotating the head entity by the relation and measuring the distance between the tail entity and the head entity after being rotated. Despite being similar to the rotation family introduced in Section 5.3, the main difference is that in our method these two steps are performed in different spaces. The rotating step is performed in the tangent space with the aim to use quaternions in order to capture role chains from data, and the distance measuring step is executed in hyperbolic space so as to form a tree-like embedding space for hierarchical data. Mathematically, for a triple $\langle h, r, t\rangle$ in a KG, these two steps can be formalized as follows. Note that for entities and relations, their embeddings are first randomly initialized, denoted as $\mathbf{h}^E, \mathbf{r}^E, \mathbf{t}^E \in \mathbb{R}^d$ (d is the dimension), and are learned automatically through training.

In the first step, a 3D rotation on the head entity h performed by relation r is achieved by Theorem 1. Concretely, head entities are modeled as 3D points to be rotated, and tail entities are modeled as results of head entities being rotated by relations (i.e., 3D rotation). In order to utilize quaternions to implement 3D rotation, we convert real value entries in $\mathbf{h}^E$ and $\mathbf{r}^E$ into quaternions. Hence each head embedding $\mathbf{h}^E \in \mathbb{R}^d$ can be expressed as $\frac{d}{3}$ pure quaternions. Specifically, it can be written as $V_h^E = [h_1, h_2, ..., h_i]^T$, where $h_i = [0, \mathbf{h}_i]$ is a pure quaternion and $\mathbf{h}_i \in \mathbb{R}^3$ $(i \in \{1, 2, ..., \frac{d}{3}\})$ denotes a 3D point. Similarly, each relation is represented by $\frac{d}{3}$ unitary quaternions, whose embedding can be written as $Q_r^E = [q_{r,1}, q_{r,2}, ..., q_{r,\frac{d}{3}}]^T$, where each $q_{r,i}$ $(i \in \{1, 2, ..., \frac{d}{3}\})$ is a unitary quaternion. According to Eq. 5.3, 3D rotation in the embedding space is given as follows:

$$\mathbf{Rot}_{hr,4}^{E} = \mathrm{Rot3D}(\mathbf{h}^{E}, \mathbf{r}^{E}) = Q_r^E \circledcirc V_h^E \circledcirc (Q_r^E)^* \tag{5.8}$$

$$\mathbf{Rot}_{hr}^{E} = concat(\mathbf{Rot}_{hr,4}^{E}) \tag{5.9}$$

where $\circledcirc$ denotes element-wise quaternion multiplication and $(Q_r^E)^* = [h_1^*, h_2^*, ..., h_i^*]^T$ denotes the conjugate of Q_r^E. $\mathbf{Rot}_{hr,4}^{E}$ is the rotating result of the head entity and contains $\frac{d}{3}$ pure quaternions. $concat(\cdot)$ is to concatenate three imagery components of these pure quaternions in order to recover the original dimension d.

In the second step, to form a tree-like embedding space for hierarchical data, we measure the distance between the resulting head embedding and the tail embedding in hyperbolic space. Since the first step is performed in tangent spaces, we first map Euclidean embeddings into hyperbolic embeddings via exponential maps shown in Eq. 5.6. However, rather than using a generic curvature c, a relation-aware learnable curvature c_r is introduced for each relation because relations of different kinds may yield hierarchical structures of varying degrees. For example, a graph where only the relation *tangential proper part* holds between entities would have a higher hierarchy index than the one induced by the relation *disconnected*. The relation-aware exponential maps are shown below.

$$\mathbf{Rot}_{hr}^{H} = \exp_0^{c_r}(\mathbf{Rot}_{hr}^{E}) = \tanh(\sqrt{c_r}\|\mathbf{Rot}_{hr}^{E}\|)\frac{\mathbf{Rot}_{hr}^{E}}{\sqrt{c_r}\|\mathbf{Rot}_{hr}^{E}\|} \tag{5.10}$$

$$\mathbf{t}^{H} = \exp_0^{c_r}(\mathbf{t}^{E}) = \tanh(\sqrt{c_r}\|\mathbf{t}^{E}\|)\frac{\mathbf{t}^{E}}{\sqrt{c_r}\|\mathbf{t}^{E}\|} \tag{5.11}$$

where $\mathbf{Rot}_{hr}^{H}$ and $\mathbf{t}^{H}$ are embeddings of $\mathbf{Rot}_{hr}^{E}$ and $\mathbf{t}^{E}$ in hyperbolic space, respectively.

Finally, the distance is calculated by using the following formula:

$$d^{c_r}(\mathbf{Rot}^{H}_{hr}, \mathbf{t}^{H}) = \frac{2}{\sqrt{c_r}} arctanh(\sqrt{c_r}\ \|(-\mathbf{Rot}^{H}_{hr}) \oplus_{c_r} \mathbf{t}^{H}\|) \tag{5.12}$$

Eq. 5.12 is originated from Eq. 5.4, but contains a relation-aware learnable curvature c_r to consider the difference of embedding spaces induced by various relations.

Similar to previous work [66, 65], we optimize the model by minimizing the distance between $\mathbf{Rot}^{H}_{hr}$ and a valid tail t (meaning that $\langle h, r, t\rangle$ exists in our KG) and maximizing that to a negative tail. More specifically, for a triple $\langle h, r, t\rangle$ in a KG, t itself is a positive tail and we construct negative tails by replacing t with another entity (i.e., t'), which is randomly picked from all other entities. It is done by n times in order to obtain n negative tails. Finally, the optimizer is to pull the correct t towards $\mathbf{Rot}^{H}_{hr}$ as close as possible while pushing negative ones far away, which can be formalized as:

$$\mathcal{L} = -log\ \sigma(\gamma - d^{c_r}(\mathbf{Rot}^{H}_{hr}, \mathbf{t}^{H})) - \frac{1}{n}\sum_{i=1}^{n} log\ \sigma(d^{c_r}(\mathbf{Rot}^{H}_{hr}, \mathbf{t'_i}^{H})) - \gamma) \tag{5.13}$$

where σ denotes the sigmoid function and γ is a hyper-parameter indicating the tolerance of distance between the positive/negative and the resulting entity embedding.

Likewise, with regard to relation inference, for each positive triple $\langle h, r, t\rangle$, we corrupt it by replacing r with other (spatial/temporal) relations n_r times so as to generate n_r relation-based negative samples. To consider both tasks, we construct a joint loss function and use a scalar β to adjust their respective contributions:

$$\mathcal{L}' = \mathcal{L} - \beta\frac{1}{n_r}\sum_{i=1}^{n_r} log\ \sigma(d^{c_{r_i}}(\mathbf{Rot}^{H}_{hr_i}, \mathbf{t}^{H})) - \gamma) \tag{5.14}$$

Last but not least, we introduce a way of representing relations such that they can be ensured to be unitary quaternions. This is of great importance to achieve 3D rotations based on Theorem 1. Recall that only three values are needed to determine a unitary quaternion. So for any three arbitrary values $\alpha, \theta_1, \theta_2 \in [-\pi, \pi]$, a unitary quaternion can be constructed as follows:

$$q_u = cos(\alpha) + sin(\alpha)cos(\theta_1)cos(\theta_2)i + sin(\alpha)cos(\theta_1)sin(\theta_2)j + sin(\alpha)sin(\theta_1)k \tag{5.15}$$

Based on the definition of quaternion norm (see Property 3), $\|q_u\| = 1$ can be readily ensured (See Appendix B.1.1 for proofs). In what follows, we analyze relation properties and composition patterns that are preserved by using the proposed model.

Lemma 3 *HyperQuaternionE can model symmetric/anti-symmetric properties of relations.*

If $\langle h, r, t\rangle$ and $\langle t, r, h\rangle$hold, according to Theorem 1, in the tangent space for each rotation we have:

$$\mathbf{t}_i^E = q_{r,i}\mathbf{h}_i^E q_{r,i}^* \tag{5.16}$$

$$\mathbf{h}_i^E = q_{r,i}\mathbf{t}_i^E q_{r,i}^* \tag{5.17}$$

Thus, when we plug Eq. 5.16 into Eq. 5.17, it yields:

$$\mathbf{h}_i^E = q_{r,i}(q_{r,i}\mathbf{h}_i^E q_{r,i}^*)q_{r,i}^* = q_{r,i}^2\mathbf{h}_i^E(q_{r,i}^*)^2 \tag{5.18}$$

The correspondence of $\mathbf{h}_i^E$ in hyperbolic space is given by Eq. 5.6:

$$\mathbf{h}_i^H = \tanh(\sqrt{c_r}\|\mathbf{h}_i^E\|)\frac{\mathbf{h}_i^E}{\sqrt{c_r}\|\mathbf{h}_i^E\|} \tag{5.19}$$

When we substitute $\mathbf{h}_i^E$ in Eq.5.19 with Eq.5.18, we obtain the following:

$$\begin{aligned}
\mathbf{h}_i^H &= \tanh(\sqrt{c_r}\|\mathbf{h}_i^E\|)\frac{q_{r,i}^2\mathbf{h}_i^E(q_{r,i}^*)^2}{\sqrt{c_r}\|\mathbf{h}_i^E\|} \\
&= q_{r,i}^2\frac{\tanh(\sqrt{c_r}\|\mathbf{h}_i^E\|)\mathbf{h}_i^E}{\sqrt{c_r}\|\mathbf{h}_i^E\|}(q_{r,i}^*)^2 \\
&= q_{r,i}^2\mathbf{h}_i^H(q_{r,i}^*)^2 \\
&\Leftrightarrow q_{r,i}^2 = \pm 1
\end{aligned}$$

It indicates that the sufficient and necessary condition of modeling symmetric relations is that $q_{r,i}^2 = \pm 1$ holds. Clearly, in 3D space, a rotation angle of $k * 180°$ ($k \in \{1, 3, 5, ...\}$) satisfies this condition. Likewise, we can derive that $q_{r,i}^2 \neq \pm 1$ is the sufficient and necessary condition for modeling anti-symmetric relations.

Lemma 4 *HyperQuaternionE can model inversion of relations.*

If $\langle h, r1, t\rangle$ and $\langle t, r2, h\rangle$ hold, similarly, according to Theorem 1, in the tangent space for each rotation we have:

$$\mathbf{t}_i^E = q_{r1,i}\mathbf{h}_i^E q_{r1,i}^* \tag{5.20}$$

$$\mathbf{h}_i^E = q_{r2,i}\mathbf{t}_i^E q_{r2,i}^* \tag{5.21}$$

The correspondence of $\mathbf{h}_i^E$ in hyperbolic space is given by Eq. 5.6:

$$\mathbf{h}_i^H = \tanh(\sqrt{c_{r2}}\|\mathbf{h}_i^E\|)\frac{\mathbf{h}_i^E}{\sqrt{c_{r2}}\|\mathbf{h}_i^E\|} \tag{5.22}$$

Then, we can obtain:

$$\begin{aligned}\mathbf{h}_i^H &= \tanh(\sqrt{c_{r2}}\|\mathbf{h}_i^E\|)\frac{(q_{r2,i}q_{r1,i})\mathbf{h}_i^E(q_{r2,i}q_{r1,i})^*}{\sqrt{c_{r2}}\|\mathbf{h}_i^E\|} \\ &= (q_{r2,i}q_{r1,i})\frac{\tanh(\sqrt{c_{r2}}\|\mathbf{h}_i^E\|)\mathbf{h}_i^E}{\sqrt{c_{r2}}\|\mathbf{h}_i^E\|}(q_{r2,i}q_{r1,i})^* \\ &= (q_{r2,i}q_{r1,i})\mathbf{h}_i^H(q_{r2,i}q_{r1,i})^* \\ &\Rightarrow q_{r2,i} = \pm q_{r1,i}^*\end{aligned}$$

Clearly, this equation can have multiple solutions. For instance, for a relation $r1$ with its quaternion representation in a dimension being $q_{r1,i} = [\alpha_1, \mathbf{v}_1]$, it inverse relation $r2$ at the same dimension can be constructed as $q_{r2,i} = [\alpha_1, -\mathbf{v}_1]$ or $q_{r2,i} = [-\alpha_1, \mathbf{v}_1]$.

Lemma 5 *HyperQuaternionE can capture non-commutative patterns of relation composition. In special cases, HyperQuaternionE can model commutative patterns.*

Non-commutative composition of relations implies that $r1 \circ r2 \neq r2 \circ r1$ while commutative composition indicates that $r1 \circ r2 = r2 \circ r1$. Here $\circ$ refers to quaternion multiplication. According to Theorem 5.4.2, $r1 \circ r2$ yields another relation $r3$, namely $r1 \circ r2 = r3$, and likewise $r2 \circ r1 = r4$. Due to the non-commutative law of quaternion multiplication (see Eq.5.1), $r3 \neq r4$ can be naturally guaranteed. On the other hand, in special cases, for example, when $r1$ and $r2$ share the same rotating axis, we can conclude that $r1 \circ r2 = r2 \circ r1 = r3 = r4$ (i.e., commutative composition).

Table 5.3 summarizes varying properties of relations and patterns of relation composition that different models can preserve. As can be seen, the proposed HuperQuaternionE achieves all.

		TransE [62]	RotatE [65]	Rotate3D [144]	HyperRotatE [152]	HyperQuaternionE
Property	Symmetric	✗	✓	✓	✓	✓
	Anti-symmetric	✓	✓	✓	✓	✓
	Inversion	✓	✓	✓	-	✓
Composition	commutative	✓	✓	✓	-	✓
	non-commutative	✗	✗	✓	-	✓
Hierarchy	induced by transitive relations	✗	✗	✗	✓	✓

*Note that - means inapplicable.

Table 5.3: Varying properties and patterns modeled by differing models

5.5 Experiments

In this section, we introduce the experimental data and baseline methods. Plus, experimental results are reported quantitatively and qualitatively.

5.5.1 Data Preparation

We synthesize two datasets –region187 and interval205 for spatial reasoning and temporal reasoning, respectively. Both datasets are generated from randomly generated rectangular regions and intervals. For region187, we first generate 200 pairs of points. Each pair is used to represent the top left and bottom right corners of a rectangle. We further filter out invalid cases (e.g., the top left and the bottom right points share the same x/y value). Then we calculate the spatial (topological) relation between any two rectangles based on their geometries and organized them as triples (e.g., (rectangle 1, *dc*, rectangle 2)). Additionally, we sample 5 rectangles to establish more *eq* relations since it is relatively rare to yield the same rectangles from the previous step. A similar process is adopted to generate interval205. Finally, we randomly split both datasets into training (70%), validation(15%), and testing sets (15%). Table 5.4 describes statistics of the two datasets.

Dataset	#entities	#relations	#train	#valid	#test
region187	187	8	24,460	5,241	5,243
interval205	205	13	29,399	6300	6301

Table 5.4: Statistics of region187 and interval205

5.5.2 Baseline Methods

Our model is compared with four baselines: three embedding models and one traditional method used in spatial and temporal reasoning. The three embedding methods (i.e., RotatE, QuaternionE/Rotate3D, HyperRotatE/RotH) are chose upon Table 5.3[2]. All these models are unified in the same framework and thus adopt the same protocols for data processing, training, as well as evaluation. Experimental data along with the framework is openly shared for reproducibility and replicability[3].

Traditional methods are built upon path consistency checking over a constraint network, where nodes represent entities (e.g., rectangles or intervals in this paper) and edges are labelled with a set of possible relations between entities [29]. By propagating temporal/spatial composition tables over the network [158], this network will be refined as the relations between entities that do not conform to composition tables will be ruled out. Similarly, in our experiment, we construct a network by using training and testing datasets, where relations in the testing set all are changed to be a set of all possible relations in the beginning (namely eight relations for spatial reasoning and thirteen relations for temporal reasoning). Through propagation, relations that lead to inconsistency will be discarded and the remaining relations are viewed as inference results. Figure 5.1 gives an illustrative interpretation. We name this method as constraint network method and use an open-sourced package to implement it[4].

[2] We omited TransE here, since its performance is relatively weak.

[3] we will release the github repository after the blind review process

[4] `https://github.com/alreich/qualreas`

5.5.3 Experimental Settings

In order to achieve a fair comparison, we ensure that all compared models share approximately the same number of parameters. The number of learnable parameters used in each model is shown in Appendix B.2. Similar to [152], we carry out two experimental settings – low-dimensional and high-dimensional. More details on the number of parameters as well as the best parameter setting are shown in Table 5.5. Note that four hyper-parameters are chose from various ranges: learning rate – lr:[0.05, 0.1], margin in Eq.6.1 – γ:[8, 10, 12], batch size – b :[512, 1024] and negative samples – n: [8, 16, 32, 64]. For the weighting parameter β in Eq. 6.1, we set it as 0.5 empirically.

Models on region187	low-dimensional	high-dimensional
HyperQuaternionE	lr0.1-b512-g8-n8-h30 (5,858)	lr0.05-b1024-g12-n8-h120 (23,408)
HyperRotatE	lr0.1-b512-g0-n8-h26 (5,681)	lr0.05-b1024-g0-n64-d110 (23,405)
QuaternionE	lr0.1-b1024-g12-n8-h30 (5,850)	lr0.1-b1024-g12-n64-h120 (23,400)
RotatE	lr0.1-b512-g10-n64-h16 (6,112)	lr0.1-b1024-g12-n64-h62 (23,684)
Models on interval205	low-dimensional	high-dimensional
HyperQuaternionE	lr0.01-b1024-g8-n8-h45 (9,823)	lr0.05-b1024-g8-n32-h150 (32,713)
HyperRotatE	lr0.05-b1024-g0-n16-h40 (9,978)	lr0.05-b1024-g0-n64-h132 (32,631)
QuaternionE	lr0.1-b1024-g12-n16-h45 (9,810)	lr0.1-b512-g12-n64-h150 (32,700)
RotatE	lr0.05-b1024-g12-n32-h23 (9,729)	lr0.05-b1024-g12-n32-h78 (32,994)

Table 5.5: Best parameter setting for each model on two datasets (low-dimensional vs. high-dimensional)

Evaluation Metrics

At testing, we compare different methods on two tasks: entity inference (Definition 5) and relation inference (Definition 6). Note that the constraint network method can only achieve the relation inference task while being incapable of inferring missing entities. Specifically, for each test sample $\langle h, r, t\rangle$, we generate three queries: $\langle ?h, r, t\rangle$ and $\langle h, r, ?t\rangle$ for the former task, and $\langle h, ?r, t\rangle$ for the latter. For each query, we utilize Eq.5.12 as the scoring function and measure distances between each candidate entity or relation and the correct answer. Then all candidate entities/relations are scored and later ranked by distances in the inference process. A smaller distance means a better fit to a query, indicating a higher likelihood of the entity/relation to be true. Following previous works [63, 146, 62], we choose two popular ranking-based metrics, namely Mean Reciprocal Rank (*MRR*), which measures inverse ranks of gold answers over all test samples on average and $H@k$ ($k \in \{1, 2, 3\}$), which measures the proportion of gold answers being ranked in the top k on average. In general, the higher the rank is, the better a model performs. Meanwhile, during the evaluation, we also follow [62] to filter out inference results that are already true in the KG[5].

5.5.4 Experimental Results

In this subsection, we first report the performance of our model in comparison with other embedding methods and traditional methods, and analyze what our model learns.

Comparison with embedding methods

Figure 5.4 and Figure 5.5 show our model performance against baseline embedding methods on the task of entity inference, and Figure 5.6 and Figure 5.7 report results on the task of relation inference. We summarize our main findings as below.

(1) **Our proposed method consistently outperforms baseline methods on two datasets in both low-dimensional and high-dimensional settings.** More specifically, in terms of the task of entity inference, compared with the strongest baseline method - HyperRotatE (in orange), HyperQuaternionE (in blue) gains around 8-point improvements in terms of *MRR* in both low-dimensional and high-dimensional settings,

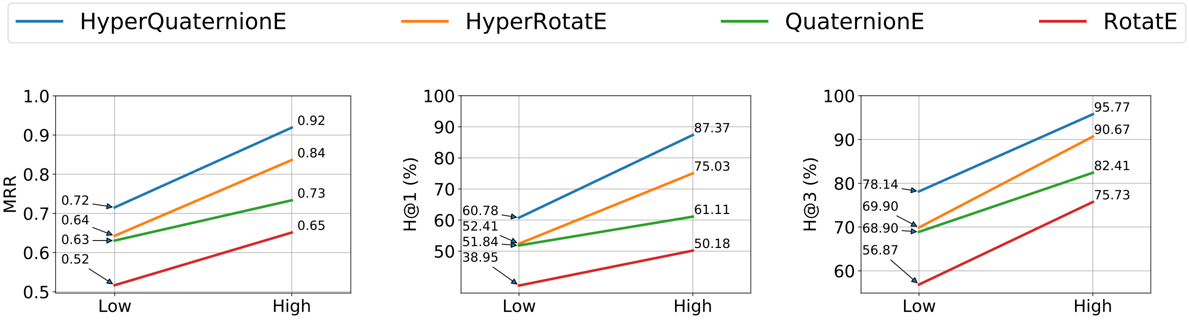
[5] For example, for a test query (geometry 1, dc, geometry 2), it is expected that a model should output *geometry 2* as the correct answer to a query (geometry 1, dc, ?t). However, there may exist other geometries in the KG that can satisfy the query. In such cases, the model should not be penalized if other valid geometries are ranked ahead of geometry 2.

respectively (see Figure 5.4). In terms of *H@1*, HyperQuaternionE beats HyperRotatE by around 8% in the low-dimensional setting, and by around 12% in the high-dimensional setting. On the interval205 dataset (See Figure 5.5), all embedding methods perform very well and the difference between our method and HyperRotatE is slightly subtle. Specifically, even in the low-dimensional setting (with 9,823 parameters), HyperQuaternionE reaches to around 91% in terms of *H@1* and 97.85% in terms of *H*@3.

