UNIVERSITY OF CALIFORNIA

Los Angeles

One Step towards Autonomous AI Agent: Reasoning, Alignment and Planning

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Computer Science

by

Xiusi Chen

2024

ABSTRACT OF THE DISSERTATION

One Step towards Autonomous AI Agent: Reasoning, Alignment and Planning

by

Xiusi Chen
Doctor of Philosophy in Computer Science
University of California, Los Angeles, 2024
Professor Wei Wang, Chair

The recent development of artificial intelligence (AI) has facilitated the prosperity of foundation models, such as large language models (LLMs) and vision models. The foundation models have reshaped the way people interact with tools to improve productivity and creativity by taking over many use cases where people use conventional computer software. Being aware of the promising emerging abilities observed from the foundation models, a more interactive picture has been envisioned where the foundation models drive a group of AI agents that play different roles to fulfill more diverse and complex tasks, further benefiting human society. Like humans, AI agents should be able to reason and plan over complex tasks, and they should also be well aligned with human preferences and values. Advanced foundation models provide a solid foundation for the implementation of AI agents. However, agents based on the current foundation models have intrinsic limitations inherited from existing foundation models. In addition to hallucination, these foundation models can demonstrate biases presented in their training data, resulting in output that can be discriminatory. LLMs can expose sensitive or personal information embedded in their training data, risking user

privacy and security. Finally, due to the generative nature of the prevailing foundation models, it is desirable to incorporate planning modules generatively, so the planning process can be seamlessly accomplished during the generation process. In summary, the gaps between the current state-of-the-art and the goals underscore the need for further efforts to improve the reasoning ability, the alignment of human values and the generative planning ability of the foundation models.

My ultimate research goal is to build AI agents that are reliable, unbiased, and capable of planning so that they can be safely and effectively applied in various domains. To achieve this goal, I have divided my research into the following subtasks:

- **Knowledge-Enhanced Reasoning** that aims to improve the factual accuracy and logical coherence of LLM outputs by integrating external knowledge [CZD23, THC24].
- **Minimally Supervised Data Generation and Selection** that aims to improve the efficiency of fine-tuning or in-context learning by selecting the most informative training data [CJC24a].
- **Automatic Constitution Discovery and Self-alignment** that aims to mitigate the risk of generating incorrect, nonsensical, biased or private information [CWN24].
- **Agents Planning** that aims to enable multi-agent strategic learning by incorporating generative goal-guided planning [CJJ22, CWH24].

In this thesis, I will first emphasize the significance of building such reliable, unbiased, capable-of-planning AI agents, and then introduce four lines of my work, and finally the future challenges and opportunities.

The dissertation of Xiusi Chen is approved.

University of California, Los Angeles

2024

It's not the destination, it's the journey.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

This dissertation marks the end of my Ph.D. journey. As I reflect on this journey, I feel gratified and perfectly content. Not only because of the achievement that comes with the journey, but because the journey itself is already enjoyable enough. I could not have enjoyed this journey alone and I am deeply indebted to the people who shared the ride with me.

First, I express my ultimate gratitude to my academic advisor, Prof. Wei Wang. I still remember the exact moment we first met in person in the FIT building of Tsinghua University. At that time, I did not realize that moment was so decisive that it shaped my academic career. Throughout the years, Prof. Wang has been encouraging me to explore only the most meaningful and valuable directions to work on, which ultimately shaped my research taste and my mindset. Meanwhile, she taught me to never settle for the ordinary. Despite being such an established professor, Prof. Wang still stays up late in the night to go over my paper submission word by word and makes concrete edits or comments. Prof. Wang's dedication shows me what it means to work extremely hard for something that I am passionate about. I am fortunate to have had many deep conversations with Prof. Wang, from which I learned so much from her. Prof. Wang is the best advisor I could have ever asked for. It is one of my proudest achievements to be able to earn a Ph.D. under the guidance of Prof. Wang. Over the years, Prof. Wang has turned from a role model to a family for me.