In terms of the relation inference task (see Figure 5.6 and Figure 5.7), HyperQuaternionE still consistently outperforms all other embedding methods on all evaluation metrics. For example, HyperQuaternionE surpasses HyperRotatE by around 5% and 3 points in terms of *H@1* and *MRR* on the interval205 dataset, respectively. On the region187 dataset, our method improves HyperRotatE by around 5% and 2% in terms of *H@1* in the low-dimensional setting and high-dimensional setting, respectively. It is worth-noting that all embedding methods perform very well on the task of relation inference with *H@1* being over 95%. We compare our method with traditional reasoning methods in Section 5.5.4 on this task.

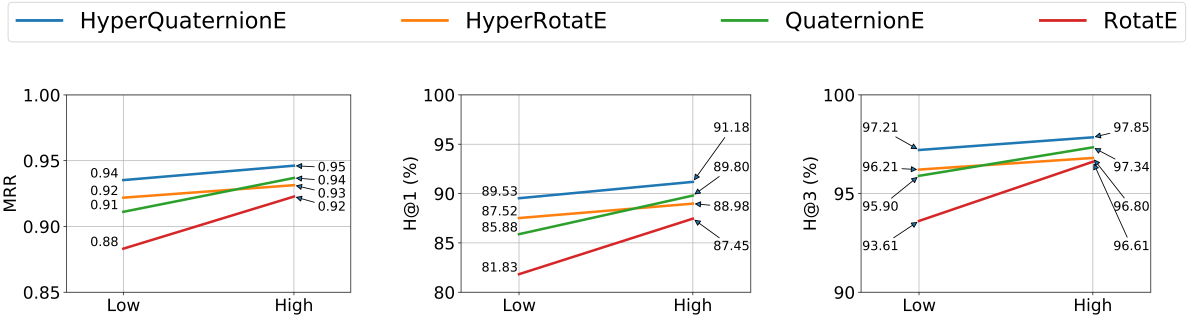
(2) **Hyperbolic embedding methods are more robust than Euclidean methods when handling spatial and temporal reasoning**. Apparently, hyperbolic embedding methods (i.e., HyperQuaternionE consistently exceeds their Euclidean alternatives (i.e., QuaternionE and RotatE) on both datasets for both tasks. For example, in the high-dimensional setting in Figure 5.4b, HyperQuaternionE improves over QuaternionE by around 14 points and HyperRotatE gains around 19 points against RotatE. In Figure 5.5c, HyperQuaternionE and HyperRotatE achieve improvements of 6.6% and 6% over their Euclidean alternatives, respectively. More remarkably, we find that the performance of hyperbolic embedding methods in low-dimensional settings is even comparable to that of their Euclidean equivalents in high-dimensional settings. In Figure 5.4b and 5.4c, HyperQuaternionE in the low-dimensional setting (5,858 parameters) is on a

par with QuaternionE in the high-dimensional setting (23,400 parameters). For instance, the difference in *MRR* (0.72 for low-dimensional HyperQuaternionE v.s. 0.73 for high-dimensional QuaternionE) is subtle.



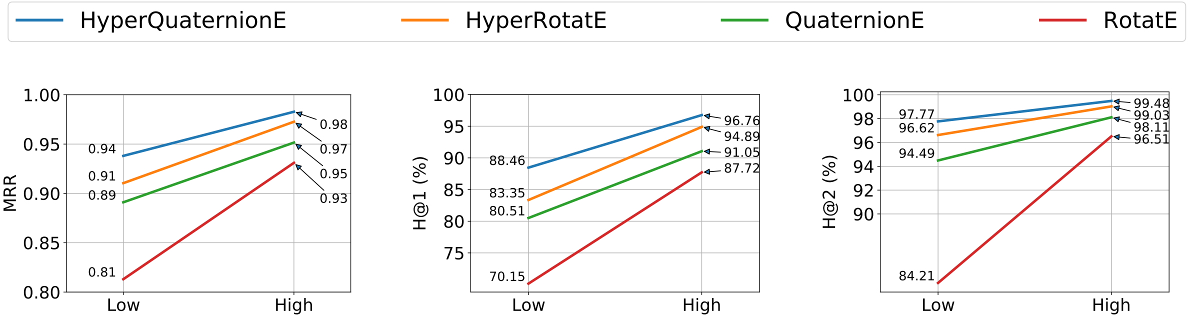
(b) MRR (c) H@1 (d) H@3

Figure 5.4: Model performance on the region187 dataset – entity inference task



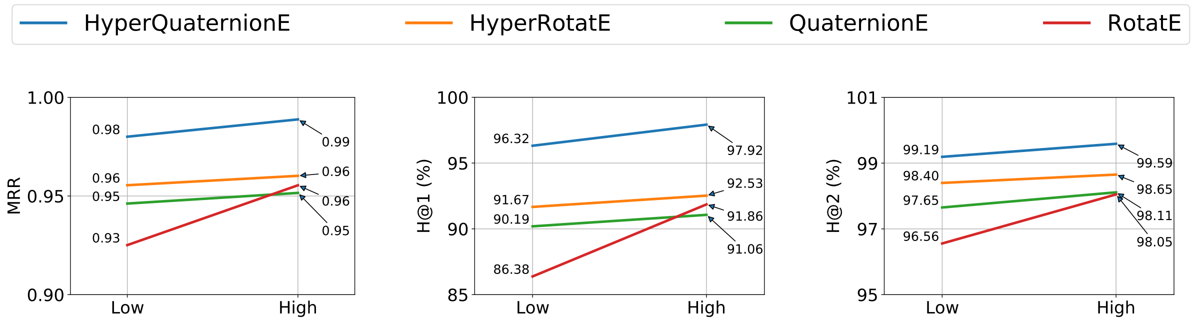
(b) MRR (c) H@1 (d) H@3

Figure 5.5: Model performance on the temporal205 dataset – entity inference task



(b) MRR (c) H@1 (d) H@2

Figure 5.6: Model performance on the region187 dataset – relation inference task



(b) MRR (c) H@1 (d) H@2

Figure 5.7: Model performance on the interval205 dataset – relation inference task

Comparison with traditional reasoners

We compare embedding methods in high-dimensional settings with a traditional method (i.e, the constraint network method which relies on composition tables) on the relation inference task. A challenge in this experiment is how to evaluate their inference results quantitatively. A traditional reasoner built upon RCC8/temporal composition tables usually yields *a set of* possible relations that could be held between two entities, despite the fact that there must be exactly one (spatial/temporal) relation holds between two entities. Differently, embedding methods usually output *a ranked list* of relations sorted by a scoring function (e.g., Eq. 5.12); see Table 5.6 for more details. In order to compare these two methods, we use five evaluation metrics - two absolute metrics for accuracy evaluation, two relative metrics for error evaluation and one for recall evaluation.

In terms of absolute metrics, we stick with *H@1* and *MRR* to evaluate their inference accuracy. For *H@1*, when the constraint network method yields only one relation, we call it a success since in theory only one (spatial/temporal) relation would be held between any two entities; otherwise, we view it as a "failure". This is a "strict" evaluation. In order to take into account the contribution of those "failures", we use *MRR*. In this case, if the constraint network method yields exactly one relation for a testing sample (e.g.,

Examples	Subject	Relation	Object	Constraint Network	HyperQuaternionE (Top 1)
0	103	$\boldsymbol{dc}$	72	dc, ec, po	dc
1	39	$\boldsymbol{ec}$	153	ec, po	ec
2	134	$\boldsymbol{po}$	140	po	po
3	49	$\boldsymbol{po}$	61	po	po
4	76	$\boldsymbol{dc}$	92	dc	dc
5	102	$\boldsymbol{eq}$	186	$dc, ec, eq, po, tpp, tppi$	eq
6	150	$\boldsymbol{tppi}$	31	$po, tppi$	$tppi$
7	22	$\boldsymbol{po}$	3	dc, ec, po	po
8	65	$\boldsymbol{tpp}$	150	$ntpp, tpp$	tpp
9	122	$\boldsymbol{tppi}$	40	$po, tppi$	$tppi$

Table 5.6: Examples of relation inference results. Both methods aim to infer the relation between a subject and an object. Column Relation denotes the correct relation, column Constraint Network and HyperQuaternionE denote their respective inference results. Note that constraint network method outputs a set of possible relations while HyperQuaternionE yields a ranked list of relations. Here we only show Top 1 relation from the ranked list.

$\langle h, ?r, t\rangle$), then the score for this sample is 1. Otherwise, the score for a sample with a set of inferred relations will be the average *MRR*s of the correct relation being ranked at any position in the answer set, which is $\frac{1}{|s|}\sum_{n=1}^{|s|}\frac{1}{n}$ ($|s|$ is the number of elements in the set s).

	constraint network		HyperQuaternionE	
Training Size	H@1	MRR	H@1	MRR
70%	76.8%	0.927	**96.8%±0.3%**	**0.983±0.002**
60%	74.9%	0.920	**93.5%±0.1%**	**0.965±0.004**
50%	71.3%	0.906	**91.0%±0.5%**	**0.951±0.003**
40%	67.1%	0.890	**88.3%±0.8%**	**0.935±0.003**
30%	60.9%	0.865	**82.8%±0.4%**	**0.902±0.002**

Table 5.7: *H@1* and *MRR* on the region187 dataset. ± indicates the following is the standard deviation.

Table 5.7 and 5.8 show the accuracy comparison between the constraint network method and HyperQuaternionE with varying sizes of training data. We find that **our model outperforms the constraint network method on spatial reasoning tasks by significant margins in terms of different training sizes and achieves com-**

	constraint network		HyperQuaternionE	
Training Size	H@1	MRR	H@1	MRR
70%	96.8%	0.989	**97.9%±0.2%**	0.989±0.001
60%	96.6%	0.986	**97.1%±0.3%**	0.984±0.002
50%	96.0%	0.984	**96.7%±0.2%**	0.982±0.002
40%	95.0%	0.981	**95.8%±0.3%**	0.979±0.004
30%	93.0%	0.971	**94.2%±0.4%**	0.970±0.002

Table 5.8: *H@1* and *MRR* on the interval205 dataset.

parable results on temporal reasoning tasks. With respect to the "strict" accuracy evaluation – *H@1*, HyperQuaternionE consistently surpasses the constraint network method on both spatial and temporal relation inference. In Table 5.7, HyperQuaternionE beats the constraint network method by over 20% for all different training sizes on the region187 dataset. On the interval205 dataset (see Table 5.8), our method consistently outperforms the constraint network method by around 1%. Additionally, with the training size increasing, we observe that both methods improve as we expect. It is worth-noting that even with only 30% data (of the entire graph) being in the training set, our method can obtain 82.8% and 97.9% in terms of *H@1* on these two datasets, respectively. In terms of *MRR*, a similar pattern of their performance is observed: HyperQuaternionE outperforms the constraint network method by around 5 points on the region187 dataset; however the differences between both methods on the interval205 dataset are relatively subtle but both achieve very high scores (i.e., over 0.97) for all different training sizes.

Despite the fact that the constraint network method does not necessarily to uncover the single (*true*) relation between entities, inference results are theoretically guaranteed by composition tables based on the amount of data given. Put differently, the correct relation is always a member of the result/answer set. We denote this inferred results as *theoretical results*. Here we are interested in evaluating errors of our inference against the theoretical results. We use two relative metrics - *Error Ratio* and *Recall-Coverage Ratio* to achieve this. *Error Ratio* - *ER* measures the failure of our model against the inference of composition tables. For a testing sample, it examines whether the Top 1 relation produced by our method is a member of the theoretical results yielded by the constraint network method. We use the average score over all testing samples as its final *Error Ratio* of our model. It can be expressed as follows.

$$\textit{Error Ratio} = \frac{1}{n}\sum_{i=1}^{n} TrueOrFalse_i \tag{5.23}$$

Here, for a testing sample i, if Top 1 relation in our ranked list is *not* a member of its corresponding theoretical relation set, then $TrueOrFalse_i$ will be 1; otherwise, $TrueOrFalse_i$ will be 0. n is the number of testing samples.

In addition, we introduce a *Recall-Coverage Ratio - RC-R* to measure the difficulty of our model in recalling results from the classical RRC8 reasoner. Specifically, for a ranked list of relations produced by our model regarding a testing sample $\langle h, ?r, t\rangle$, we calculate the ratio of the cardinality of the theoretical result set over the minimal length of a ranked list (staring from the first position) containing all relations in the theoretical set. This measure can be formulated as follows:

$$\textit{Recall-Coverage Ratio} = \frac{1}{n}\sum_{i=1}^{n} \frac{|s_i|}{\max_{r\in s_i} pos(r)} \tag{5.24}$$

Here, s_i is the result set from the classical RCC8 reasoner for a testing sample i and $pos(r)$ denotes the position index of relation r (from s_i) in our ranked list (1-index).

Additionally, we calculate the *Recall (R)* of our method. In the literature, *Recall* is defined to measure whether a true relation is contained in the result produced by a model. For the constraint network method, its *Recall* is always 1. As mentioned above, for a testing sample, its inference result always contains the correct relation, since the method performs a filtering-out operation, which excludes impossible relations between two entities. In our method, we also examine our *Recall* against the constraint network method. For each testing sample, we check whether the correct relation is contained in the top $|s|$ of our ranked list ($|s|$ is the cardinality of the relation set s produced by the constraint network method). This ensures that the sublist of our ranked list used in the *Recall* calculation has the same length as the relation set from the constraint network.

	region187			interval205		
Training Size	RC-R	ER	R	RC-R	ER	R
70%	96.64%	2.29%	100%	98.91%	1.38%	100%
60%	93.80%	4.08%	100%	98.59%	1.83%	100%
50%	92.46%	5.95%	100%	98.38%	1.95%	100%
40%	91.52%	6.69%	100%	97.66%	2.44%	100%
30%	88.68%	9.63%	100%	96.37%	3.49%	100%

Table 5.9: Error Ratio, Recall-Coverage Ratio and Recall on two datasets.

Table 5.9 shows *Error Ratio (ER)*, *Recall-Coverage Ratio (RC-R)* and *Recall (R)* of our method against the theoretical results. As expected, *Error Ratio* increases and *Recall-Coverage Ratio* drops as the training size decreases. When the training size is 70%, *ER* is as low as 2.29% and 1.38% on the region187 dataset and interval205, respectively. Meanwhile, *Recall-Coverage Ratio* reaches to 96.64% and 98.91%, respectively. Even when the training size drops to 30%, *Error Ratio* is still low (9.63% on the region187 dataset and 3.49% on the interval205 dataset). Similarly, the *Recall-Coverage Ratio* is 88.86% and 96.37%, respectively. Moreover, it is worth noting that we achieve the same *Recall* as the constraint network method does, meaning that the correct answer is also contained in the top $|s|$ of our ranked list. Overall, the results from Table 5.9 clearly show the suitability of our method for inference.

Summing up all presented evaluations, the results demonstrate that our embedding method can produce results of a *higher* accuracy for reasoning over relations than the constraint network method. Moreover, although our method can also achieve a *Recall* of as high as 100% as the constraint network method does, *Recall-Coverage Ratio* in Table 5.9 indicates these two methods may adopt different reasoning mechanisms or our embedding method may use other implicit inference. It would be interesting to study and analyze the underlying reasoning techniques in the future. In Section 5.5.4, we qualitatively analyze our model and examine what has been learned by our model from data.

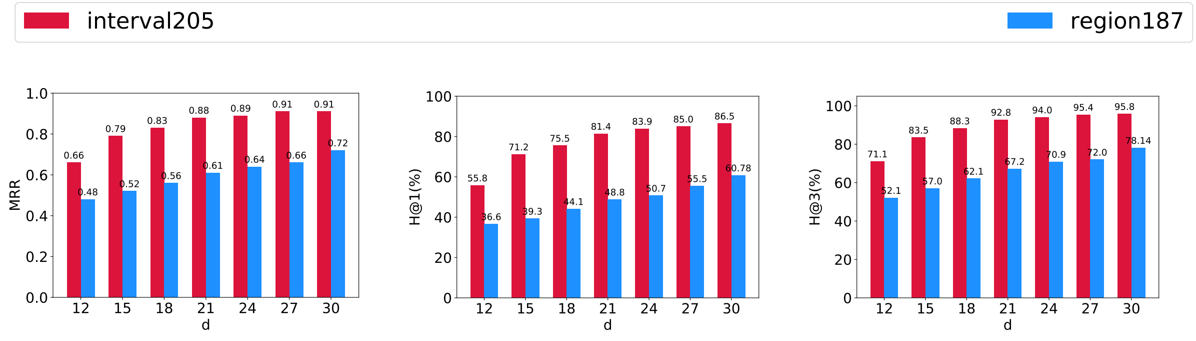
Comparison between spatial reasoning and temporal reasoning tasks

By contrasting the performance of spatial reasoning and temporal reasoning (e.g., Figure 5.4b and Figure 5.5b, Figure 5.6b and Figure 5.7b, Figure 5.4c and Figure 5.5c, etc.), we can easily find that achieving temporal reasoning is relatively easier than spatial reasoning, at least when the proportion of missing relations is the same. Note that we use 70% of the entire dataset as the training set for both spatial and temporal reasoning (see Table 5.4). In low-dimensional settings (see Figure 5.4b and Figure 5.5b), HyperQuaternionE yields an *MRR* of 0.72 on the region187 dataset while obtaining an *MRR* of 0.94 on the interval205 dataset. Similarly, in Figure 5.7c and 5.6c, HyperQuaternionE in low-dimensional settings yields 88.46% and 96.32% on the region187 dataset and interval205, respectively. Moreover, we observe a similar pattern from Table 5.7 and 5.8. For instance, we can see that when the training size is the same, both the constraint network method and our method are better at reasoning about temporal relations.

In order to further test the hypothesis that temporal reasoning is relatively easier to achieve, we conduct experiments to compare the performance of our model in spatial reasoning and temporal reasoning tasks with changing hidden dimensions, which determines the number of learnable parameters (see Appendix B.2) and thus impacts the training efficiency[6]. Figure 5.8 and 5.9 demonstrate that our model indeed consistently performs better on temporal reasoning tasks, particularly on the task of entity inference. For instance, with a hidden dimension of 12, our model can yield an *H@1* of 55.8 for temporal

[6] Usually a training process needs more time when the hidden dimension is high.

entity inference while obtaining 36.6 for spatial entity inference. With the hidden dimension increasing, the gap between them is shrinking even though it is still significant. With a hidden dimension of 30, when the model reaches to 0.91 in terms of MRR on the temporal entity inference task, MRR of the spatial case yields 0.72. This observation may also be viewed as a potential advantage of embedding methods against traditional methods that rely on path-consistency checking (e.g., the constraint network method). For path-consistency checking based methods, as the number of relations increases, composition tables often become more complicated and thus reasoning over relations will be inefficient. That is, the efficiency of the traditional reasoner is bounded by the complication of composition tables as relations involved increase. However, empirical experiments shown above disclose that embedding-based methods like HyperQuaternionE, with less parameters can obtain a even better result when reasoning over temporal relations than over spatial relations; thus they are more efficient on reasoning over temporal relations. This observation indicates the fact that the performance and training efficiency of embedding methods may not be bounded by the complication of composition tables, which is another advantage of embedding methods. We leave more in-depth theoretical and empirical analyses as future work.

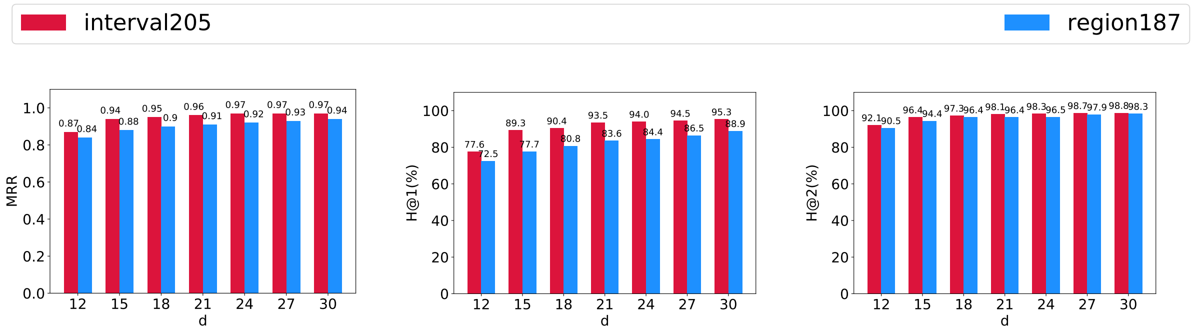


(b) MRR

(c) H@1

(d) H@3

Figure 5.8: Performance comparison between temporal and spatial entity inference tasks. d is the hidden dimension.



(b) MRR (c) H@1 (d) H@2

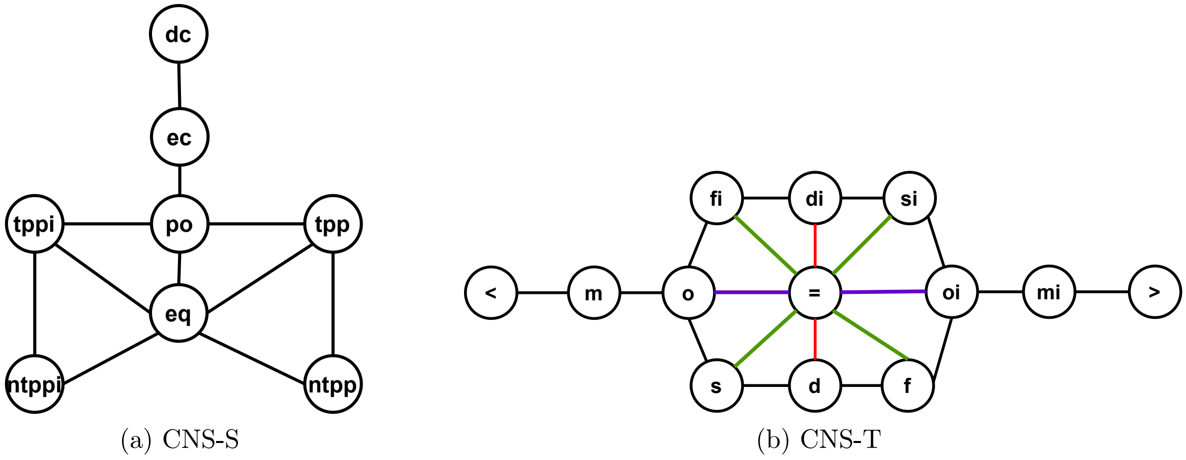
Figure 5.9: Performance comparison between temporal and spatial relation inference tasks.

Qualitative Analysis

In this section, we are interested in the question whether embedding methods are able to implicitly learn knowledge from data. This perspective not only suggests promoting embedding methods as a new tool for knowledge discovery, but may also help the design of new models. That is, if some domain knowledge can be learned implicitly, there is no need to make theories/domain knowledge explicit during the model design.

In particular, we examine whether embedding methods could learn conceptual neighborhood structures implicitly, which is fundamental to spatial and temporal reasoning. According to [41, 127, 128], if two relations between pairs of entities (i.e., geometries or events) can be directly transformed from one to the other by continuous deformation of entities (i.e., enlarging, shrinking, lengthening or shortening), these two relations are conceptual neighbors. Conceptual neighborhood structures of spatial and temporal relations are illustrated in Figure 5.10.

In order to investigate whether embedding methods manage to learn these structures, we create (spatial/temporal) relation networks. In a spatial/temporal network, nodes are relations and the linkages between relations are determined by the result of the relation inference task. More specifically, in the relation inference task, for a testing sample, (e.g., $< h, ?r, t >$), our model will output a ranked list of all relations sorted by scores in a



(a) CNS-S

(b) CNS-T

Figure 5.10: Conceptual neighborhood structure (CNS) [117, 41]. Figure 5.10a illustrates conceptual neighbors of spatial relations. Figure 5.10b reveals conceptual neighbors of temporal relations, in which there are three types of neighboring relations to the relation *equal* (i.e., =), distinguished by three different colors.

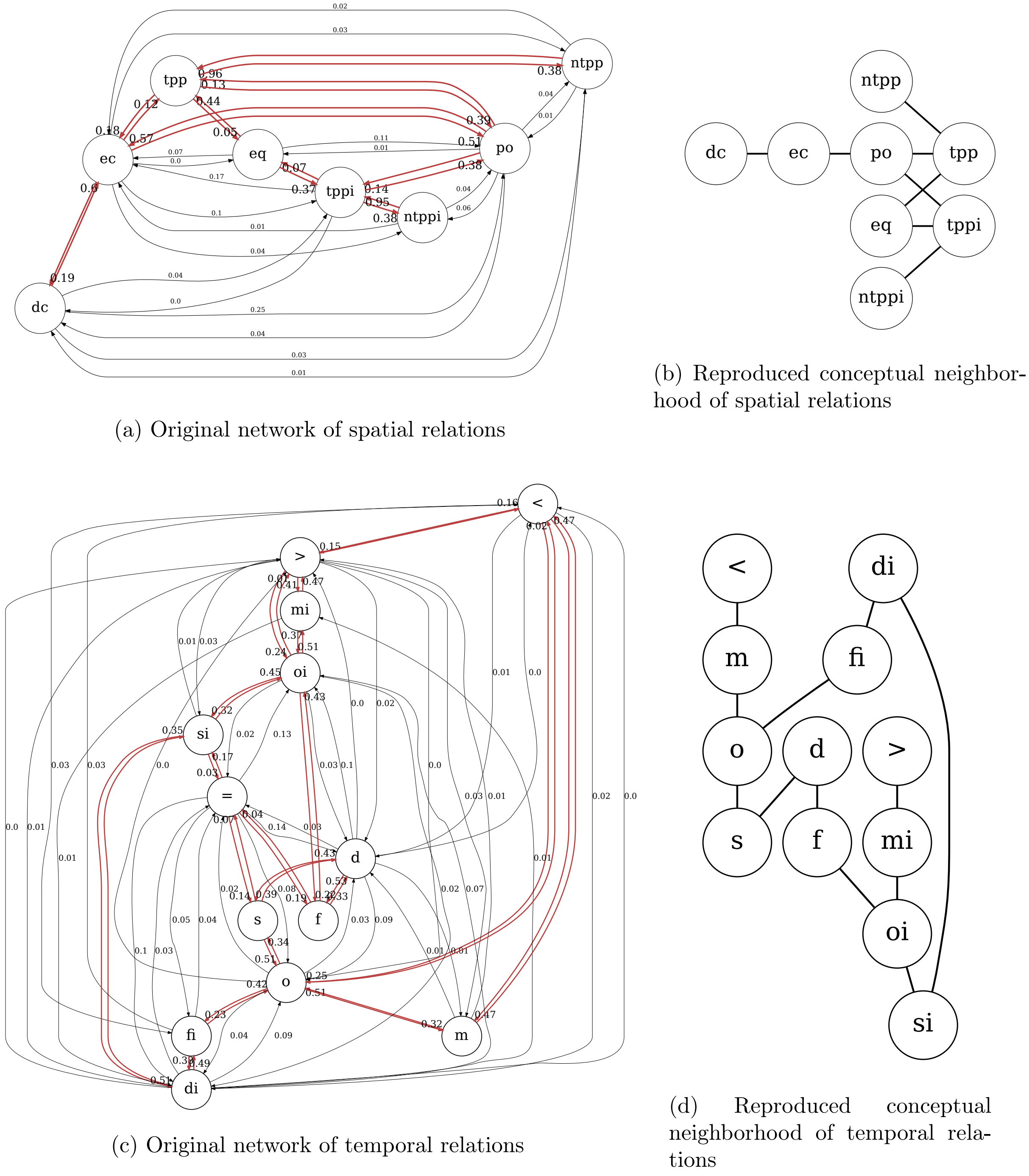
descent order, in which a relation with a high score means a higher likelihood to be the relation held between h and t. We pick Top 1 and Top 2 relation from the ranked list and establish a directed edge from Top 1 relation to Top 2 relation to indicate these two relations are likely to be concept neighbors. The underlying rationale is that relations that are conceptual neighbors are hard to be distinguished when determining which one is the true relation held between two entities, thus neighboring relations are supposed to be ranked closely by embedding methods on the task of relation inference. After going through all the samples, we obtain a directed relation network. In order to measure the strength of connections between two relation nodes, we weight each directed edge by the ratio of outgoing edges from the source relation node to the target relation node over the total number of outgoing edges from the source relation node.

Figure 5.11 reveals original relation networks as well as conceptual neighbor structures yielded by HyperQuaternionE. Figure 5.11a and 5.11c are original relation networks, where nodes are spatial/temporal relations and the label on a directed edge is the strength of connections. Edges between two nodes are highlighted in red when the sum of weights in both directions is over threshold of 0.40[7], which turns out to be neighborhood structures of relations shown in Figure 5.11b and 5.11d after removing labels and arrows. In general, Figure 5.11b and Figure 5.10a are alike and Figure 5.11d is similar to Figure 5.10b. It indicates that our embedding method is capable of implicitly learning conceptual neighborhood structure of spatial/temporal relations. However, due to a lack of *equal* relations in both region187 and interval205[8], it fails to completely reproduce the structure around *eq*/=. In addition, we find that for temporal relations another reason of failure for *equal* relation is that it has multiple conceptual neighbors and the proportion of outgoing edges to each target relation is marginal. Thereby, a relatively large threshold would easily filter out edges linked to the relation = (see Figure 5.12). It reveals that our method successfully rules out four relations (i.e., <, m, mi, and >) that are impossible to be conceptual neighbors of the relation = and learns that all the other eight relations can be transformed from it by differing proportions ($0.05 - 0.19$). This echos the neighborhood structures around relation = in Figure 5.10b.

Moreover, we set varying thresholds to investigate the closeness of neighboring relations. Figure 5.13 reveals that *nttpi-tppi* and *ntpp-tpp* are densely connected over changes, which is in line with the discovery of [117] that topological distances between them are the least. Furthermore, our method identifies another closely-connected chain: *dc-ec-po*, which intuitively makes sense as *ec* is the critical condition of continuous transformation between *dc* and *po*. Figure 5.14b, 5.14c and 5.14d confirm the stability of the found network structure between temporal relations. Meanwhile, it is interesting to see even when the threshold is set as large as 0.7 (meaning only edges with the strongest connections remains), two chain structures are recognized, where each relation and its

[7] This threshold is chose empirically. We also report results when thresholds vary in Figure 5.13 and 5.14.

[8] There are only 192 triples with *eq* relation (187 of which is self-equal (e.g., $< h, eq, h >$)) and 210 triples with = relation (205 of which is self-equal).



(a) Original network of spatial relations

(b) Reproduced conceptual neighborhood of spatial relations

(c) Original network of temporal relations

(d) Reproduced conceptual neighborhood of temporal relations

Figure 5.11: Conceptual neighborhood structures yielded by HyperQuaternionE.

inverse are separated in different chains.

Last but not least, we compare network structures of relations yielded by different

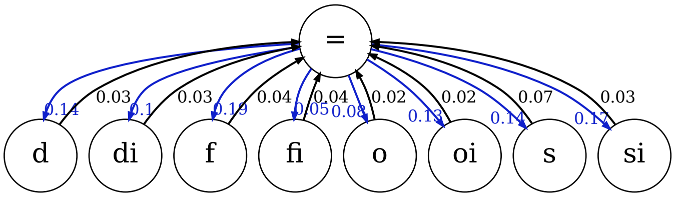
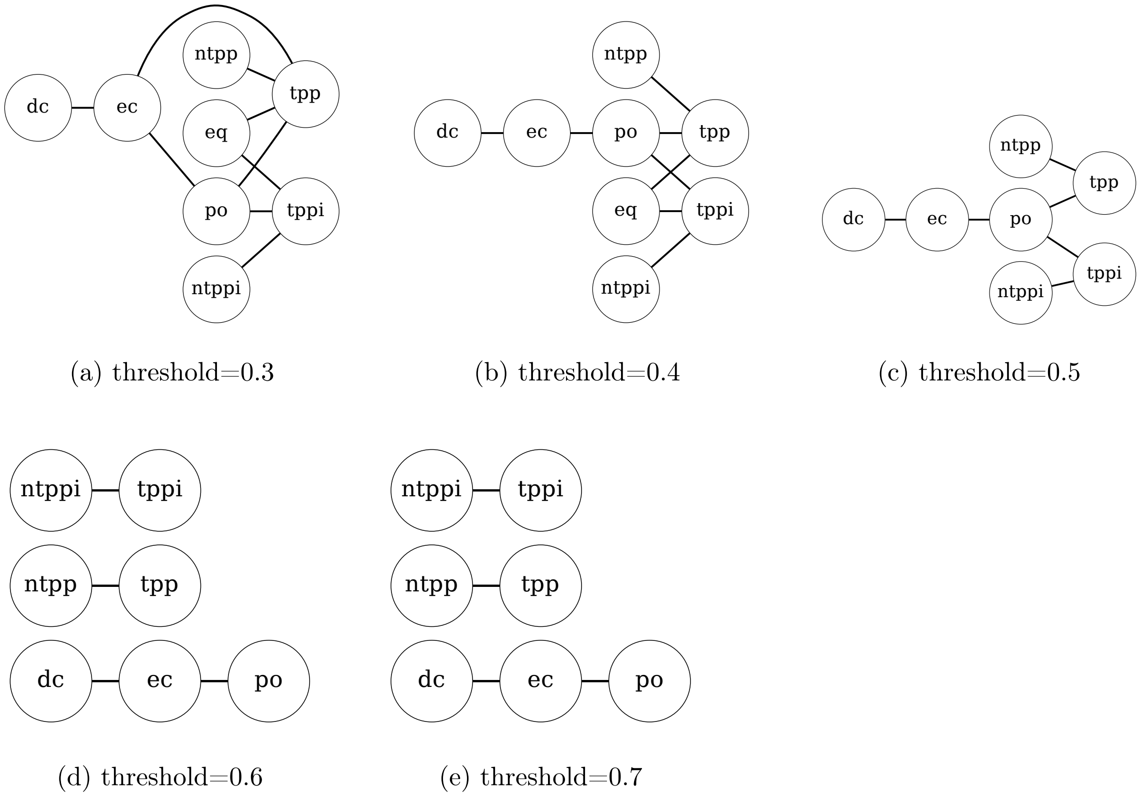


Figure 5.12: Original relation network around the relation =. Edges in blue are its outgoing edges while edges in black incoming.

embedding models (see Figure B.1 and B.2 in Appendix). In general, results show that all embedding models are capable of implicitly learning neighborhood structures of relations with nuanced differences.



(a) threshold=0.3

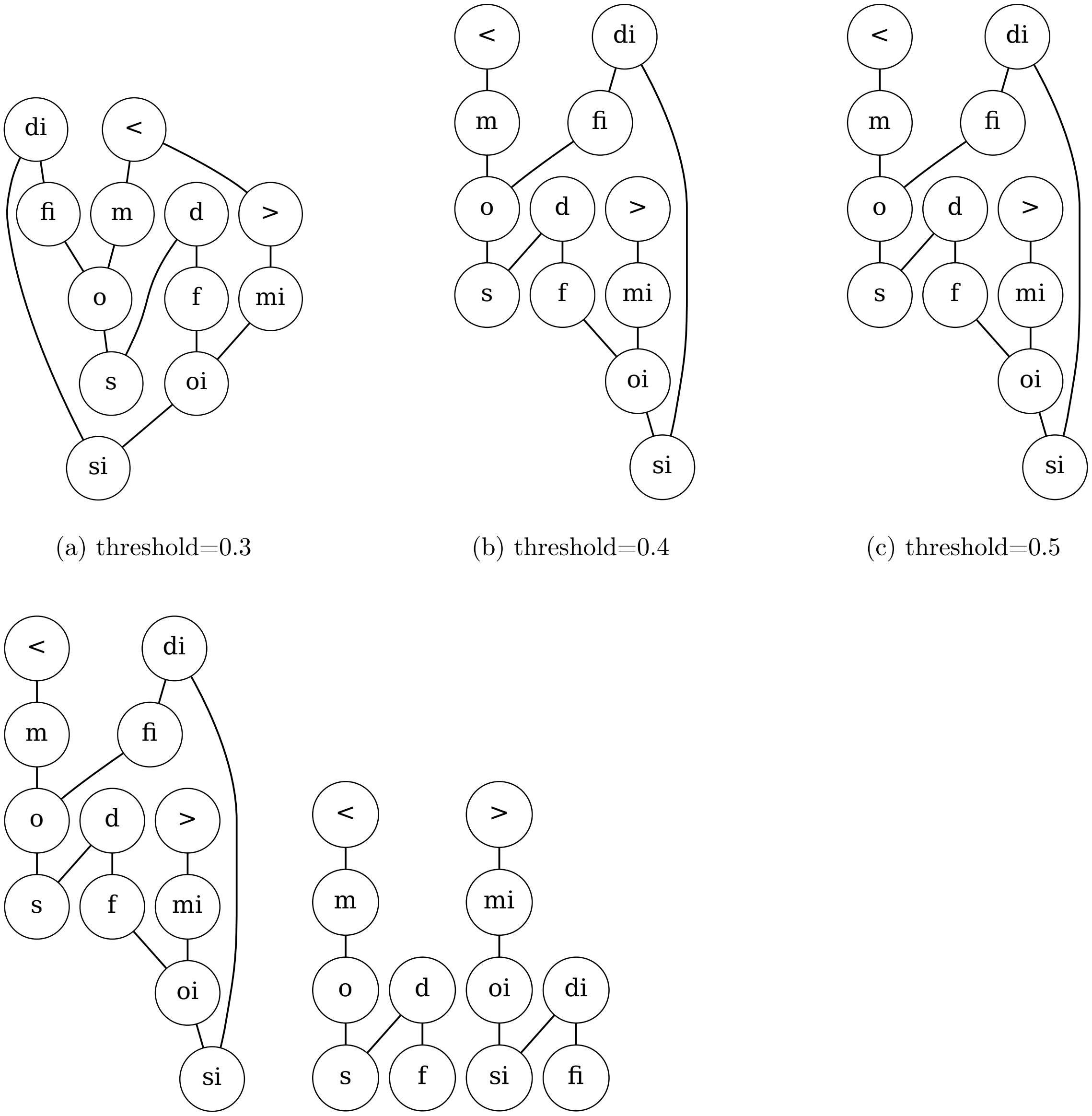
(b) threshold=0.4

(c) threshold=0.5

(d) threshold=0.6

(e) threshold=0.7

Figure 5.13: Network structures with varying thresholds (spatial relations).



(a) threshold=0.3

(b) threshold=0.4

(c) threshold=0.5

(d) threshold=0.6

(e) threshold=0.7

Figure 5.14: Network structures with varying thresholds (temporal relations).

5.6 Discussion and Future Work

Qualitative spatial and temporal reasoning [127, 40] have played a crucial role for a wide range of tasks such as topological integrity constraints in GIS, spatial queries, navigation and orientation in robotics, representing spatial human cognition, and so forth. Traditionally, composition tables of RCC-8 relations and temporal relations have been widely adopted in spatial reasoners to accomplish inference tasks. However, such symbolic reasoning with explicitly-injected knowledge has many restrictions that arise from the inability to efficiently deal with noise, missing data, high-order neighborhood information, or large datasets in general. This makes existing techniques unsuitable for many interesting applications, such as knowledge base completion and knowledge graph-based recommendation. Recently, success stories in Machine Learning (ML), in particular embedding techniques, shed light on spatial and temporal reasoning, thanks to their sub-symbolic and learnable representations of knowledge. In this paper, we designed novel embedding-based methods for spatial and temporal reasoning and examined how these methods perform when compared against traditional methods. We were especially interested in examining whether embedding-based methods learn domain knowledge implicitly from data.

In order to answer these questions, we developed an embedding model, named as HyperQuaternionE. Our method is able to encode symmetric/anti-symmetric properties of relations and inverse relations, and can automatically find and capture composition patterns of relations from data, which is key to automatic spatial and temporal reasoning. Moreover, our method provides a hyperbolic embedding space to embed tree-like structures over entities induced by transitive relations such as *after* and *non-tangentially proper part*. We evaluated our work using two synthetic datasets (region187 and interval205), and compared different methods against relation inference and entity inference tasks. **The experimental results revealed that our embedding method achieves superior performance on both datasets in terms of both tasks and outperformed both other baseline embedding methods and the constraint network method relying on composition tables.**

We hypothesize that such strong results are partially because embedding methods are capable of capturing constraints from both local and global high-order information through training. Representations of entities and relations are learnable and updated globally over iterations. Another advantage of embedding methods lies in that they yield ranked lists of relations with high precision rather than sets of relations without order produced by traditional methods. A ranked list is more preferable, since in theory exactly one topological relation between two geographical entities holds due to the relations' jointly exhaustiveness and pairwise disjoint (JEPD) characteristic. Moreover, we argued that embedding methods have much broader applications than traditional reasoners, such as entity inference and checking the validity of relations between two entities.

In order to answer the second research question, we analysed relation inference results and found that **embedding methods implicitly learned conceptual neighborhood structures of spatial relations and temporal relations, and some neighborhood structures are much more closely connected (such as *dc-ec-po* and *nttpi-tppi*) than others.** This is a valuable discovery in two aspects. First, from the viewpoint of model interpretation, it helps explain why embedding methods succeed in spatial and temporal reasoning. Early on, [127, 128] pointed out that the representation and/or reasoning processes will be considerably simplified by incorporating conceptually neighboring relations into reasoning. Second, from the viewpoint of model design, this suggests that understanding and analyzing what machine learning methods are able to learn from existing data is of great importance to theory-informed model design. For instance, with "enough" data available, as shown in our paper, conceptual neighbors of relations can be learned automatically and implicitly by models from data, and, thus, incorporating such theories/spatial thinking explicitly would not supply extra useful information.

Following the discussion above, this work raises several questions that deserve further investigation. First, in this paper we focused on the *qualitative* reasoning capability of embedding methods, and, thus, intuitively we assume the developed methods would not be affected by the original geometries of geographical entities. However, given that geographical entities with complex geometries (e.g., arbitrary polygons, polygons with holes, etc.) may bring about complex topological relations, it is worth examining the adaptability of embedding methods to such cases. Second, it is worth further exploring what other spatial theories or knowledge in spatial and temporal reasoning can be/have been learned implicitly in addition to conceptual neighborhood structures. This direction, broadly speaking, falls into the the bigger trend of *explainable AI* and ML in geography which is key for accountable data-driven decision making.

Chapter 6

Automatically Discovering Conceptual Neighborhoods Using Machine Learning Methods

This chapter addresses a follow-up question raised from last chapter, namely why subsymbolic reasoning methods achieve better results in terms of qualitative spatial and temporal reasoning. A graph-based method is presented to measure similarities of qualitative relations by analyzing reasoning results produced by subsymbolic methods. I find that the resulting structure of similar relations yielded by the proposed method is well-aligned with the conceptual neighborhood structure of temporal relations in theoretical literature. Meanwhile, extensive experiments are conducted to study how many training data are needed for subsymbolic methods to recover such structure. This chapter concludes that subsymbolic methods can learn conceptual neighborhood structures purely from data.

Peer Reviewed Publication	
Title	Automatically Discovering Conceptual Neighborhoods Using Machine Learning Methods
Author	Ling Cai, Krzysztof Janowicz, and Rui Zhu
Venue	15th International Conference on Spatial Information Theory (COSIT 2022)
Editors	Toru Ishikawa, Sara Irina Fabrikant, and Stephan Winter
Publisher	Schloss Dagstuhl – Leibniz-Zentrum für Informatik
Pages	3:1–3:14
Publication Date	05 September 2022
Copyright	All authors
DOI	10.4230/LIPIcs.COSIT.2022.3

Abstract Qualitative spatio-temporal reasoning (QSTR) plays a key role in spatial cognition and artificial intelligence (AI) research. In the past, research and applications of QSTR have often taken place in the context of declarative forms of knowledge representation. For instance, conceptual neighborhoods (CN) and composition tables (CT) of relations are introduced explicitly and utilized for spatial/temporal reasoning. Orthogonal to this line of study, we focus on bottom-up machine learning (ML) approaches to investigate QSTR. More specifically, we are interested in questions of whether similarities between qualitative relations can be learned from data purely based on ML models, and, if so, how these models differ from the ones studied by traditional approaches. To achieve this, we propose a graph-based approach to examine the similarity of relations by analyzing trained ML models. Using various experiments on synthetic data, we demonstrate that the relationships discovered by ML models are well-aligned with CN structures introduced in the (theoretical) literature, for both spatial and temporal reasoning. Noticeably, even with significantly limited qualitative information for training, ML models are still able to automatically construct neighborhood structures. Moreover, patterns of asymmetric similarities between relations are disclosed using such a data-driven approach. To the best of our knowledge, our work is the first to automatically discover CNs without

any domain knowledge. Our results can be applied to discovering CNs of any set of jointly exhaustive and pairwise disjoint (JEPD) relations.