I extend my sincere gratitude to the members of my thesis committee, Prof. Yizhou Sun, Prof. P. Jeffrey Brantingham, and Prof. Kai-Wei Chang. Prof. Sun touches me with her kindness and integrity. Prof. Brantingham always shares insightful viewpoints and pushes me to think beyond the methodology. I am deeply grateful to Prof. Brantingham for his unconditional support throughout the years. Prof. Chang's sharpness pushes me to always think about the weaknesses of my research. I could not finish this thesis without the support of these committee members.

I am greatly grateful to Prof. Hong Mei and Prof. Jun Gao from Peking University for their guidance in preparing me for my Ph.D. study and their support in offering me the opportunity of visiting the University of Illinois Urbana-Champaign. It is due to my experience at UIUC that I decided to pursue an academic career. I am deeply grateful to Prof. ChengXiang Zhai for his support during my time at UIUC. I could not have secured a Ph.D. offer if it were not for these mentors. Even during my Ph.D. years, Prof. Mei, Prof. Gao and Prof. Zhai often ask me about my progress and updates, and keep providing me with continual guidance and encouragement.

I am grateful to have met groups of brilliant mentors and collaborators from my multiple internships. I have learned so much from them, especially the mentality of always being open-minded and always sharing with people new ideas. I would like to thank Wenchao Yu and Wei Cheng from NEC Laboratories America, Jyun-Yu Jiang, Wei-Cheng Chang, Prof. Cho-Jui Hsieh, and Hsiang-Fu Yu from the PECOS team at Amazon Search, and Hongzhi Wen, Yu Wang, Bowen Jin, Chen Luo, Qingyu Yin, Ruirui Li, and Sreyashi Nag from the Query Understanding team at Amazon Search.

I am fortunate to be accompanied by a group of outstanding labmates from the UCLA ScAi lab, and they are Wei Cheng, Ruirui Li, Chelsea Jui-Ting Ju, Wenchao Yu, Cheng Zheng, Jyun-Yu Jiang, Zeyu Li, Guangyu Zhou, Yichao Zhou, Seungbae Kim, Yunsheng Bai, Junheng Hao, Ziniu Hu, Kewei Cheng, Song Jiang, Yewen Wang, Zhiping Xiao, Zijie Huang, Shichang Zhang, Roshni Iyer, Zongyue Qin, Derek Xu, Alexander Taylor, Mingyu Ma, Yanqiao Zhu, Anthony Cuturrufo, and Jeehyun Hwang. Special thanks to Jyun-Yu and Ziniu for their discussions on multiple collaborations. I would like to thank my external collaborators and friends: Chao Zhang, Yu Zhang, Jinliang Deng, Meng Qu, Qi Zhu, Jingbo Shang, Zhankui He, Jiacheng Li, Yupeng Hou, An Yan, Yijun Tian, Andrew Bai, Kun Jin, and Zekai Chen. In particular, I would like to thank Chao Zhang, Yu Zhang and Jinliang Deng, who are great friends, collaborators, and families. Chao, Yu and Jinliang share my joys and tears throughout my Ph.D. journey, and they always cheer me up and calm me down whenever needed, and they are always by my side whenever I need advice on crucial life-changing decisions. This thesis could not be finished without them and I look forward to our future collaborations. I also would like to give a special shoutout to Andrew. The conversations with Andrew about life and work and the times we spent together will be in my memory for a long time.

My ultimate gratitude to my father Hongping Chen, and mother Chongfei Luo, for their endless love and constant support in all aspects of my life. My father always challenges me and pushes me to reach beyond my limits and achieve higher goals. My mother shows what it means to be mentally strong and relentless. This thesis is dedicated to my father and my mother.

Lastly, my deepest gratitude to my fiancé Ruihan Wu. Ruihan is my dearest sunshine that lights up my world. Without Ruihan I would not have made it this far and been so happy every day. I am so lucky to have you around and I thank you for your love, encouragement, and support in this wonderful journey.