6.1 Introduction

Since the 90s, Qualitative Spatio-Temporal Reasoning (QSTR) has attracted attentions from researchers and practitioners in several fields, such as geographical information science, artificial intelligence and cognitive science [159, 117, 160, 120, 161, 127]. Aside from the clear connection to human representations and linguistic communication of the spatial configuration of our environment, QSTR has numerous advantages over its quantitative counterpart [118]. Representing qualitative information by using symbols and developing calculi to infer unknown qualitative information is the key to QSTR. Different sets of qualitative spatial relations (such as directional and topological relations) along with a system of qualitative caculi are developed [162], among which reasoning over topological relations becomes the most well-established area in QSTR.

As far as regions are concerned, the most well-known formalizations for qualitative topological relations are - the Region Connection Calculus (RCC-8) [42] and the 9-Intersection Model (9-IM) [163, 117]. Both arrive at the same conclusion that there exist eight base topological relations between regions in 2D space, although they are developed independently during the earlier 90s [40]. Those relations form the foundation for a variety of qualitative spatial reasoning techniques [164, 117, 126, 165]. Two major (and interconnected) lines of works are: (1) Composition Tables (CTs) (i.e., transitivity tables), which store possible resulting relations arising from the composition of two relations [29, 42, 116, 166]. (2) Conceptual Neighborhood Graphs (CNGs), which formalize transitions between relations. Conceptual neighbors of a relation are defined as a set of relations that can be directly transformed into/from the relation by deforming (e.g., moving and scaling) the related entities continuously (in a topological sense) [128]. In a CNG, relations are modeled as nodes and an undirected edge is established between two neighboring relations (see Figure 6.3f). CNGs play an essential role for reasoning with uncertain or incomplete information [127], and have been used in research of cognitive similarity assessment [167, 168] and modeling of linguistic spatial terms [169]. In addition to topological relations, composition tables, and conceptual neighborhoods have also been developed for reasoning over temporal relations [29, 128].

Those reasoning methods follow a top-down manner, which usually requires (noise-free) explicit domain knowledge. On the contrary, success in data-driven Machine Learning (ML) approaches, which are insensitive to noise and good at dealing with incomplete information as well as uncertainty, provides new opportunities to study QSTR from a bottom-up perspective. ML models rely solely on training data to discover patterns/rules that can be implicitly used for reasoning rather than explicitly injecting domain knowledge into the model. However, the question of why they succeed and whether they are able to (re)discover theories, here in the sense of rule sets or CNGs, is unexplored.

In this paper, we propose a graph-based approach to investigate similarities of qualitative relations from a bottom-up perspective. Particularly, we are interested in how the similarities derived from ML methods are related to classic theoretical studies (e.g., on conceptual neighborhoods). By conducting extensive experiments on synthetic data regarding spatial reasoning (here, *RCC-8* relations) and temporal reasoning (here, Allen's thirteen interval relations), we are able to demonstrate that ML models can automatically discover conceptual neighborhood graphs. In addition, experiment results showcase that such graphs can be easily discovered by ML methods even when limited data are available for training. Moreover, the similarities of relations are mostly asymmetric, which echos the findings in [167] from a perspective of cognitive assessment. Furthermore, patterns observed in asymmetric similarities of relations are disclosed. To the best of our knowledge, we are the first to automatically discover conceptual neighborhood graphs of qualitative relations from a bottom-up perspective by analyzing ML methods. In theory, our approach can be used to discover CNGs for any calculus with jointly exhaustive and pairwise disjoint (JEPD) relations.

The remainder of this paper is structured as follows: Section 6.2 introduces background about how to perform QSTR by using machine learning methods. Section 6.3 elaborates on the proposed graph-based approach to discover similarities among relations. Section 6.4 describes the generation of synthetic data, evaluation metrics, and reports experimental results. Section 6.5 discusses our findings and points out the direction for future studies.

6.2 Background

In this section, we introduce preliminaries of ML methods to achieve QSTR. We summarize notations and abbreviations we use in this paper in Table 6.1 for quick reference.

Terms (abbrev.)		
Qualitative Spatio-temporal Reasoning (QSTR)	Conceptual Neighborhood Graphs (CNGs)	
Machine Learning (ML)	Artificial Intelligence (AI)	
Knowledge Graphs (KGs)	Knowledge Graph Embedding (KGE)	
Composition Tables (CTs)	Jointly Exhaustive and Pairwise Disjoint (JEPD) relations	
RCC-8 Relations	**IR-13 Relations**	
disconnected (dc)	before (<)	after (>)
externally connected (ec)	meets (m)	met-by (mi)
partially overlapping (po)	overlaps (o)	overlapped-by (oi)
tangentially proper part (tpp)	during (d)	contains (di)
tangentially proper part inverse (tppi)	starts (s)	started-by (si)
non-tangentially proper part (ntpp)	finishes (f)	finished-by (fi)
non-tangentially proper part inverse (ntppi)	equal (=)	
equal (eq)		

Table 6.1: Terms and their abbreviations used in this paper.

6.2.1 Qualitative Representation of Relations

In this paper, we store binary relations between entities in form of triples. A triple of the form $\langle s, r, o\rangle$ represents an entity *subject* that has a *relation* to another entity *object*. For instance, the statement that a house is externally connected (ec) to a park can be represented as $\langle house, ec, park\rangle$. A set of such tripled is called a knowledge graph (KG). In our paper, a KG is a simple directed graph, consisting of entities being modeled as nodes and relations between them being modeled as labels of edges. Formally, it can be represented as $G = (V, E)$, where V and E are the set of nodes/entities and edges with relations being labels, respectively.

6.2.2 Relation Prediction Task

We will focus on a task known as relation prediction, namely inferring the relation between two entities based on other information. It is equivalent to answering the query $\langle s, ?r, o\rangle$. Examples include: *what is the topological relation between Los Angeles and Santa Monica?* or *what is the temporal relation between the Battle of Trafalgar and the Napoleonic Wars?*

Symbolic Reasoning Methods

Traditionally, symbolic representations are adopted to represent entities and relations, on top of which qualitative calculi are developed to perform reasoning tasks. For instance, CTs along with path-consistency algorithms are often used to infer missing relation between entities [116]. Given that (*property A, tangential proper part (tpp), park B*) and (*park B, disconnect (dc), house C*), we are able to infer that (*property A, disconnect (dc), house C*) by checking the CT of *RCC-8*. Usually such top-down approaches (which are based on qualitative calculi) fall into the group of symbolic reasoning. Despite their great success in qualitative reasoning in the past, such approaches are faced with noticeable limitations. For instance, they are sensitive to erroneous information or noise. Moreover, they can only be applied to a limited range of reasoning tasks, do not scale well over large datasets, and cannot be easily applied in combination with numeric approaches [170].

Knowledge Graph Embedding Methods

Knowledge Graph Embedding (KGE) methods are an embedding technique in ML that has been empirically proven to be effective in reasoning in a subsymbolic way.

Generally speaking, the goal of KGE methods is to learn subsymbolic representations of entities and relations in a high-dimensional continuous vector space while preserving the connectivity between entities and relations from KGs. Typically, developing a KGE model requires the following three components. (I) The first is to randomly initialize subsymbolic representations for each entity/relation in a high-dimensional continuous vector space. By doing so, each entity/relation is initialized as a high-dimensional vector (a.k.a embedding or subsymbolic representation) and can be viewed as a point in such high-dimensional vector space. The vector space could be Euclidean space, Hyperbolic space, Spherical space, etc., which vary between different KGE models. The embedding of an entity v, or a relation r, can be expressed as $\mathbf{v} \in \mathbb{U}^d$, or $\mathbf{r} \in \mathbb{U}^d$, where $\mathbb{U}$ denotes the vector space and d is its dimension.

(II) a scoring function is required to measure the likelihood of a triple being positive (i.e., a true statement). Various KGE models specify different scoring functions. For instance, TransE [62], the first KGE model, assumes that for a triple $\langle s, r, o \rangle$, the relation r is a transformation operator in a vector space, which translates the subject s to the object o. Thus the embedding of an object entity $\mathbf{o}$ should be equivalent to the resulting embedding of a subject entity $\mathbf{s}$ being translated by the relation $\mathbf{r}$ in the vector space. Then the distance between the embedding of the object entity and the resulting entity can be used as a scoring function: $score(s, r, o) = \|\mathbf{s} + \mathbf{r} - \mathbf{o}\|$. Thus, triples that are present in KGs (i.e., positive triples) will obtain a lower score while triples that are not present will gain a higher score.

(III) an objective function is needed for training through a process of optimization. A commonly used way of constructing such an objective function is by contrasting scores obtained by positive triples with those of negative triples. Often, the objective function is built upon the task of entity prediction (namely answering queries such as $\langle ?s, r, o\rangle$ or $\langle s, r, ?o\rangle$). For each positive triple $\langle s, r, o\rangle$, a number of negative triples (e.g., k) are generated by switching the subject s and/or the object o with other randomly selected entities (e.g., s_i or o_i). Then an objective function $\mathcal{L}$ can be defined to minimize scores for positive triples while maximizing scores for negative ones:

$$\mathcal{L} = -log\ \sigma(\gamma - score(s, r, o)) - \frac{1}{k}\sum_{i=1}^{k} log\ \sigma(score(s_i, r, o) - \gamma) \tag{6.1}$$

where σ is the sigmoid function and γ is a pre-specified hyper-parameter as a margin. $\langle s_i, r, o\rangle$ is a negative sample of $\langle s, r, o\rangle$.

After a number of iterative optimization over the training data, minimizing the objective function yields embeddings (representations) for all entities and relations in the KG. The optimized KGE model then can be used in various downstream tasks, such as entity prediction relation prediction, and triple classification. A plethora of KGE models have been developed in the the past years, e.g., [62, 63] and various scoring functions have been used (refer to [171] for more details).

Here we elaborate on how to perform relation prediction (i.e., answering a query $\langle s, ?r, o\rangle$) by using trained KGE methods, since they are closely related to our approach discussed in Section 6.3. Concretely, we enumerate all possible relations ($r' \in \mathcal{R}$) to replace $?r$ individually and then sort these relations by $score(s, r', o)$ in an ascending/descending order. Finally the embedding method regards the relation ranked first as the correct answer to the query. The table on the left in Figure 6.1 shows examples of ordered sets of relations produced by a trained KGE model regarding different testing queries.

6.3 Knowledge Graph Embedding Methods as Knowledge Miner

Similarity is one of the most commonly used measures to examine relationships of objects. For instance, domain experts introduce conceptual neighbors to indicate similar qualitative relations [128]. Likewise, in this section we introduce an approach to examine similarities between qualitative relations by analyzing trained KGE models from a bottom-up perspective. There are two steps in this approach - initial construction of a relation graph and its refinement.

The first question is how to derive similarities between any two relations in the set $\mathcal{R}$ from a trained KGE model. Our assumption is that it would be difficult for a trained embedding model to distinguish relations that are similar in a topological sense. That is, in terms of the task of relation prediction, for a testing query (geometry A, ?r, geometry B) (whose target answer is *externally connected* (*ec*)), we hypothesize the embedding-based model may yield similar scores for (geometry A, ***ec***, geometry B) and (geometry A, *partially overlap* (***po***), geometry B), because *po* and *ec* are topologically similar. Put differently, the sorted set of predicted relations reveals structural similarities among relations in the sense that similar relations are more easily confused in relation prediction (see Figure 6.1).

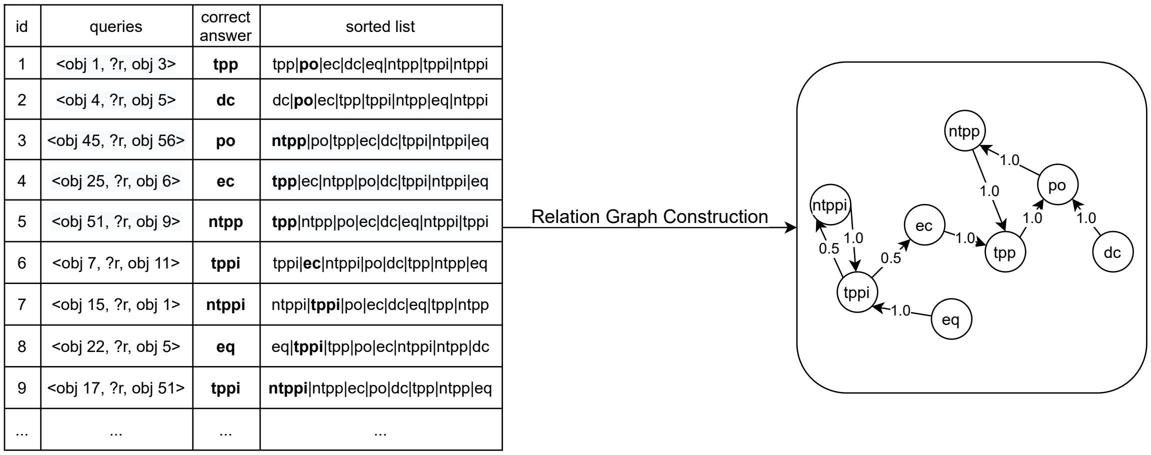
Based on this assumption, we initiate a graph in which vertices are different types of relations. For each testing query $\langle s, ?r, o\rangle$, a directed edge is established from the correct relation to either the relation ranked at first (top 1) or second (top 2) in the ordered list of relations. Such a choice relies on whether the relation at top 1 is the correct relation or not. When the correct is ranked at top 1, we do not introduce a loop. Instead, a directed edge starting from the correct relation to the relation at Top 2 is added. If the relation at Top 1 is not the correct, then an edge is built from the correct relation to Top 1. The resulting graph is a *directed* graph, whose edges originate from the correct relation to a relation identified as most similar to the correct by the KGE model. In a directed edge, we use terms - head and tail - to refer to the source and the target of an edge, respectively. The direction of edges reflects which candidate relation (tail) is similar to the target relation (head). Note that by such a distinction, we are able to examine the asymmetric similarities between relations.

The graph constructed above only illustrates which relations are considered as similar by a KGE model, but does not quantify similarities between relations. Here, we design a weighting function to quantify these similarities. Specifically, the weight of an edge is estimated as the proportion of the number of edges from a head to a tail relation over the total number of edges originating from the head. This function can be formulated as follows:

$$weight(r_i \rightarrow r_j) = \frac{count(r_i \rightarrow r_j)}{\sum_{r' \in \mathcal{R}} count(r_i \rightarrow r')} \tag{6.2}$$

where $count(r_i \rightarrow r_j)$ is the cardinality of edges originating from r_i (head) to r_j (tail) (with shortest paths). An example of the construction process is shown in Figure 6.1.

So far, we obtain a directed and weighted graph, which reveals the similarities between different relations; see Figure 6.2a. We observe that this graph is almost complete (i.e., any two relations/vertices are connected via an edge), because eventually any two relations are likely to be thought of as similar by a KGE model. However, not all these similarities are significant; for instance many edges only have marginal weights (e.g.,



id	queries	correct answer	sorted list
1	<obj 1, ?r, obj 3>	**tpp**	tpp|**po**|ec|dc|eq|ntpp|tppi|ntppi
2	<obj 4, ?r, obj 5>	**dc**	dc|**po**|ec|tpp|tppi|ntpp|eq|ntppi
3	<obj 45, ?r, obj 56>	**po**	**ntpp**|po|tpp|ec|dc|tppi|ntppi|eq
4	<obj 25, ?r, obj 6>	**ec**	**tpp**|ec|ntpp|po|dc|tppi|ntppi|eq
5	<obj 51, ?r, obj 9>	**ntpp**	**tpp**|ntpp|po|ec|dc|eq|ntppi|tppi
6	<obj 7, ?r, obj 11>	**tppi**	tppi|**ec**|ntppi|po|dc|tpp|ntpp|eq
7	<obj 15, ?r, obj 1>	**ntppi**	ntppi|**tppi**|po|ec|dc|eq|tpp|ntpp
8	<obj 22, ?r, obj 5>	**eq**	eq|**tppi**|tpp|po|ec|ntppi|ntpp|dc
9	<obj 17, ?r, obj 51>	**tppi**	**ntppi**|ntpp|ec|po|dc|tpp|ntpp|eq
...	...	...	...

Figure 6.1: Relation Graph Construction. Here nine queries are used as examples and the *sorted list* column shows relations sorted by a scoring function from a KGE method. Each relation is represented as a vertex in the graph and edges are established from the correct answer (column 3) to the relation in bold in the sorted list. Weights are calculated by using Eq. 6.2.

0.01). In order to extract significant relationships from the initial relation graph, the next step is to prune insignificant edges to get a refined graph.

Intuitively one could enumerate different thresholds (for instance, by gradually increasing a threshold (i.e., 0.0, 0.05, 0.1,..., 1.0)) to cut off edges whose weights are insignificant. Then one can terminate the enumeration process by manually checking whether the refined graph is aligned with our domain knowledge/cognition. However, without enough domain knowledge, it is hard to conclude which graph is meaningful and this means the proposed solution is not truly bottom-up. In order to reduce human intervention in the refinement process, we define a condition to automatically terminate the enumeration. The condition is based on the naive fact that *all relations/vertices must be preserved/connected in the graph after the refinement*, since our focus in this paper is on the relationships of all *relations*. Based on graph theory, such a fact boils down to ensuring that there is always one connected component in the graph after the refinement. Therefore, we can gradually increase thresholds by constant margins (e.g., 0.05) until the initial graph is no longer one connected component. In summary, in the process of refining relation graphs, we generate a number of candidate thresholds (within the range of $(0.0, 1.0)$ and a step of 0.05) in an ascending order and find the maximal threshold that leads to only one connected component in the graph, which is regarded as the refined relation graph.

6.4 Experiments

In this section, we introduce the synthetic data we use to test our method, the evaluation metrics used for graph similarity measure, and present experimental results. Although theoretically our proposed approach can be applied to any set of JEPD relations to automatically discover a graph of relations, we focus on *RCC-8* and *IR-13* here.

6.4.1 Data Preparation

Since real-life datasets are usually incomplete, we generate sets of synthetic data for the purpose of demonstration. Specifically, we choose rectangles as primitive geographical entities for *RCC-8* relations and closed-intervals as primitive temporal entities for *IR-13* relations.

To generate rectangles, we first set up a main area, in which rectangles should be located. By default, the main area is set to be a 15×15 unit square with the origin being its bottom-left corner. Then we randomly generate pairs of points within the square and each pair of points compose the top-left corner and the bottom-right corner of a rectangle[1]. Finally, we compute *RCC-8* relations between any two rectangles to generate synthetic spatial relation triples. Likewise, we generate a number of closed-intervals on the x-axis within the range [0, 500]. Specifically, we randomly select two integers from

[1] We ensure that each rectangle is valid. For example, if the two points in a pair align along the same axis, we will remove this pair.

the range and use the smaller one as the beginning of an interval and the bigger one as the ending of the interval. Then we compute the *IR-13* relations between any two intervals to generate synthetic temporal relation triples. We call the set of all synthetic triples as *complete synthetic data*.

However, without any prior knowledge, it is hard to decide how many rectangles/intervals should be generated within the given main area/line segment. Meanwhile, the number of rectangles/intervals generated in the same area/line may affect discovered relation graphs. Therefore, we independently generate several sets of synthetic triples for both the *RCC-8* and the *IR-13* relations with different number of rectangles/intervals (i.e., [64, 128, 256, 512, 1024]). These sets of triples have different densities of rectangles/intervals. The proportions of different relations generated with respect to different numbers of rectangles/intervals are shown in Table 6.2.

N	*64*	*128*	*256*	*512*	*1024*
dc	43.5	43.4	42.7	42.8	42.3
ec	12.2	11.9	11.8	11.5	11.8
eq	1.6	0.8	0.4	0.2	0.1
ntpp	1.2	1.4	1.1	1.5	1.5
ntppi	1.2	1.4	1.1	1.5	1.5
po	35.6	34.8	37.4	36.4	36.7
tpp	2.4	3.1	2.8	3.1	3.1
tppi	2.4	3.1	2.8	3.1	3.1

N	*64*	*128*	*256*	*512*	*1024*
$<$	18.7	14.9	16.1	16.4	16.5
$=$	1.6	0.8	0.4	0.2	0.1
$>$	18.7	14.9	16.1	16.4	16.5
d	15.7	17.2	16.4	16.2	16.9
di	15.7	17.2	16.4	16.2	16.9
f	0.1	0.1	0.1	0.1	0.1
fi	0.1	0.1	0.1	0.1	0.1
m	0	0.1	0.1	0.1	0.1
mi	0	0.1	0.1	0.1	0.1
o	14.6	17.1	16.9	16.9	16.1
oi	14.6	17.1	16.9	16.9	16.1
s	0.1	0.2	0.1	0.1	0.1
si	0.1	0.2	0.1	0.1	0.1

Table 6.2: Relation proportions of *RCC-8* (on the left) and *IR-13* (on the right) regarding different numbers of rectangles/intervals *N=64, 128, 256, 512 or 1024*. All values are multiplied by 100.

6.4.2 Experiment Settings

We choose *HyperRotatE* [152] as the embedding model to learn subsymbolic representations of entities and relations, thanks to its ability of modeling the composition of relations (which is relevant to composition tables) and tree-like graph structures (which is useful for modeling transitive relations (e.g., *ntpp*)). This model also contains the three components mentioned in Section 6.2.2 and has a different scoring function. We use the original implementation of *HyperRotatE* to learn embeddings for entities and relations[2]. Hyper-parameters used for the *RCC-8* and the *IR-13* relations include learning rates: 0.05 (for the *RCC-8* relations) and 0.1 (for the *IR-13* relations), batch sizes: 1024 for both, negative samples: 64 (for the *RCC-8* relations) and 32 (for the *IR-13* relations), and dimensions: 110 (for the *RCC-8* relations) and 18 (for the *IR-13* relations). For *IR-13* relations, we use the same hyper-parameters for all synthetic data. For *RCC-8* relations, we increase the dimension of the embedding space to 200 when the number of entities is 512 or 1024[3]. In the experiment, we train *HyperRotatE* and then perform relation prediction over the *complete synthetic data* by default [4].

6.4.3 Evaluation Metrics

In order to quantify the differences between the learned relation graph and from CNGs, we introduce three metrics to measure commonality and difference. One solution is to convert graphs to sets of edges (each edge consists of a pair of relations) and to use set operations for quantification. Three metrics can be defined: (1) *False Recall* (i.e., number of false positives): the number of edges that are in our generated graph but not in CNGs (set difference). (2) *True Recall* (i.e., number of true positives): the number of edges that are in both our generated graph and CNGs (set intersection). (3) *Failed Recall* (i.e., number of false negatives): the number of edges that are not in our generated graph but in CNGs (set difference). Clearly, a graph that is similar to CNGs should have a low

[2] https://github.com/HazyResearch/KGEmb

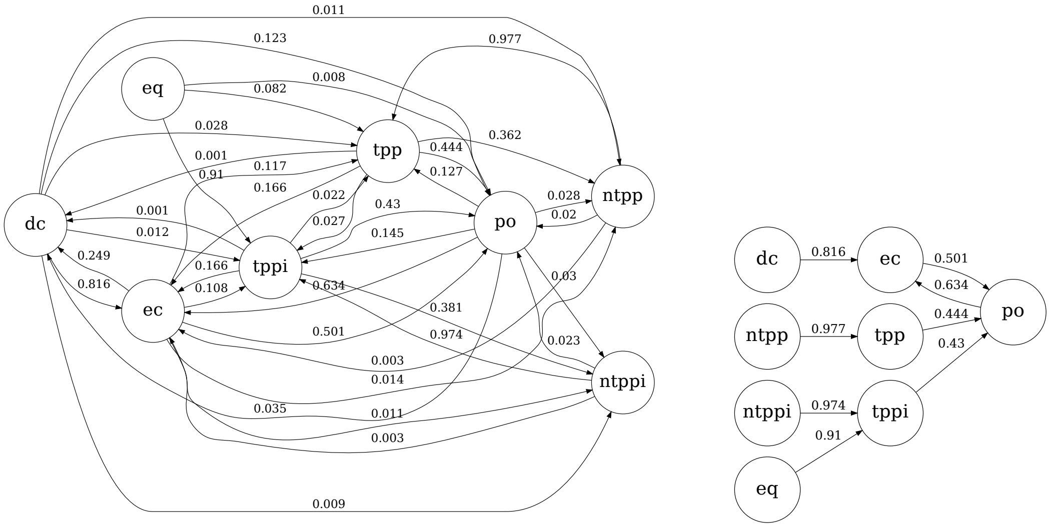
[3] When the number of entities increased to 512/1024, the model's performance greatly deteriorated. We assume the performance is compromised due to lack of learnable parameters. Thus, we increase the dimensions to provide more learnable parameters for our models to learn.

[4] Note that we do not tune these hyper-parameters but choose them by empirical experiences. It is worthwhile to investigate the impact of hyper-parameters on the experiment results in the future.

False Recall, a high *True Recall*, and a low *Failed Recall*.

6.4.4 Experimental Results

In this section, we first show direct results from our approach introduced in Section 6.3. Figure 6.2 illustrates (a) the initial relation graph resulting from the construction steps and (b) the refined relation graph after pruning. Next we report main findings based on the refined relation graph.



(a) Initial Relation Graph

(b) Refined Relation Graph

Figure 6.2: Examples of initial/refined relation graphs produced by our approach.

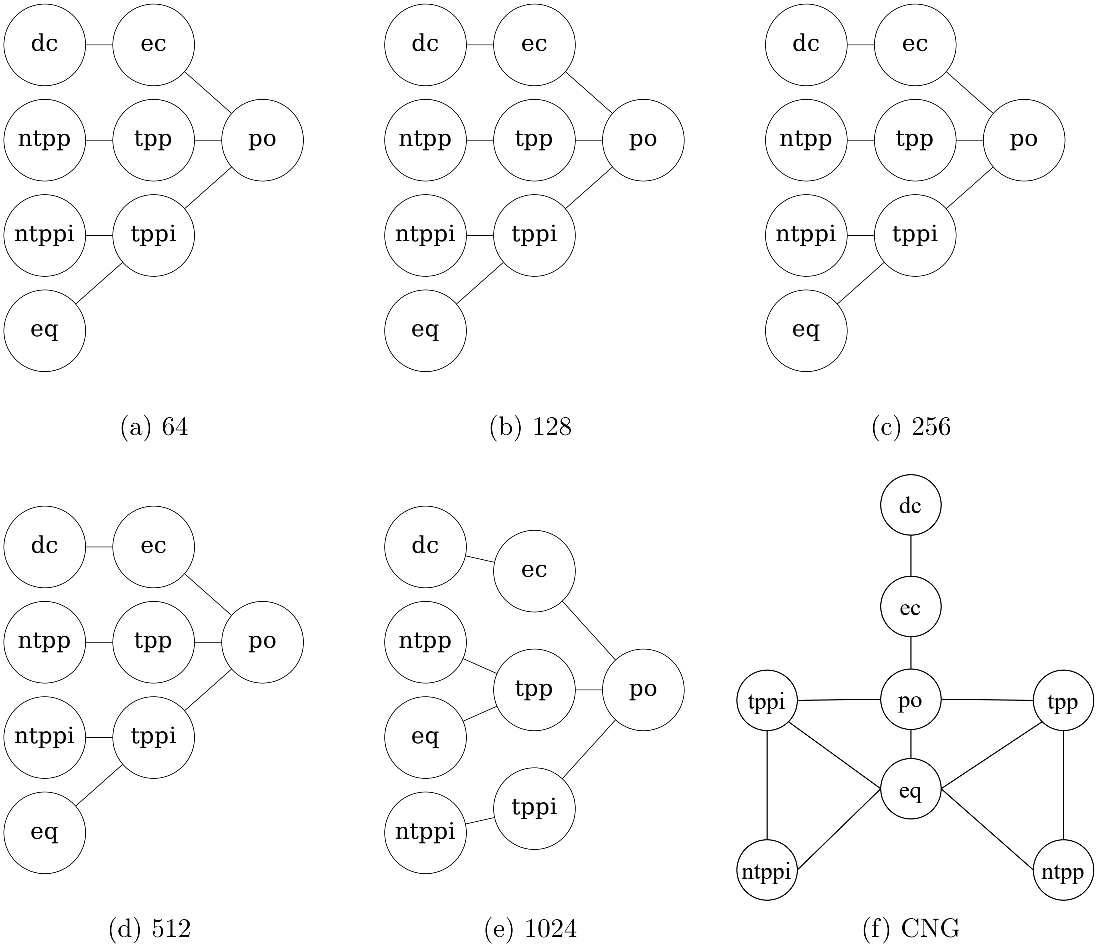
1. Relation graphs automatically discovered by our approach are well-aligned with CNGs for both *RCC-8* and *IR-13* relations.

Figure 6.2b implies that our refined relation graph resembles conceptual neighborhood graphs (Figure 6.3f). This motivates us to examine how similar our refined relation graphs are CNGs from the literature and whether this is merely a coincidence. In order to make our refined graphs comparable with CNGs, we convert the refined graphs into undirected and unweighted relation graphs (*UU-RGs*).

Figure 6.3 and Figure 6.4 report the results for *RCC-8* and *IR-13* relations, respectively, with different number of entities being considered. Noticeably, our approach discovers stable relation graphs for both *RCC-8* and *IR-13* relations. In specific, relation graphs for *RCC-8* remain almost unchanged with an increasing number of rectangles and relation graphs for *IR-13* begin to be fixed (except for the *equal* relation) when the number of intervals is 256. This observation also aligns with the statistics shown in Table 6.2, in which the relation proportions become relatively stable when the number of entities reaches 256. This indicates that the KGE model is mainly affected by the proportion of relations in the synthetic data. Moreover, by comparing Figure 6.3a, 6.3b, 6.3c, 6.3d and 6.3e with Figure 6.3f (or comparing Figure 6.4c, 6.4d and 6.4e with Figure 6.4f), we can observe that the discovered relation graphs are well-aligned with the CNGs which are defined in the literature (see Figure 6.3f and Figure 6.4f), except for differences around the *equal* relation(i.e., "eq" and "="). This observation demonstrates the ability of ML models in learning domain knowledge purely from data and the effectiveness of our approach in automatically discovering relationships of JEPD relations (*RCC-8* and *IR-13* as examples here). *This demonstrates that conceptual neighborhood graphs can be reproduced from data without any domain knowledge/inductive bias.*

As for the differences around the *equal* relation, one explanation is the lack of enough *equal* relations in our synthetic data. Because we randomly generate rectangles/intervals within a given area/segment, it is relatively rare to yield two rectangles/intervals that have the same geometry. As a result, most *equal* relations are just self-equivalent (e.g., $\langle s, eq, s \rangle$), which in fact does not provide enough useful information for the model to learn. Hence, we do not consider this a shortcoming of the model.

2. Similarities of relations are asymmetric and certain relations are more similar. Several patterns in asymmetric similarities of relations are also disclosed.



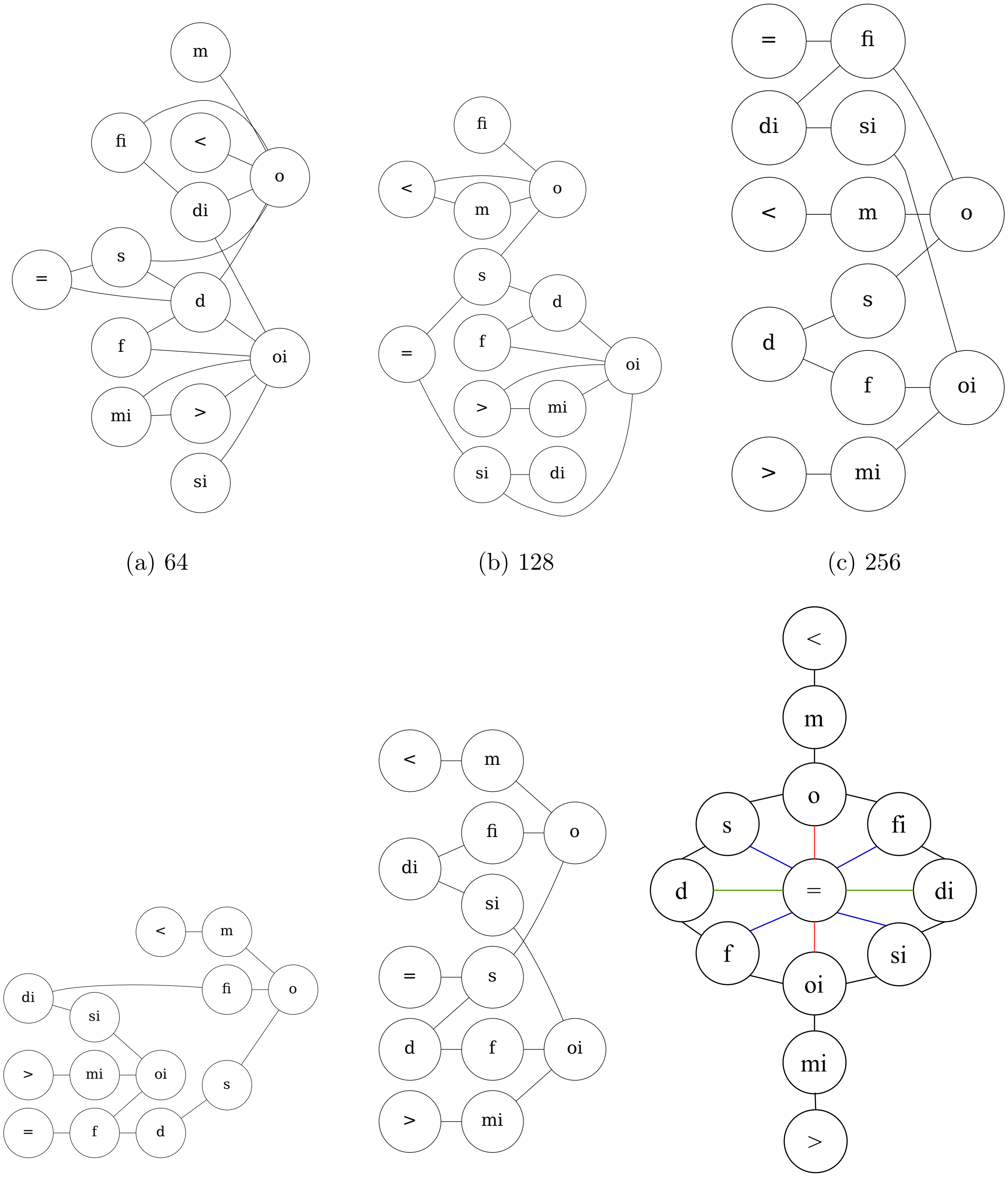
(a) 64 (b) 128 (c) 256

(d) 512 (e) 1024 (f) CNG

Figure 6.3: The relation graph of the RCC-8 relations w.r.t. different number of rectangles.

In this experiment, we examine similarities of relations, which are quantified by weights in Eq. 6.2. We extract a subgraph from our initial relation graphs (see Figure 6.2a) that contain edges presented in the theoretical CGNs except for edges that are connected to the *equal* relation (since the *equal* relation is not well-reproduced). We set the number of entities to 1024 and run the *HyperRotatE* model for 20 times to obtain average weights/similarity scores. The extracted subgraphs for *RCC-8* and *IR-13* relations are illustrated in Figure 6.5.

Apparently, we can observe that similarities of relations are *asymmetric*. In other



(a) 64

(b) 128

(c) 256

(d) 512

(e) 1024

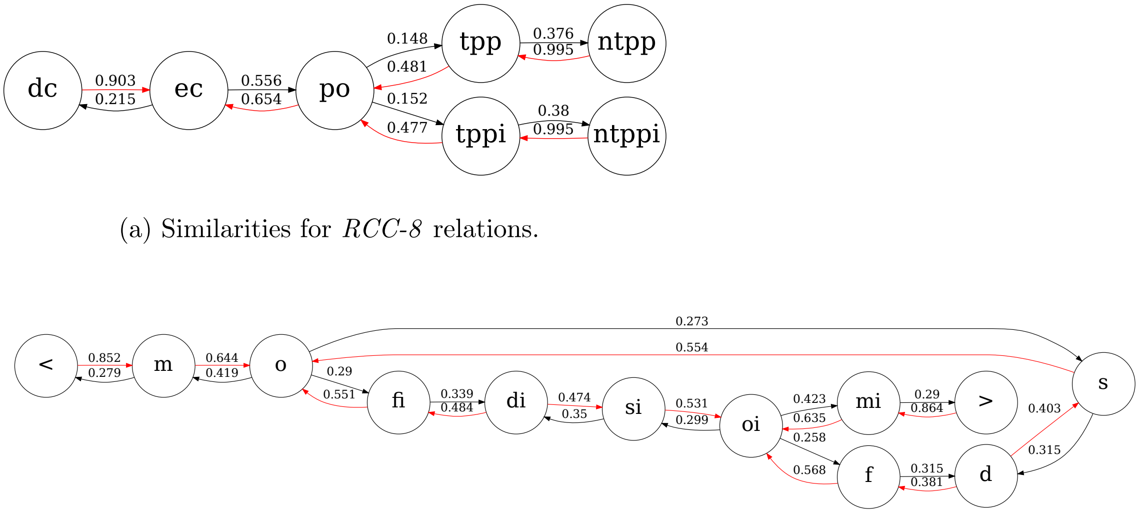
(f) CNG

Figure 6.4: The relation graph of the IR-13 relations w.r.t. different number of intervals.

words, the statement that a is similar to b differs from that b is similar to a (a and b are relations). For instance, the similarity between *dc* and *ec* is 0.903 while the inverse similarity is 0.215. Namely, *dc* is more similar to *ec* while *ec* is less similar to *dc*. In fact, Figure 6.5a shows *ec* is most similar to *po*, and both *dc* and *po* are most similar to *ec*. Meanwhile, we find that *ec* and *po* are more similar in general with higher similarities of 0.556 and 0.654. Additionally, there exist similar patterns between relations and their inverses in terms of their asymmetric similarities to other relations. For instance, Figure 6.5b shows $<$ is most similar to m and m is most similar to o. In terms of their inverse relations, $>$ is most similar to *mi* and *mi* is most similar to *oi*. Moreover, in Figure 6.5a, *ntpp* is most similar to *tpp* and *ntppi* is most similar to *tppi*. Similar patterns are shown between *d–>f–>oi* and *di–>fi–>o*, as well as between *d–>s->o* and *di–>si–>oi*. Another interesting observation is that all neighboring relations of the *overlapping* relation (i.e., *po* in *RCC-8* and *o* and *oi* in *IR-13*) are most similar to the *overlapping* relation (see the arrows that point to the *overlapping* relation). By contrast, in Figure 6.5b, both *d* and *di* are most similar to their neighboring relations (the red arrows around them leave out of them). Interestingly, similarity assessments in the cognitive science literature have been shown to be highly non-symmetric as well due to differences in (feature) alignment. For instance, Klippel et al. disclosed that the similarity between *RCC-8* relations vary from different scenarios (such as hurricane, cannon and geometry). In addition, Mark et al. also found that some topological relations indeed are conceptually more similar to others [172].

3. Even with limited training data (i.e., as low as 15% of the *complete synthetic data*), *HyperRotatE* is still capable of reproducing CNGs.

Finally, we are interested in the question of how much training data is needed for *HyperRotatE* to reproduce CNGs. In order to answer this question, we extract subsets of the *complete synthetic data* with different proportions and use the three metrics introduced

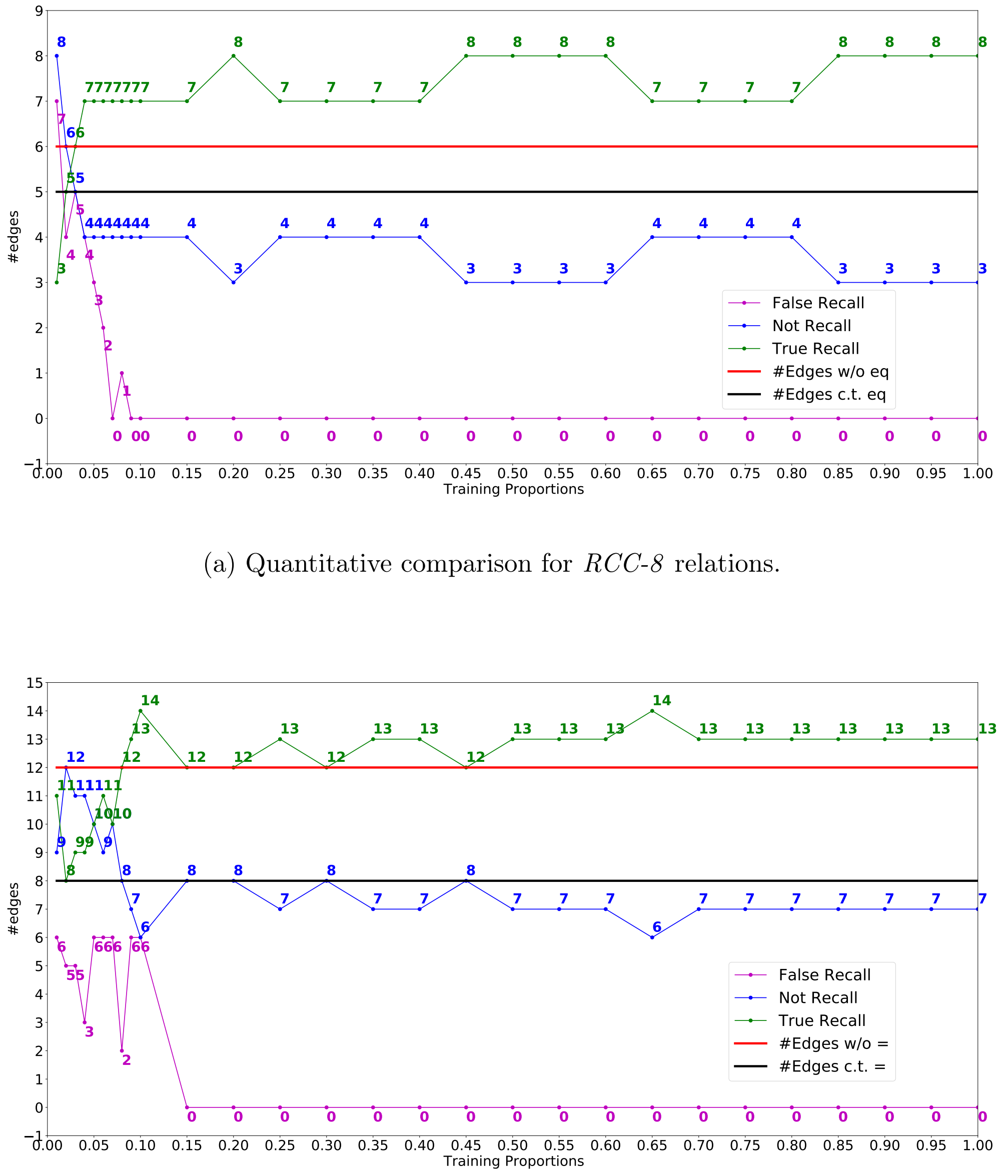


(a) Similarities for *RCC-8* relations.

(b) Similarities for *IR-13* relations.