VITA

2011–2015	B.S. (Computer Science), Peking University.
2015–2018	M.S. (Computer Science), Peking University.
2019–2020	Teaching Assistant and Asscociate, Computer Science Department, University of California, Los Angeles.
2018–2024	Graduate Student Researcher, Computer Science Department, University of California, Los Angeles.
2019	Research Intern, NEC Laboratories America.
2022	Applied Scientist Intern, Amazon Research.
2023	Applied Scientist Intern, Amazon Research.

PUBLICATIONS

[CJW22] **Xiusi Chen**, Jyun-Yu Jiang, and Wei Wang. "Scalable Graph Representation Learning via Locality-Sensitive Hashing." In Proceedings of the 31st ACM International Conference on Information & Knowledge Management. (**CIKM 2022**)

[CJJ22] **Xiusi Chen**, Jyun-Yu Jiang, Kun Jin, Yichao Zhou, Mingyan Liu, P. Jeffrey Brantingham, and Wei Wang. "Reliable: Offline reinforcement learning for tactical strategies in professional basketball games." In Proceedings of the 31st ACM International Conference on Information & Knowledge Management. (**CIKM 2022**)

[CZD23] **Xiusi Chen**, Yu Zhang, Jinliang Deng, Jyun-Yu Jiang, and Wei Wang. "Gotta: generative few-shot question answering by prompt-based cloze data augmentation." In Proceedings of the 2023 SIAM International Conference on Data Mining. (**SDM 2023**)

[CWN24] **Xiusi Chen**, Hongzhi Wen, Sreyashi Nag, Chen Luo, Qingyu Yin, Ruirui Li, Zheng Li, and Wei Wang. "IterAlign: Iterative constitutional alignment of large language models." In Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. (**NAACL 2024**)

[CJC24b] **Xiusi Chen**, Jyun-Yu Jiang, Wei-Cheng Chang, Cho-Jui Hsieh, Hsiang-Fu Yu, and Wei Wang. "MinPrompt: Graph-based Minimal Prompt Data Augmentation for Few-shot Question Answering." In Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics. (**ACL 2024**)

[CWH24] **Xiusi Chen**, Wei-Yao Wang, Ziniu Hu, David Reynoso, Kun Jin, Mingyan Liu, P Jeffrey Brantingham, and Wei Wang. "PlayBest: Professional Basketball Player Behavior Synthesis via Planning with Diffusion." In Proceedings of the 33rd ACM International Conference on Information & Knowledge Management. (**CIKM 2024**)

[ZCJ24] Yu Zhang*, **Xiusi Chen***, Bowen Jin*, Sheng Wang, Shuiwang Ji, Wei Wang, and Jiawei Han. "A Comprehensive Survey of Scientific Large Language Models and Their Applications in Scientific Discovery." Under review, on ArXiv.

[THC24] Yijun Tian*, Yikun Han*, **Xiusi Chen***, Wei Wang, Nitesh V Chawla. "TinyLLM: Learning a Small Student from Multiple Large Language Models." Under review, on ArXiv.

CHAPTER 1

Introduction

1.1 Motivation

Building fully autonomous agents through artificial intelligence (AI) has been a long-term goal for the field of computer science. Recent developments in generative foundation models have provided a solid basis for implementation. In particular, large language models (LLMs) are transforming our world with their incredible versatility and power. The BERT model [DCL18] marks a milestone that starts the era of pre-trained language models. Before that time, different kinds of specialized model have been developed to accomplish very specific tasks. However, BERT introduced a new paradigm by enabling models to be pre-trained on vast amounts of text data and then fine-tuned for various downstream tasks. This approach significantly improved performance across a wide range of applications, from natural language understanding and translation to sentiment analysis and question-answering systems. The success of BERT paved the way for more advanced models such as GPT-3 [Ope23b], PaLM [CND23], and GPT-4 [Ope23a], which further expanded the capabilities and applications of LLMs. Today, LLMs are integral to many sectors, including healthcare, research, finance, education, etc. Likewise, recent developments in vision models have seen a significant advancement with the introduction of diffusion models, which have revolutionized the field of image generation and understanding. A notable example of this innovation is the OpenAI Sora [BPH24], which leverages diffusion processes to achieve state-of-the-art performance in computer vision tasks. The Sora model stands out due to its ability to generate high-quality videos from text descriptions, still images, or videos, bridging the gap between natural language and visual content with unprecedented accuracy. Most importantly, it is found that video models exhibit several interesting emergent capabilities when trained at scale. These capabilities enable Sora to simulate some aspects of people, animals and environments from the physical world. These properties emerge without any explicit inductive biases, but they are purely phenomena of scale. This breakthrough underscores the potential of diffusion-based vision models to transform various applications, from creative industries to scientific research.