Figure 6.5: Asymmetric similarities of relations. For two edges between two vertices, the edge with a larger weight is highlighted in red.

in Section 6.4.3 to evaluate the commonality and difference between *UU-RGs* and CNGs. Experiment results with the number of entities being 256 are shown in Figure 6.6. Red lines and black lines are theoretical references, indicating numbers of edges that are connected to the *equal* relation and that are not in theoretical CNGs, respectively. Clearly, regarding *RCC-8* relations, when more than 10% of the *complete synthetic data* are used for training, *HyperRotatE* is able to reproduce CNGs with stable recalls. Specifically, when the proportion is larger than 10%, *False Recall* continues to be 0, *True Recall* is either 7 or 8 and *Failed Recall* is either 3 or 4. Noticeably, *True Recall* is always above the red line (i.e., 6 – the theoretical number of edges that are not connected to "=" in the CNG) and *Not Recall* is close to the black line (i.e., 5 – the theoretical number of edges that are connected to "=" in the CNG). That is, the relation graphs (except for conceptual neighbors of the *equal* relation) is well-aligned with the theoretical CNGs even when only 10% of the *complete synthetic data* are available. Similar observations are shown for *IR-13* relations; see Figure 6.6b; however, the same pattern is observed when the training proportion is larger than 15%. In summary, *HyperRotatE* is a robust knowledge miner, which succeeds in discovering CNGs even with limited training data.



(a) Quantitative comparison for *RCC-8* relations.

(b) Quantitative comparison for *IR-13* relations.

Figure 6.6: Quantitative comparison between *UU-RGs* and CGNs. *UU-RGs* are reproduced w.r.t. different proportions of the *complete synthetic data* as training data. Lines in red denote the number of edges that are not connected to the *equal* relation in CNGs and lines in black denotes the opposite.

6.5 Conclusion

In this work, we presented a graph-based approach to examine similarities among *RCC-8* and *IR-13* relations in neighborhood graphs since they are important to spatio-temporal reasoning and spatial queries. In contrast to traditional approaches that heavily rely on top-down techniques and rule sets, we address this problem in a bottom-up manner without the need of any domain knowledge. Specifically, we focus on the task of relation prediction; namely to answer the query $\langle s, ?r, o\rangle$. Our rationale is that it would be difficult for machine learning methods to distinguish relations that are topologically similar when predicting missing relations between two entities. Therefore, we can pull similar relations out of the relation prediction task, and then use the proposed method to construct a graph to examine the structure among relations. Our experiments on synthetic data about *RCC-8* and *IR-13* relations reveal that (1) the extracted relation graphs are well-aligned with conceptual neighborhood graphs introduced in [128] and [117] except for neighboring relations of the *equal* relation. We believe this may be caused by a lack of enough *equal* relations in generated training data, which is left for future work; that (2) similarities of relations are asymmetric, and patterns in asymmetric similarities of relations are the same as those in their inverse relations; and that (3) the presented embedding models are robust in mining qualitative spatial and temporal knowledge (i.e., CNGs), even with limited training data.

Theoretically, our approach could be applied to any calculus with JEPD relations [159] to automatically discover CNGs. We believe our research would benefit theoretical studies of CNGs in general and contribute to a broader field, such as geospatial artificial intelligence, by promoting a deeper understanding of what machines really learn from data in a bottom-up manner. In the future, we plan to study whether such CNGs will be preserved when realistic data (particularly when non-spatial information is also considered) are used at training.

Chapter 7

Conclusions and Future Work

This chapter concludes this dissertation and points out research directions for further exploration. Firstly, it starts with a concrete summary of each chapter (including key questions, solutions and findings) as well as a discussion of the relationships among these chapters. Secondly, this chapter summarizes the potential contributions of this dissertation from both theoretical and practical perspectives. Finally, the chapter closed by identifying the potential limitations of my dissertation and lists research areas that would benefit from further exploration.

7.1 Summary and Discussions

This dissertation revisits areas related to common sense reasoning in Artificial Intelligence, namely temporal reasoning, and addresses the research problem of how to develop a subsymbolic temporal reasoning system. This proposal is mostly motivated by the fact that the world is ever-changing, and statements about the world change and grow over time. In the past few years, with advancements in data collection and information extraction, more and more time-related information has become available and is stored in large-scale KGs. Despite the sheer amount of information, temporal information may be incomplete, contain errors, and have diverse expressions. This imposes much burden on traditional temporal reasoning systems that are based on logic and algebra. Due to the symbolic nature of their representation and reasoning, they are subject to noise, scalability issues, and incompleteness.

The central idea of addressing challenges that traditional temporal reasoning systems are faced with is to find an alternative representation and reasoning method. Advanced techniques in machine learning and deep learning methods, representation learning/embedding techniques in particular, have made tremendous breakthroughs in natural language modeling, natural language understanding, image classification, and computer vision, etc. in the past decade. Most recently, they also successfully applied to generic knowledge graph-related downstream tasks. Key insights of these methods are to learn numeric/subsymbolic representations of things. Then the accomplishment of different downstream tasks amounts to performing various numeric operations over these representations. Extensive research has shown that such methods are noise-tolerant, easy to scale, and can handle incompleteness, which addresses most of the aforementioned challenges. Therefore, this subsymbolic paradigm for representation and task accomplishment provides a good potential to achieve a temporal reasoner. Following such a paradigm, a subsymbolic temporal reasoning system is developed step by step in this dissertation by analogy to the key components in a classic symbolic temporal reasoner.

More concretely, Chapter 3 focuses on sybsymbolic representations of time instants, which are chosen as the ontological primitive in this dissertation. Traffic prediction is used as the application case, since traffic data are usually timestamped (time instant), and time plays a critical role in traffic speed prediction. However, as for temporal information, traditional methods only take into account the sequential information of traffic data while ignoring the quantitative temporal information associated with each data point. I argue that this is insufficient and previous methods can only tackle traffic data of the same resolution. Instead, this dissertation makes full use of quantitative temporal information and learns numeric representations of time instants by considering the continuity and periodicity of traffic data. Because numeric representations can be easily integrated with each other, the learned representations for time instants can be directly combined with numeric traffic data as temporal signals for traffic prediction. Such a direct combination results in an improvement in performance compared to previous methods. Meanwhile, this result demonstrates the efficacy of learning representations for time instants. Chapter 4 focuses on subsymbolic representations of time intervals (open intervals, semi-closed intervals, and closed intervals) and presents a subsymbolic method for quantitative temporal reasoning. It addresses a number of representational and reasoning issues, such as how to represent intervals subsymbolically based on the subsymbolic representation of the ontological primitive, how to link temporal information with statements and how to reason about time in a subsymbolical manner. In addition, the proposed method is tested on two downstream tasks by using two datasets: temporal link prediction and temporal scoping prediction. Experimental results show that the proposed method can boost performance on both datasets in terms of these two downstream tasks. This work presents a complete workflow for developing a subsymbolic reasoning system for quantitative time. Chapter 5, by contrast, focuses on qualitative temporal reasoning and presents a temporally-explicit subsymbolic approach. The design of this approach is motivated by general theories of time developed in classic symbolic temporal reasoning (such as composition tables), and properties of temporal relations (such as transitivity, symmetricity, etc.). In order to empower the subsymbolic method to automatically discover these prior knowledge from data, quaternions are introduced to hyperbolic space to fulfill representational and reasoning purposes. The proposed method has been successfully applied to qualitative temporal reasoning as well as qualitative spatial reasoning. Chapter 6 answers a follow-up question raised from experimental results in Chapter 5, namely why subsymbolic methods perform better than traditional symbolic methods in terms of qualitative spatial and temporal reasoning? My assumption is that the subsymbolic method must have learned some domain theory purely from data. Since two primary reasoning mechanisms - composition tables and conceptual neighborhood structures - both focus on exploring relationships between relations for facilitating reasoning, a graph-based method is proposed as an initial attempt to examine similarities between qualitative relations. Experimental results are well-aligned with conceptual neighborhood structures of qualitative relations. This work can not only help us interpret why machine learning methods work but also contribute to theory verification/discovery from a machine learning-driven perspective.

In summary, this dissertation points out the need for new approaches to temporal reasoning by pointing out the limitations of traditional reasoning systems. It proposes to develop a subsymbolic temporal reasoning system and identifies the key ingredients to implement such a system. By decomposing this task into a few separated but correlated research questions, each chapter in this dissertation contributes to answering these research questions. In the end, the success of sub-symbolic reasoning systems is interpreted.

7.2 Research Contributions

This dissertation first points out that conventional symbolic temporal reasoning systems are faced with several limitations, e.g., prone to noise and incompleteness. Motivated by great breakthroughs made by machine learning/deep learning, in particular representation learning/embedding techniques, in the past few years, this dissertation aims to leverage such techniques to find new potentials for temporal reasoning. The key contribution of this dissertation lies in the idea of developing a subsymbolic temporal reasoning system. This dissertation accomplishes such a goal by decomposing it into separate but related research questions and addresses each of them step by step. The following two subsections summarize the theoretical contributions and practical implications of this dissertation, respectively.

7.2.1 Theoretical Contributions

This dissertation has contributed to the theoretical study of time under the context of machine learning and deep learning. I summarize the four most important contributions as below.

The idea of developing subsymbolic temporal reasoning systems. By analyzing the limitations of symbolic temporal reasoning systems, this dissertation comes up with a new idea of developing a temporal reasoning system by using machine learning/deep learning methods, particularly representation learning/embedding techniques. By analogy to key components in a symbolic temporal reasoning system, this dissertation claims that the five components described below are the ingredients for developing a subsymbolic temporal reasoning system. It includes choosing ontological primitives, learning numeric (subsymbolic) representations of the chosen ontological primitive, modeling representations of other temporal information using the primitive, linking time with atemporal statements (time formalization), and devising methods to perform subsymbolic temporal reasoning. These different parts as well as solutions for each part are discussed throughout the entire dissertation. This dissertation as a whole provides a concrete solution for developing subsymbolic temporal reasoning systems, which will lay the theoretical foundation for future study in this direction.

A subsymbolic approach to quantitative temporal reasoning. This dissertation provides a broader definition of temporal knowledge graphs, which clarifies the study subject for temporal reasoning under the context of deep learning. Chapter 3 demonstrates the efficacy of learning numeric representations of time instants using embedding techniques. Furthermore, Chapter 4 devises a concrete subsymbolic approach to implementing a subsymbolic temporal reasoning system by figuring out the five ingredients that constitute a subsymbolic time reasoning system step by step. Although there are different ways to implement such a reasoning system, this proposed approach is more generic and can be applied to KGs that contain any kind of temporal and non-temporal statements.

A subsymbolic approach to qualitative temporal reasoning. Traditionally, the success of qualitative temporal reasoning lies in the application of logic and algebra calculus. This dissertation assumes that such prior knowledge will also benefit machine learning methods to perform temporal reasoning. By considering general theories in time (e.g., transitivity/composition tables) and properties of qualitative temporal relations (e.g., transitivity, symmetricity and inverse relations), Chapter 5 presents a theory-informed subsymbolic approach to capture such priors. Theoretically, the method can be applied to any other type of qualitative reasoning, as long as the data involved can be represented in the form of triples. For example, this dissertation has proven its effectiveness in qualitative spatial reasoning.

Automatic discovery of conceptual neighborhood structures. Commonly, the development of a theory demands four stages, including tension, search, elaboration and proclamation [173]. During this process, research scientists are heavily involved and play various roles (as creator, codifier, carrier, researcher and advocate). Such a process requires long-term significant investment of time and labor. Instead, chapter 6 presents a machine/deep learning-driven method for discovering conceptual neighborhood structures for qualitative relations. It follows a common paradigm for theory discovery, made up of hypothesis formalization, diagnosis of individual cases and hypothesis testing [174], though only two types of qualitative reasoning are diagnosed. This method successfully re-discovers conceptual neighborhood structures for both qualitative spatial and temporal relations. Theoretically, such an idea can be applied to any other JPED relations to find conceptual neighborhood structures.

7.2.2 Practical Contributions

This work can also contribute to practical problems, which are listed in the following.

Automatic Knowledge Graph Construction. Although automatic knowledge graph construction primarily relies on information extraction techniques that directly apply to natural language texts [175, 176, 177]. The task of knowledge graph completion can contribute a lot as well once an initial knowledge graph is ready [178]. Chapter 4 presents a generic method that could be applied to any generic knowledge graph to predict missing statements as well as missing validity periods of statements. This method can be complementary to the mainstream automatic construction methods, particularly in terms of providing temporal information. Evidences [179, 180, 181] show it is usually hard to accurately exact temporal information from texts due to its diverse expressions used in different contexts.

Query/Question Answering System. Chapter 4 and Chapter 5 presents two subsymbolic methods for quantitative and qualitative temporal reasoning, respectively. They are complementary and can be integrated easily to implement a temporal query answering system [182, 183]. The system is able to answer quantitative/qualitative temporal queries and non-temporal queries. Furthermore, this query-answering system can be combined with other query-answering systems, which are also based on subsymbolic reasoning methods, to build an even larger query-answering system.

Time-Aware Recommendation System. Users' preference/inclination for items usually changes over time (e.g., people's tastes on music/news) due to personal issues or external driving force [184, 185, 186]. It is essential to capture such changes in order to recommend correct items to correct users timely. By using time representations proposed in Chapter 3, time-aware recommendation system can be developed to track users' habits and detect changes in their preferences. Such time-aware recommendation systems could be useful for news, movie, location, route,s and food recommendation, to name a few.

Time Series Prediction. Chapter 3 presents a good example of how to apply subsymbolic representations of time instants to traffic prediction. As a matter of fact, the same idea can be generalized to any time series predictions. Unlike prior time series prediction, we can directly incorporate quantitative temporal information into prediction models and tackle temporal patterns at different resolutions (e.g., weekly, seasonal, monthly patterns). In order to deal with the issue of diverse temporal patterns, we could learn numeric representations of time at each resolution and integrate them to obtain a mixed representation for later usage. This idea opens a new door to time series prediction.

7.3 Limitations and Future Work

In this section, I outline the limitations of this dissertation and point out research areas that deserve further investigation.

The choice of ontological primitives. In this dissertation, time instants are chosen as the ontological primitive for time, because representation learning methods (e.g., embedding techniques) aim to learn numeric representations for *discrete/categorical things*. However, the computational bottleneck should not be the only factor to consider when we decide on ontological primitives. One factor that deserves consideration is the difficulty of using the chose ontological primitive to represent the other temporal information. In this dissertation, it is challenging to obtain numeric representations of time intervals when time instants are chosen as the primitive. Even though we might agree intervals can be modelled by a set of consecutive instants conceptually (scoped by start instant and end instant), how to aggregate representations of such instants to obtain numeric representations of intervals (both closed and semi-closed intervals) lacks a thorough study. Another factor to consider should be thediscovery resolution of time instants (e.g., years, months, minutes). The resolution used in this dissertation varies from one chapter to another. However, this is not ideal, given that a real-life dataset usually contains temporal information of different resolutions. This calls for a more generic representation learning method for quantitative time.

Impact of time formalization of different kinds. Section 2.2.3 introduces different ways of incorporating time into statements for symbolic temporal query and reasoning purposes, and emphasizes on their distinct impacts on query performance, data storage and maintenance, etc. In this dissertation, I come up with one simple solution to link time with statements in the context of subsymbolic temporal reasoning, namely regarding temporal information as another relation held between entities. Such solution, though straightforward, decomposes a quadruple (<*subject, predicate, object, time*>) into two independent triples (<*subject, predicate, object*> and <*subject, time, object*>). It breaks up the semantics among them. Therefore, this needs improvements and other possibilities deserve exploration. In addition, it would also be interesting to investigate how different ways of linking time with statements will affect the reasoning performance of different subsymbolic temporal reasoning methods. This question is also related to a broader research question that how syntax/structures of inputs will impact representation learning methods.

Undoing learning to achieve non-monotonic reasoning. Statements about the world change as time goes by, and, thus, they usually only hold during a certain period of time. We view statements as temporally scoped statements in this dissertation. However, there are other perspectives. For example, we can also treat them as data streams, and each of them can have two states (corresponding to the start instant and end instant of its validity period). One state indicates it starts to be valid when it appears, therefore the information it carries should be added to a subsymbolic reasoning model. The other state indicates it is not valid anymore when it is falsified or turns invalid. In this case, the information it brings to the reasoning model should be canceled out. Although it may seem reasonable, this requires a reasoning model to be non-monotonic. However, most machine learning/deep learning models conform to a repetitive training process and are optimized iteratively over training datasets. That said, they cannot instantaneously undo what they have learned from previous statements, because all the information that has been in the reasoning model is interconnected to some degree and training data are used repeatedly. Despite these challenges, figuring out how to undo learning to achieve a non-monotonic reasoning model is definitely a fascinating research direction.

Machine learning-driven theory discovery. One of the theoretical contributions of this dissertation is that it demonstrates that conceptual neighborhood structures of any kind of JPED relations can in principle be discovered by machine learning/deep learning methods purely from data. However, this is just a starting point. The success of conventional symbolic (temporal) reasoning lies in the application of various logic calculus, such as composition tables. Then a follow-up question, which is even more intriguing, is whether it is possible that machine learning/deep learning methods have discovered them as well and utilize them implicitly to help perform reasoning. If this assumption is reasonable, how could we test our hypothesis? I believe that this research direction could contribute to both explainable AI and theory discovery/verification.

Appendix A

A.1 Data Statistics

Generation of WIKIDATA114k We extracted a sport-centric subgraph from WIKIDADA432k. We first picked out statements where the relation *memberOfSportsTeam* appears and obtained an entity set from those statements. Then we find all the statements that entities obtained from the previous step participate in as our initial subgraph. Finally, we ensure that each entity/relation is associated with at least 5 statements and the time period is restricted to [1883, 2023] for temporal statements, which encloses most of the temporal statements in the initial subgraph. This results in 1.7 million statements with 114k entities and 126 relations, and thus named as WIKIDATA114k. See Table A.1 for data statistics.

A.2 Hyperparameter Settings

We tune models by the MRR on the validation set. Grid search is performed over negative samples k = [16, 32, 64, 128], learning rate lr = [0.003, 0.002, 0.001], batch size b = [1500, 2000, 2500, 3000, 3500]; dimension d = [200, 300, 400], and weight for time smoothness regularizer β = [0.0, 0.1, 0.001, 0.0001], as shown in Table A.2.[1] We find that effects of different hyperparameters are minimal except for learning rate as the trained model usually converge to similar MRRs as long as they are trained thoroughly. We also observe that time smoothness regularizer is useful in learning time embeddings on WIKIDATA12k while failing to improve the model on WIKIDATA114k. This may be due to data sparsity with regard to time. As the time span of WIKIDATA114k is much smaller, time information is intensive and thus models are capable of learning temporal order between timestamps implicitly.

[1] Experiments are terminated after 10000 steps.

		WIKIDATA12k	WIKIDATA114k
#entities		12,544	114,351
#relations		24	126
time period		[19, 2020]	[1883, 2023]
train	#all	32,497	1,670,969
	#time instant	14,099	175,637
	#start time only	4,089	44,809
	#end time only	1,273	2,164
	#full time interval	13,035	402,135
	#no time	0	1,046,224
valid	#all	4,051	11,720
	#time instant	1,857	1,177
	#start time only	322	342
	#end time only	76	11
	#full time interval	1,796	2,655
	#no time	0	7,535
#test	#all	4,043	11,854
	#time instant	1,844	1,219
	#start time only	324	306
	#end time only	56	15
	#full time interval	1,819	2,790
	#no time	0	7,524

Table A.1: Statistics of these datasets used.

	#negative samples				# learning rate		
	16	32	64	128	0.003	0.002	0.001
MRR	36.02	36.68	37.06	37.30	36.64	36.82	37.30
MR	97	100	98	101	126	103	101
HITS@1	25.96	26.81	27.25	27.38	26.83	26.73	27.38
	#batch size				# dimension		
	2000	2500	3000	3500	200	300	400
MRR	36.71	36.87	37.30	36.78	36.12	36.88	37.30
MR	100	114	101	100	106	101	101
HITS@1	26.59	26.88	27.38	26.77	25.98	26.88	27.38

Table A.2: Effects of hyper-parameters on WIKIDATA12k

A.3 Experimental Setup

Upon inspection on implementations of TKBC models, we find there are two common issues.

First, SOTAs only learn time embeddings for timestamps that appear in training set, which would be problematic at testing. For instance, suppose a sorted (ascending) list of timestamps occurring in training set is [1540, 1569, 1788, 1789, 1790], SOTAs only learn embeddings for these timestamps, while ignoring intermediate timestamps. As a result, they cannot answer queries when the associated time is not in the list, such as (s, r, ?o, *1955*). This problem would be even worse regarding time interval generation. As when we need to grow a time point to a time interval by extending it to the left or the right, we may jump from one year to a year far away from it. For instance, from 1569 to 1540 (left) or 1788 (right). This is not reasonable and thus may severely affect the evaluation on time prediction. In order to address this issue, we enumerate all the time points in the time span of the training set with a fixed granularity (i.e., year) and use them for all models at training periods.

The other issue is about the evaluation of link prediction task on time interval-based statements (including closed interval-based and left/right-open interval-based statements). In existing works, the evaluation boils down to assessing the correctness of answering a timestamp-based query by randomly picking one timestamp from a set of timestamps within the time interval and then measuring the performance on the newly generated query (i.e., the timestamp-based query). However, this is problematic. For closed interval-based samples, the evaluation results may vary from randomly sampled timestamps and thus may not be stable. For left/right-open interval-based statements, it is more severe. For instance, for a left-open interval-based test sample *(Albert Einstein, educatedAt, ?o, [-, 1905])*, [110] randomly pick a year before 1905, say 1000, and evaluate whether a model can output the correct answer (*University of Zurich*) to the new query *(Albert Einstein, educatedAt, ?o, 1000)*. Clearly, there is no correct answer at all since he was born in 1879. Therefore, the evaluation on such test samples may not be plausible. In order to address these issues, for a closed interval-based sample, we enumerate all the time points in the interval and do evaluation on each time point separately. Then we use the average performance over them as the overall evaluation. For the latter, we only consider the known endpoint in an interval, namely $(s, r, ?o, st)$ for right-open cases and $(s, r, ?o, et)$ for left-open cases.

A.4 Link Prediction Performance by types of validity information

Table A.3 shows the comparison between different methods in terms of different types of validity information.

Datasets	WIKIDATA12k							
Types	Time Interval (O)		Time Interval (C)		Time Instant		No Time	
Methods	TIMEPLEX base	TIME2BOX	TIMEPLEX base	TIME2BOX	TIMEPLEX base	TIME2BOX	TIMEPLEX base	TIME2BOX
MRR	46.74	**51.48**	25.30	**28.44**	41.11	**43.13**	-	-
MR	203	**68**	273	**84**	350	**125**	-	-
HITS@1	19.21	**41.05**	11.54	**18.5**	32.6	**33.30**	-	-
Datasets	WIKIDATA114k							
Types	Time Interval (O)		Time Interval (C)		Time Instant		No Time	
Methods	TIMEPLEX	TIME2BOX	TIMEPLEX	TIME2BOX	TIMEPLEX	TIME2BOX	TIMEPLEX	TIME2BOX
MRR	**22.63**	22.43	17.72	**18.85**	20.81	**21.32**	67.85	**68.40**
MR	346	**168**	155	**147**	**176**	**193**	430	**172**
HITS@1	4.98	**11.21**	3.94	**8.35**	11.07	**11.16**	**61.52**	60.30

Table A.3: Link prediction evaluation by types of validity information. Time Interval (O) denotes left/right-open interval-based statements, and Time Interval (C) refers to closed interval-based statements.

A.5 Time Prediction Performance by duration length

Table A.4 and A.5 compare the performance of TIMEPLEX and TIME2BOX on the time prediction task across different duration lengths on two datasets. Test samples are first classified into three groups by duration (du) and then evaluate the performance of each group. For an interval I, $du = I_{max} - I_{min} + 1$. It shows that our improvements are more pronounced in terms of shorter durations in general.

	WIKIDATA12k					
Duration (du)	du=1		1<du<=5		du>5	
Method	TIMEPLEX base	TIME2BOX	TIMEPLEX base	TIME2BOX	TIMEPLEX base	TIME2BOX
gIOU@1	30.29	**38.09**	39.51	**43.68**	**47.4**	46.99
aeIOU@1	20.84	**28.34**	15.86	**22.95**	**18.23**	13.20
gaeIOU@1	12.47	**18.62**	11.73	**16.34**	**16.85**	11.20

Table A.4: Time prediction by duration on WIKIDATA12k

	WIKIDATA114k					
Duration (du)	du=1		1<du<=5		du>5	
Method	TIMEPLEX base	TIME2BOX	TIMEPLEX base	TIME2BOX	TIMEPLEX base	TIME2BOX
gIOU@1	28.75	**37.03**	29.77	**38.36**	27.99	**39.07**
aeIOU@1	25.80	**34.16**	16.52	**21.54**	7.09	**9.94**
gaeIOU@1	14.69	**21.08**	10.50	**14.50**	3.85	**7.02**

Table A.5: Time prediction by duration on WIKIDATA114k

A.6 Model Parameter Comparison

Table A.6 summarizes the number of parameters used in each method.

Models	**Number of parameters**
TNTComplex	2d($\|E\|$ + $\|T\|$ + 4$\|R\|$)
TIMEPLEX base	2d($\|E\|$ + $\|T\|$ + 6$\|R\|$)
TIME2BOX	d($\|E\|$ + 2$\|T\|$ + 2$\|R\|$) + 4d^2

Table A.6: Number of parameters for each model

Appendix B

B.1 Some proofs

B.1.1 Unitary Quaternion

$$q_u = cos(\alpha) + sin(\alpha)cos(\theta_1)cos(\theta_2)i + sin(\alpha)cos(\theta_1)sin(\theta_2)j + sin(\alpha)sin(\theta_1)k \quad \text{(B.1)}$$

$$\begin{aligned}\|q_u\| &= \sqrt{cos(\alpha)^2 + (sin(\alpha)cos(\theta_1)cos(\theta_2))^2 + (sin(\alpha)cos(\theta_1)sin(\theta_2))^2 + (sin(\alpha)sin(\theta_1))^2} \\ &= \sqrt{cos(\alpha)^2 + (sin(\alpha)cos(\theta_1))^2 + (sin(\alpha)sin(\theta_1))^2} \\ &= \sqrt{cos(\alpha)^2 + sin(\alpha)^2} = 1.\end{aligned}$$

B.2 Number of Learnable Parameters

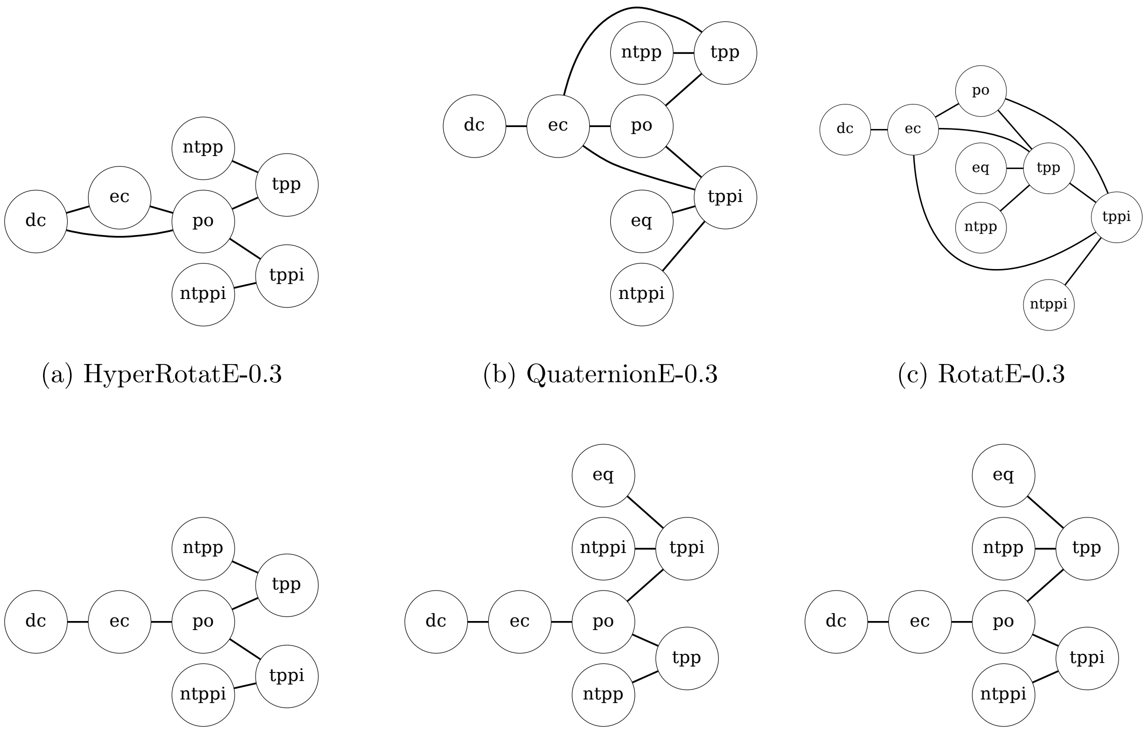
Model	#parameters
HyperQuaternionE	$d * \|E\| + (d+1) * \|R\|$
HyperRotatE	$(d+1) * \|E\| + (3d+1) * \|R\|$
QuaternionE	$d * \|E\| + d * \|R\|$
RotatE	$2d * \|E\| + d * \|R\|$

Table B.1: Number of parameters in each model. $|E|$ and $|R|$ are number of entities and relations, respectively.

B.3 Network structures by different embedding models

Here we compare network structures yielded by different embedding models. Note that since there is no practical guideline on how to determine thresholds to extract closely-connected substructures, we choose threshold=0.3 and threshold=0.4 empirically.

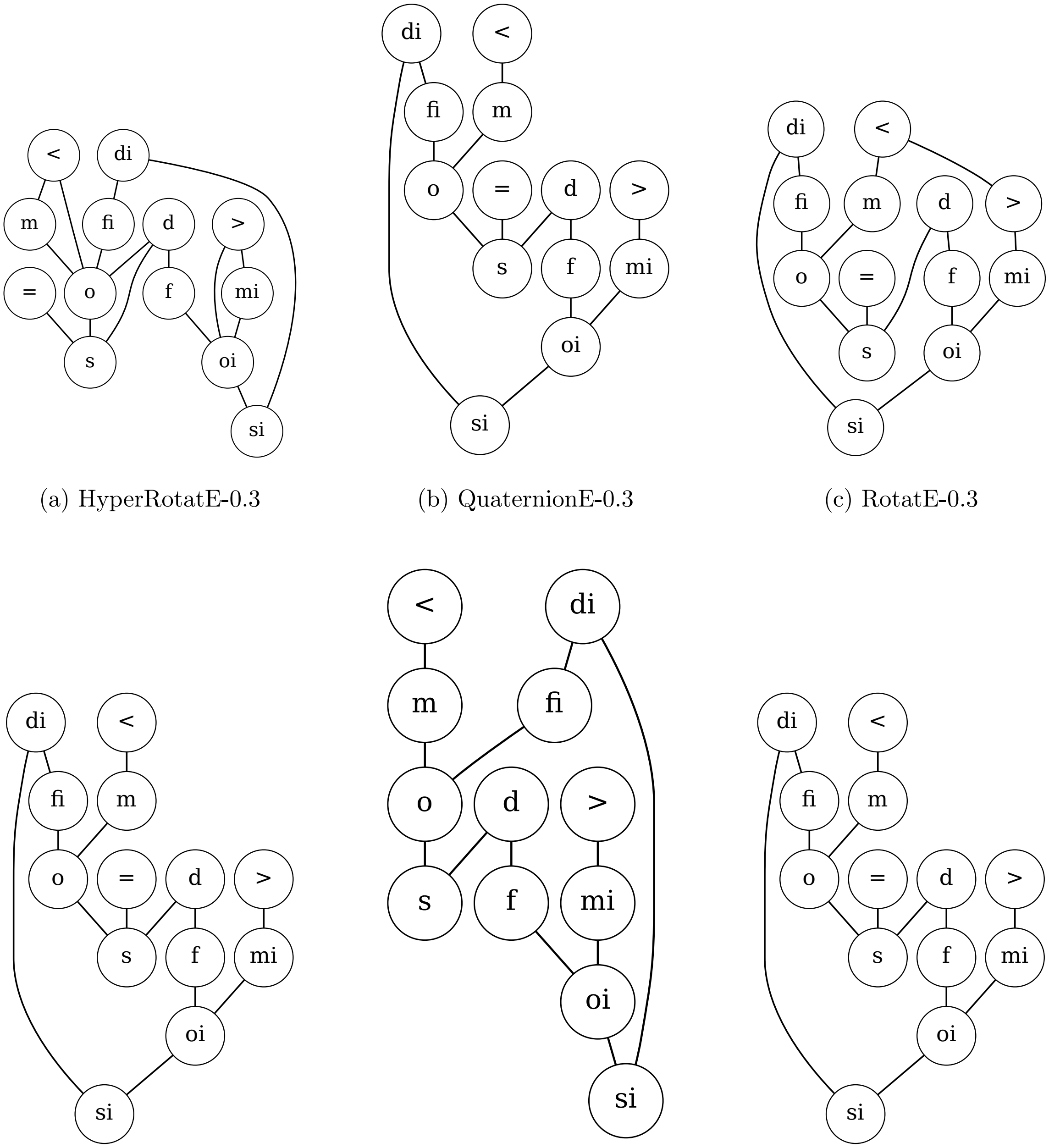
In general, according to Figure B.1 and B.2, we can conclude that all embedding models are capable of implicitly learning neighborhood structures of relations with nuanced differences. By comparing Figure B.1d, B.1e, B.1f with Figure 5.13b, we can find that our model yields a better structure as part of the substructure around *eq* is discovered successfully while others fail to do so. For network structures of temporal relations (Figure B.2e, B.2e and 5.14b), they all exhibit much similarity except for small differences around =, which is partly attributed to a lack of *equal* relations in datasets. However, as discussed in Section 5.5.4, our model in fact discovers the inner structure around =, which is filtered out by thresholds yet. Therefore, our model is superior to other embedding models in discovering relationships between relations.



(a) HyperRotatE-0.3 (b) QuaternionE-0.3 (c) RotatE-0.3

(d) HyperRotatE-0.4 (e) QuaternionE-0.4 (f) RotatE-0.4

Figure B.1: Network structures by different models (spatial relations).



(a) HyperRotatE-0.3

(b) QuaternionE-0.3

(c) RotatE-0.3

(d) HyperRotatE-0.4

(e) QuaternionE-0.4

(f) RotatE-0.4

Figure B.2: Network structures by different models (temporal relations).

Bibliography

[1] M. F. Goodchild, *A geographer looks at spatial information theory*, in *International conference on spatial information theory*, pp. 1–13, Springer, 2001.

[2] M. F. Goodchild and P. A. Longley, *Geographic information science*, in *Handbook of Regional Science*, pp. 1597–1614. Springer, 2021.

[3] C. Claramunt, *Ontologies for geospatial information: Progress and challenges ahead*, *Journal of spatial information science-JOSIS* (2020), no. 20 35–41.

[4] K. Janowicz, *Observation-driven geo-ontology engineering*, *Transactions in GIS* **16** (2012), no. 3 351–374.

[5] H. Couclelis, *Ontologies of geographic information*, *International Journal of Geographical Information Science* **24** (2010), no. 12 1785–1809.

[6] M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg, J.-W. Boiten, L. B. da Silva Santos, P. E. Bourne, *et. al.*, *The fair guiding principles for scientific data management and stewardship*, *Scientific data* **3** (2016), no. 1 1–9.

[7] S. Vere, *Temporal scope of assertions and window cutoff.*, in *IJCAI*, vol. 85, pp. 1055–1059, Citeseer, 1985.

[8] A. Rula, M. Palmonari, S. Rubinacci, A.-C. Ngonga Ngomo, J. Lehmann, A. Maurino, and D. Esteves, *Tisco: Temporal scoping of facts*, in *Companion Proceedings of The 2019 World Wide Web Conference*, pp. 959–960, 2019.

[9] O. Etzioni, M. Banko, S. Soderland, and D. S. Weld, *Open information extraction from the web*, *Communications of the ACM* **51** (2008), no. 12 68–74.

[10] X. Ling and D. S. Weld, *Temporal information extraction*, in *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

[11] Z. Wang, J. Zhang, J. Feng, and Z. Chen, *Knowledge graph embedding by translating on hyperplanes*, in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 28, 2014.

[12] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel, *Convolutional 2d knowledge graph embeddings*, in *Thirty-second AAAI conference on artificial intelligence*, 2018.

[13] I. Balažević, C. Allen, and T. M. Hospedales, *Hypernetwork knowledge graph embeddings*, in *International Conference on Artificial Neural Networks*, pp. 553–565, Springer, 2019.

[14] W. Xiong, T. Hoang, and W. Y. Wang, *Deeppath: A reinforcement learning method for knowledge graph reasoning*, in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 564–573, 2017.

[15] L. Guo, Z. Sun, and W. Hu, *Learning to exploit long-term relational dependencies in knowledge graphs*, in *International Conference on Machine Learning*, pp. 2505–2514, PMLR, 2019.

[16] Y. Lin, Z. Liu, H. Luan, M. Sun, S. Rao, and S. Liu, *Modeling relation paths for representation learning of knowledge bases*, in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 705–714, 2015.

[17] 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.

[18] V. Piuri, *Analysis of fault tolerance in artificial neural networks*, *Journal of Parallel and Distributed Computing* **61** (2001), no. 1 18–48.

[19] S. Guo, Q. Wang, L. Wang, B. Wang, and L. Guo, *Knowledge graph embedding with iterative guidance from soft rules*, in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, 2018.

[20] X. Wang, X. He, Y. Cao, M. Liu, and T.-S. Chua, *Kgat: Knowledge graph attention network for recommendation*, in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 950–958, 2019.

[21] X. Huang, J. Zhang, D. Li, and P. Li, *Knowledge graph embedding based question answering*, in *Proceedings of the twelfth ACM international conference on web search and data mining*, pp. 105–113, 2019.

[22] W. Liu, P. Zhou, Z. Zhao, Z. Wang, Q. Ju, H. Deng, and P. Wang, *K-bert: Enabling language representation with knowledge graph*, in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, pp. 2901–2908, 2020.

[23] G. Brewka, *Nonmonotonic reasoning: logical foundations of commonsense*, vol. 12. Cambridge University Press, 1991.

[24] L. Vila, *A survey on temporal reasoning in artificial intelligence*, *Ai Communications* **7** (1994), no. 1 4–28.

[25] J. McCarthy and P. J. Hayes, *Some philosophical problems from the standpoint of artificial intelligence*, in *Readings in artificial intelligence*, pp. 431–450. Elsevier, 1981.

[26] D. McDermott, *A temporal logic for reasoning about processes and plans*, *Cognitive science* **6** (1982), no. 2 101–155.

[27] J. F. Allen, *Towards a general theory of action and time*, *Artificial intelligence* **23** (1984), no. 2 123–154.

[28] M. B. Vilain, *A system for reasoning about time.*, in *AAAI*, vol. 82, pp. 197–201, 1982.

[29] J. F. Allen, *Maintaining knowledge about temporal intervals*, *Communications of the ACM* **26** (1983), no. 11 832–843.

[30] J. F. Allen and P. J. Hayes, *A common-sense theory of time.*, in *IJCAI*, vol. 85, pp. 528–531, Citeseer, 1985.

[31] J. Van Benthem, *The logic of time: a model-theoretic investigation into the varieties of temporal ontology and temporal discourse*, vol. 156. Springer Science & Business Media, 2013.

[32] B. Russell, *The principles of mathematics.* Routledge, 2020.

[33] B. A. Haugh, *Non-standard semantics for the method of temporal arguments.*, in *IJCAI*, pp. 449–455, 1987.

[34] S. Kripke, *Semantical considerations of the modal logic*, *Studia Philosophica* **1** (2007).

[35] R. McArthur, *Tense logic*, vol. 111. Springer Science & Business Media, 2013.

[36] R. Kowalski and M. Sergot, *A logic-based calculus of events*, in *Foundations of knowledge base management*, pp. 23–55. Springer, 1989.

[37] T. L. Dean and D. V. McDermott, *Temporal data base management*, *Artificial intelligence* **32** (1987), no. 1 1–55.

[38] K. Kahn and G. A. Gorry, *Mechanizing temporal knowledge*, *Artificial intelligence* **9** (1977), no. 1 87–108.

[39] B. C. Bruce, *A model for temporal references and its application in a question answering program*, *Artificial intelligence* **3** (1972) 1–25.

[40] R. Billen and N. Van de Weghe, *Qualitative spatial reasoning*, *International Encyclopaedia of Human Geography* (2009) 12–18.

[41] C. Freksa, *Conceptual neighborhood and its role in temporal and spatial reasoning*, in *Proc. of the IMACS Workshop on Decision Support Systems and Qualitative Reasoning*, pp. 181–187, 1991.

[42] D. A. Randell, Z. Cui, and A. G. Cohn, *A spatial logic based on regions and connection.*, *KR* **92** (1992) 165–176.

[43] R. Dechter, I. Meiri, and J. Pearl, *Temporal constraint networks*, *Artificial intelligence* **49** (1991), no. 1-3 61–95.

[44] H. Paulheim, *Knowledge graph refinement: A survey of approaches and evaluation methods*, *Semantic web* **8** (2017), no. 3 489–508.

[45] J. van Benthem, *Tense logic and time.*, *Notre Dame Journal of Formal Logic* **25** (1984), no. 1 1–16.

[46] P. B. Ladkin, *Models of axioms for time intervals.*, in *AAAI*, vol. 87, pp. 234–239, 1987.

[47] P. Ladkin and R. Maddux, *The algebra of convex intervals: Short version*, in *Technical Report, KES U-87-2.* Krestel Institute, 1987.

[48] D. Long, *A review of temporal logics*, *The Knowledge Engineering Review* **4** (1989), no. 2 141–162.

[49] G. Ligozat, *Towards a general characterization of conceptual neighborhoods in temporal and spatial reasoning*, in *AAAI 1994 TheTwelfth National Conference on Artificial Intelligence*, 1994.

[50] A. C. Kanmani, T. Chockalingam, and N. Guruprasad, *Rdf data model and its multi reification approaches: A comprehensive comparitive analysis*, in *2016 International Conference on Inventive Computation Technologies (ICICT)*, vol. 1, pp. 1–5, IEEE, 2016.

[51] D. Hernández, A. Hogan, and M. Krötzsch, *Reifying rdf: What works well with wikidata?*, *SSWS@ ISWC* **1457** (2015) 32–47.

[52] B. Adams and K. Janowicz, *Constructing geo-ontologies by reification of observation data*, in *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 309–318, 2011.

[53] M. Giunti, G. Sergioli, G. Vivanet, and S. Pinna, *Representing n-ary relations in the semantic web*, *Logic Journal of the IGPL* **29** (2021), no. 4 697–717.

[54] S. Batsakis and E. G. Petrakis, *Imposing restrictions over temporal properties in owl: A rule-based approach*, in *International Workshop on Rules and Rule Markup Languages for the Semantic Web*, pp. 240–247, Springer, 2012.

[55] V. Nguyen, O. Bodenreider, and A. Sheth, *Don't like rdf reification? making statements about statements using singleton property*, in *Proceedings of the 23rd international conference on World wide web*, pp. 759–770, 2014.

[56] J. J. Carroll, C. Bizer, P. Hayes, and P. Stickler, *Named graphs*, *Journal of Web Semantics* **3** (2005), no. 4 247–267.

[57] E. R. Watkins and D. A. Nicole, *Named graphs as a mechanism for reasoning about provenance*, in *Asia-Pacific Web Conference*, pp. 943–948, Springer, 2006.

[58] O. Hartig, *Rdf* and sparql*: An alternative approach to annotate statements in rdf.*, in *ISWC (Posters, Demos & Industry Tracks)*, 2017.

[59] O. Hartig, *Foundations of rdf* and sparql**, in *Alberto Mendelzon International Workshop on Foundations of Data Management and the Web, CEUR-WS. org*, 2017.

[60] L. Cai, B. Yan, G. Mai, K. Janowicz, and R. Zhu, *Transgcn: Coupling transformation assumptions with graph convolutional networks for link prediction*, in *Proceedings of the 10th International Conference on Knowledge Capture*, pp. 131–138, 2019.

[61] I. Balažević, C. Allen, and T. M. Hospedales, *Tucker: Tensor factorization for knowledge graph completion*, *arXiv preprint arXiv:1901.09590* (2019).

[62] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, *Translating embeddings for modeling multi-relational data*, *Advances in neural information processing systems* **26** (2013) 2787–2795.

[63] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, *Learning entity and relation embeddings for knowledge graph completion*, in *AAAI*, vol. 29, 2015.

[64] G. Ji, S. He, L. Xu, K. Liu, and J. Zhao, *Knowledge graph embedding via dynamic mapping matrix*, 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. 687–696, 2015.

[65] Z. Sun, Z.-H. Deng, J.-Y. Nie, and J. Tang, *Rotate: Knowledge graph embedding by relational rotation in complex space*, *arXiv preprint arXiv:1902.10197* (2019).

[66] B. Yang, W.-t. Yih, X. He, J. Gao, and L. Deng, *Embedding entities and relations for learning and inference in knowledge bases*, *arXiv preprint arXiv:1412.6575* (2014).

[67] M. Nickel, V. Tresp, and H.-P. Kriegel, *A three-way model for collective learning on multi-relational data*, in *Icml*, 2011.

[68] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, and G. Bouchard, *Complex embeddings for simple link prediction*, ICML, 2016.

[69] 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 2019*, p. 131–138, 2019.

[70] Y. Wu, H. Tan, L. Qin, B. Ran, and Z. Jiang, *A hybrid deep learning based traffic flow prediction method and its understanding*, *Transportation Research Part C: Emerging Technologies* **90** (2018) 166–180.

[71] N. Davis, G. Raina, and K. Jagannathan, *Grids versus graphs: Partitioning space for improved taxi demand-supply forecasts*, *arXiv preprint arXiv:1902.06515* (2019).

[72] B. Yu, H. Yin, and Z. Zhu, *Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting*, *arXiv preprint arXiv:1709.04875* (2017).

[73] Z. Cui, R. Ke, and Y. Wang, *Deep bidirectional and unidirectional lstm recurrent neural network for network-wide traffic speed prediction*, *arXiv preprint arXiv:1801.02143* (2018).

[74] W. Jin, Y. Lin, Z. Wu, and H. Wan, *Spatio-temporal recurrent convolutional networks for citywide short-term crowd flows prediction*, in *Proceedings of the 2nd International Conference on Compute and Data Analysis*, pp. 28–35, ACM, 2018.

[75] H. Yao, X. Tang, H. Wei, G. Zheng, and Z. Li, *Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction*, in *2019 AAAI Conference on Artificial Intelligence (AAAI'19)*, 2019.

[76] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, *Convolutional lstm network: A machine learning approach for precipitation nowcasting*, in *Advances in neural information processing systems*, pp. 802–810, 2015.

[77] Y. Li, R. Yu, C. Shahabi, and Y. Liu, *Diffusion convolutional recurrent neural network: Data-driven traffic forecasting*, .

[78] U. Khandelwal, H. He, P. Qi, and D. Jurafsky, *Sharp nearby, fuzzy far away: How neural language models use context*, in *ACL*, 2018.

[79] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, *Attention based spatial-temporal graph convolutional networks for traffic flow forecasting*, in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 922–929, 2019.

[80] 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.

[81] 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).

[82] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, *Improving language understanding by generative pre-training*, *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf* (2018).

[83] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. V. Le, and R. Salakhutdinov, *Transformer-xl: Attentive language models beyond a fixed-length context*, *arXiv preprint arXiv:1901.02860* (2019).

[84] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, *et. al.*, *Google's neural machine translation system: Bridging the gap between human and machine translation*, *arXiv preprint arXiv:1609.08144* (2016).

[85] I. Sutskever, O. Vinyals, and Q. V. Le, *Sequence to sequence learning with neural networks*, in *Advances in neural information processing systems*, pp. 3104–3112, 2014.

[86] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, and Y. Wang, *Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction*, *Sensors* **17** (2017), no. 4 818.

[87] Z. Cui, K. Henrickson, R. Ke, and Y. Wang, *Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting*, *IEEE Transactions on Intelligent Transportation Systems* (2019).

[88] W. Liu, Y. Zheng, S. Chawla, J. Yuan, and X. Xing, *Discovering spatio-temporal causal interactions in traffic data streams*, in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1010–1018, ACM, 2011.

[89] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, *Traffic flow prediction with big data: a deep learning approach*, *IEEE Transactions on Intelligent Transportation Systems* **16** (2014), no. 2 865–873.

[90] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, and H. Li, *T-gcn: A temporal graph convolutional network for traffic prediction*, *IEEE Transactions on Intelligent Transportation Systems* (2019).

[91] W. Huang, G. Song, H. Hong, and K. Xie, *Deep architecture for traffic flow prediction: deep belief networks with multitask learning*, *IEEE Transactions on Intelligent Transportation Systems* **15** (2014), no. 5 2191–2201.

[92] R. Fu, Z. Zhang, and L. Li, *Using lstm and gru neural network methods for traffic flow prediction*, in *2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pp. 324–328, IEEE, 2016.

[93] J. Zhang, Y. Zheng, D. Qi, R. Li, and X. Yi, *Dnn-based prediction model for spatio-temporal data*, in *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, p. 92, ACM, 2016.

[94] J. Ke, H. Zheng, H. Yang, and X. M. Chen, *Short-term forecasting of passenger demand under on-demand ride services: A spatio-temporal deep learning approach*, *Transportation Research Part C: Emerging Technologies* **85** (2017) 591–608.

[95] Y. Li, R. Yu, C. Shahabi, and Y. Liu, *Diffusion convolutional recurrent neural network: Data-driven traffic forecasting*, *arXiv preprint arXiv:1707.01926* (2017).

[96] X. Cao, Y. Zhong, Y. Zhou, J. Wang, C. Zhu, and W. Zhang, *Interactive temporal recurrent convolution network for traffic prediction in data centers*, *IEEE Access* **6** (2017) 5276–5289.

[97] B. Yu, H. Yin, and Z. Zhu, *Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting*, in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pp. 3634–3640, 2018.

[98] S. Li, X. Jin, Y. Xuan, X. Zhou, W. Chen, Y.-X. Wang, and X. Yan, *Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting*, in *Advances in Neural Information Processing Systems*, pp. 5244–5254, 2019.

[99] B. Lim, S. O. Arik, N. Loeff, and T. Pfister, *Temporal fusion transformers for interpretable multi-horizon time series forecasting*, *arXiv preprint arXiv:1912.09363* (2019).

[100] T. N. Kipf and M. Welling, *Semi-supervised classification with graph convolutional networks*, *arXiv preprint arXiv:1609.02907* (2016).

[101] D. K. Hammond, P. Vandergheynst, and R. Gribonval, *Wavelets on graphs via spectral graph theory*, *Applied and Computational Harmonic Analysis* **30** (2011), no. 2 129–150.

[102] J. Liu and W. Guan, *A summary of traffic flow forecasting methods [j]*, *Journal of Highway and Transportation Research and Development* **3** (2004) 82–85.

[103] M. S. Ahmed and A. R. Cook, *Analysis of freeway traffic time-series data by using Box-Jenkins techniques.* No. 722. 1979.

[104] A. J. Smola and B. Schölkopf, *A tutorial on support vector regression*, *Statistics and computing* **14** (2004), no. 3 199–222.

[105] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, *Scheduled sampling for sequence prediction with recurrent neural networks*, in *Advances in Neural Information Processing Systems*, pp. 1171–1179, 2015.

[106] P. Jain, S. Rathi, Mausam, and S. Chakrabarti, *Temporal Knowledge Base Completion: New Algorithms and Evaluation Protocols*, in *EMNLP*, (Online), pp. 3733–3747, Association for Computational Linguistics, Nov., 2020.

[107] H. Ren, W. Hu, and J. Leskovec, *Query2box: Reasoning over knowledge graphs in vector space using box embeddings*, in *ICML*, 2019.

[108] R. Trivedi, H. Dai, Y. Wang, and L. Song, *Know-evolve: deep temporal reasoning for dynamic knowledge graphs*, in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 3462–3471, 2017.

[109] W. Jin, M. Qu, X. Jin, and X. Ren, *Recurrent event network: Autoregressive structure inference over temporal knowledge graphs*, in *EMNLP*, pp. 6669–6683, 2020.

[110] T. Lacroix, G. Obozinski, and N. Usunier, *Tensor decompositions for temporal knowledge base completion*, in *ICML*, 2019.

[111] D. Bahdanau, K. Cho, and Y. Bengio, *Neural machine translation by jointly learning to align and translate*, *arXiv preprint arXiv:1409.0473* (2014).

[112] M. Zaheer, S. Kottur, S. Ravanbhakhsh, B. Póczos, R. Salakhutdinov, and A. J. Smola, *Deep sets*, in *NIPS*, pp. 3394–3404, 2017.

[113] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, *Generalized intersection over union: A metric and a loss for bounding box regression*, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 658–666, 2019.

[114] S. S. Dasgupta, S. N. Ray, and P. Talukdar, *Hyte: Hyperplane-based temporally aware knowledge graph embedding*, in *EMNLP*, 2018.

[115] A. García-Durán, S. Dumančić, and M. Niepert, *Learning sequence encoders for temporal knowledge graph completion*, in *EMNLP*, (Brussels, Belgium), pp. 4816–4821, Association for Computational Linguistics, Oct.-Nov., 2018.

[116] J. Renz and B. Nebel, *Qualitative spatial reasoning using constraint calculi*, in *Handbook of spatial logics*, pp. 161–215. Springer, 2007.

[117] M. J. Egenhofer and K. K. Al-Taha, *Reasoning about gradual changes of topological relationships*, in *Theories and methods of spatio-temporal reasoning in geographic space*, pp. 196–219. Springer, 1992.

[118] C. Freksa, *Using orientation information for qualitative spatial reasoning*, in *Theories and methods of spatio-temporal reasoning in geographic space*, pp. 162–178. Springer, 1992.

[119] K. Zimmermann, *Enhancing qualitative spatial reasoning—combining orientation and distance*, in *European Conference on Spatial Information Theory*, pp. 69–76, Springer, 1993.

[120] A. U. Frank, *Qualitative spatial reasoning about distances and directions in geographic space*, *Journal of Visual Languages & Computing* **3** (1992), no. 4 343–371.

[121] A. U. Frank, I. Campari, and U. Formentini, *Theories and methods of spatio-temporal reasoning in geographic space.* Springer-Verlag Berlin, 1992.

[122] E. Clementini, P. Di Felice, and D. Hernández, *Qualitative representation of positional information*, *Artificial intelligence* **95** (1997), no. 2 317–356.

[123] M. F. Worboys, *Nearness relations in environmental space*, *International Journal of Geographical Information Science* **15** (2001), no. 7 633–651.

[124] D. Hernandez, E. Clementini, and P. Di Felice, *Qualitative distances*, in *International Conference on Spatial Information Theory*, pp. 45–57, Springer, 1995.

[125] I. Pratt and D. Schoop, *A complete axiom system for polygonal mereotopology of the real plane*, *Journal of Philosophical Logic* **27** (1998), no. 6 621–658.

[126] M. J. Egenhofer and R. D. Franzosa, *Point-set topological spatial relations*, *International Journal of Geographical Information System* **5** (1991), no. 2 161–174.

[127] C. Freksa, *Qualitative spatial reasoning*, in *Cognitive and linguistic aspects of geographic space*, pp. 361–372. Springer, 1991.

[128] C. Freksa, *Temporal reasoning based on semi-intervals*, *Artificial intelligence* **54** (1992), no. 1-2 199–227.

[129] J. Renz and B. Nebel, *On the complexity of qualitative spatial reasoning: A maximal tractable fragment of the region connection calculus*, *Artificial Intelligence* **108** (1999), no. 1-2 69–123.

[130] J. Renz and B. Nebel, *Efficient methods for qualitative spatial reasoning*, *Journal of Artificial Intelligence Research* **15** (2001) 289–318.

[131] A. G. Cohn and S. M. Hazarika, *Qualitative spatial representation and reasoning: An overview*, *Fundamenta informaticae* **46** (2001), no. 1-2 1–29.

[132] 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.

[133] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, *Deep contextualized word representations*, in *Proc. of NAACL*, 2018.

[134] Y. Cui, Z. Chen, S. Wei, S. Wang, T. Liu, and G. Hu, *Attention-over-attention neural networks for reading comprehension*, *arXiv preprint arXiv:1607.04423* (2016).

[135] W. L. Hamilton, P. Bajaj, M. Zitnik, D. Jurafsky, and J. Leskovec, *Embedding logical queries on knowledge graphs*, in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 2030–2041, 2018.

[136] 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 *K-CAP 2019*, pp. 171–178, 2019.

[137] H. Ren and J. Leskovec, *Beta embeddings for multi-hop logical reasoning in knowledge graphs*, *Advances in Neural Information Processing Systems* **33** (2020).

[138] T. Kipf, E. Fetaya, K.-C. Wang, M. Welling, and R. Zemel, *Neural relational inference for interacting systems*, in *International Conference on Machine Learning*, pp. 2688–2697, PMLR, 2018.

[139] F. Sala, C. De Sa, A. Gu, and C. Ré, *Representation tradeoffs for hyperbolic embeddings*, in *International conference on machine learning*, pp. 4460–4469, PMLR, 2018.

[140] R. Sarkar, *Low distortion delaunay embedding of trees in hyperbolic plane*, in *International Symposium on Graph Drawing*, pp. 355–366, Springer, 2011.

[141] M. Nickel and D. Kiela, *Learning continuous hierarchies in the lorentz model of hyperbolic geometry*, in *International Conference on Machine Learning*, pp. 3779–3788, PMLR, 2018.

[142] M. Nickel and D. Kiela, *Poincaré embeddings for learning hierarchical representations*, *Advances in neural information processing systems* **30** (2017) 6338–6347.

[143] G. Ji, K. Liu, S. He, and J. Zhao, *Knowledge graph completion with adaptive sparse transfer matrix*, in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, 2016.

[144] C. Gao, C. Sun, L. Shan, L. Lin, and M. Wang, *Rotate3d: Representing relations as rotations in three-dimensional space for knowledge graph embedding*, in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pp. 385–394, 2020.

[145] Z. Cao, Q. Xu, Z. Yang, X. Cao, and Q. Huang, *Dual quaternion knowledge graph embeddings*, in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, pp. 6894–6902, 2021.

[146] T. Trouillon, C. R. Dance, J. Welbl, S. Riedel, É. Gaussier, and G. Bouchard, *Knowledge graph completion via complex tensor factorization*, *arXiv preprint arXiv:1702.06879* (2017).

[147] T. Lacroix, G. Obozinski, and N. Usunier, *Tensor decompositions for temporal knowledge base completion*, *arXiv preprint arXiv:2004.04926* (2020).

[148] O.-E. Ganea, G. Bécigneul, and T. Hofmann, *Hyperbolic neural networks*, *arXiv preprint arXiv:1805.09112* (2018).

[149] A. Gu, F. Sala, B. Gunel, and C. Ré, *Learning mixed-curvature representations in product spaces*, in *International Conference on Learning Representations*, 2018.

[150] I. Balazevic, C. Allen, and T. Hospedales, *Multi-relational poincaré graph embeddings*, *Advances in Neural Information Processing Systems* **32** (2019) 4463–4473.

[151] P. Kolyvakis, A. Kalousis, and D. Kiritsis, *Hyperbolic knowledge graph embeddings for knowledge base completion*, in *The Semantic Web* (A. Harth, S. Kirrane, A.-C. Ngonga Ngomo, H. Paulheim, A. Rula, A. L. Gentile, P. Haase, and M. Cochez, eds.), (Cham), pp. 199–214, Springer International Publishing, 2020.

[152] I. Chami, A. Wolf, D.-C. Juan, F. Sala, S. Ravi, and C. Ré, *Low-dimensional hyperbolic knowledge graph embeddings*, in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 6901–6914, 2020.

[153] D. Pletinckx, *Quaternion calculus as a basic tool in computer graphics*, *The Visual Computer* **5** (1989), no. 1 2–13.

[154] K. Shoemake, *Animating rotation with quaternion curves*, in *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pp. 245–254, 1985.

[155] W. R. Hamilton, *Elements of quaternions.* Longmans, Green, & Company, 1866.

[156] L. Vicci, *Quaternions and rotations in 3-space: The algebra and its geometric interpretation*, .

[157] C. Bisi and G. Gentili, *Möbius transformations and the poincaré distance in the quaternionic setting*, *Indiana University mathematics journal* (2009) 2729–2764.

[158] M. B. Vilain and H. A. Kautz, *Constraint propagation algorithms for temporal reasoning.*, in *Aaai*, vol. 86, pp. 377–382, 1986.

[159] A. G. Cohn and J. Renz, *Qualitative spatial representation and reasoning*, *Foundations of Artificial Intelligence* **3** (2008) 551–596.

[160] A. Klippel, *Spatial information theory meets spatial thinking: is topology the rosetta stone of spatio-temporal cognition?*, *Annals of AAG* **102** (2012), no. 6 1310–1328.

[161] J. O. Wallgrün, D. Wolter, and K.-F. Richter, *Qualitative matching of spatial information*, in *the 18th SIGSPATIAL*, pp. 300–309, 2010.

[162] M. J. Egenhofer, *The family of conceptual neighborhood graphs for region-region relations*, in *GIScience*, pp. 42–55, Springer, 2010.

[163] L. Brahim, K. Okba, and L. Robert, *Mathematical framework for topological relationships between ribbons and regions*, *Journal of Visual Languages & Computing* (2015).

[164] M. J. Egenhofer, *Deriving the composition of binary topological relations*, *Journal of Visual Languages & Computing* **5** (1994), no. 2 133–149.

[165] M. J. Egenhofer and J. Herring, *Categorizing binary topological relations between regions, lines, and points in geographic databases*, *The* **9** (1990), no. 94-1 76.

[166] J. Renz, D. Mitra, *et. al.*, *Qualitative direction calculi with arbitrary granularity*, .

[167] A. Klippel, J. Yang, J. O. Wallgrün, F. Dylla, and R. Li, *Assessing similarities of qualitative spatio-temporal relations*, in *ICSC*, pp. 242–261, Springer, 2012.

[168] A. Klippel and R. Li, *The endpoint hypothesis: A topological-cognitive assessment of geographic scale movement patterns*, in *COSIT*, pp. 177–194, Springer, 2009.

[169] M. P. Dube and M. J. Egenhofer, *An ordering of convex topological relations*, in *GIScience*, pp. 72–86, Springer, 2012.

[170] C. Schultz, M. Bhatt, J. Suchan, and P. A. Wałęga, *Answer set programming modulo 'space-time'*, in *International Joint Conference on Rules and Reasoning*, pp. 318–326, Springer, 2018.

[171] Q. Wang, Z. Mao, B. Wang, and L. Guo, *Knowledge graph embedding: A survey of approaches and applications*, *IEEE TKDE* **29** (2017), no. 12 2724–2743.

[172] D. M. Mark and M. J. Egenhofer, *Calibrating the meanings of spatial predicates from natural language: Line-region relations*, in *Proceedings, SDH 1994*, vol. 1, pp. 538–553, 1994.

[173] K. G. Smith and M. A. Hitt, *Great minds in management: The process of theory development.* OUP Oxford, 2005.

[174] A. L. George and A. Bennett, *Case studies and theory development in the social sciences.* mit Press, 2005.

[175] A. Bosselut, H. Rashkin, M. Sap, C. Malaviya, A. Celikyilmaz, and Y. Choi, *Comet: Commonsense transformers for automatic knowledge graph construction*, in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4762–4779, 2019.

[176] N. Kertkeidkachorn and R. Ichise, *An automatic knowledge graph creation framework from natural language text*, *IEICE TRANSACTIONS on Information and Systems* **101** (2018), no. 1 90–98.

[177] M. Masoud, B. Pereira, J. McCrae, and P. Buitelaar, *Automatic construction of knowledge graphs from text and structured data: A preliminary literature review*, in *3rd Conference on Language, Data and Knowledge (LDK 2021)*, Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021.

[178] Z. Chen, Y. Wang, B. Zhao, J. Cheng, X. Zhao, and Z. Duan, *Knowledge graph completion: A review*, *Ieee Access* **8** (2020) 192435–192456.

[179] R. Campos, G. Dias, A. M. Jorge, and A. Jatowt, *Survey of temporal information retrieval and related applications*, *ACM Computing Surveys (CSUR)* **47** (2014), no. 2 1–41.

[180] D. Gupta and K. Berberich, *Identifying time intervals for knowledge graph facts*, in *Companion Proceedings of the The Web Conference 2018*, pp. 37–38, 2018.

[181] F. Schilder and C. Habel, *Temporal information extraction for temporal question answering.*, in *New Directions in Question Answering*, pp. 35–44, 2003.

[182] Z. Jia, A. Abujabal, R. Saha Roy, J. Strötgen, and G. Weikum, *Tequila: Temporal question answering over knowledge bases*, in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pp. 1807–1810, 2018.

[183] Z. Jia, S. Pramanik, R. Saha Roy, and G. Weikum, *Complex temporal question answering on knowledge graphs*, in *Proceedings of the 30th ACM international conference on information & knowledge management*, pp. 792–802, 2021.

[184] P. G. Campos, F. Díez, and I. Cantador, *Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols*, *User Modeling and User-Adapted Interaction* **24** (2014), no. 1 67–119.

[185] Z. Gantner, S. Rendle, and L. Schmidt-Thieme, *Factorization models for context-/time-aware movie recommendations*, in *Proceedings of the workshop on context-aware movie recommendation*, pp. 14–19, 2010.

[186] X. Zhu, R. Hao, H. Chi, and X. Du, *Fineroute: Personalized and time-aware route recommendation based on check-ins*, *IEEE Transactions on Vehicular Technology* **66** (2017), no. 11 10461–10469.