These developments in foundation models have changed the way people interact with technology. Previously people usually interacted with computer operating systems or software, such as Windows, Linux, Excel, or PowerPoints, but now we see more and more interactions directly with foundation-model-based applications. Now, based on the promising emerging abilities observed from the foundation models, people are starting to envision an even more interactive picture where the foundation models drive a bunch of AI agents that play different roles and benefit human society by enhancing creativity and content generation. For example, in healthcare, AI agents improve medical image analysis and assist in drug discovery by generating potential molecular structures, and in customer service, intelligent chatbots handle inquiries and provide constant support. AI agents can also aid scientific research by analyzing large datasets to uncover new insights and promote environmental sustainability by optimizing resource management. These developments lead to increased efficiency, better decision-making, and enriched human experiences in various domains.

To build such AI agents, we envision different properties and components. First, AI agents must exhibit strong reasoning abilities that enable them to analyze complex problems, draw logical conclusions, and make informed decisions based on a comprehensive understanding of the diverse data. For example, in medical diagnostics, an AI agent with strong reasoning can accurately interpret patient data and suggest potential diagnoses or treatment plans. Equally important is their alignment with human preferences and values, ensuring that their actions and decisions are ethical, transparent, and beneficial to society. An example of this is in autonomous vehicles, where AI must prioritize passenger safety and adhere to traffic laws, reflecting societal norms and ethical standards. In addition, robust planning skills are crucial, allowing AI agents to anticipate future scenarios, strategize effectively, and adapt to changing conditions. In logistics, an AI with strong planning capabilities can optimize delivery routes, manage inventory, and predict supply chain disruptions, thus enhancing efficiency and reliability to achieve long-term goals. These core properties collectively ensure that AI agents are not only intelligent, but also trustworthy and effective in practical applications.

1.2 My Research Overview

Being aware of these desired properties of AI agents, my research focuses on achieving the above goals through four modules: **Knowledge-Enhanced Reasoning**, **Minimally-supervised Data Generation and Selection**, **Automatic Constitution Discovery and Self-alignment**, and **Agents Planning**.

1. The Knowledge-Enhanced Reasoning module aims to improve the **factual accuracy** and **logical coherence** of LLM outputs by integrating structured knowledge bases. This approach helps mitigate the risk of generating incorrect or non-sensical information.

    - Chapter 2 presents *Generative Few-shot Question Answering by Prompt-based Cloze Data Augmentation* (GOTTA) that augmented prompt data for reasoning;
    - Chapter 3 presents *Learning a Small Student from Multiple Large Language Models* (TINYLLM) that improves the generalization abilities of the student model by learning from rationales of multiple teacher models.

2. Building on the first module, the Minimally Supervised Data Generation and Selection module aims to make the Knowledge-Enhanced Reasoning process more **efficient**. It generates and selects high-quality training data with minimal human supervision,

thereby streamlining the enhancement process and reducing the manual effort required.

- Chapter 4 presents MinPrompt*: Graph-based Minimal Prompt Data Augmentation for Few-shot Question Answering* (MinPrompt) that selects the most informative factual sentences, and generates the data samples for fine-tune or in-context learning.

3. The Automatic Constitution Discovery and Self-alignment module focuses on enhancing the model's ability to **autonomously discover and align with ethical and privacy standards**. By incorporating these modules, my research strives to create more **reliable, unbiased, and ethically sound** language models that can be safely and effectively applied across various domains.

   - Chapter 5 presents *Iterative Constitutional Alignment of Large Language Models* (IterAlign) that automatically generates constitutions out of red teaming user data, and uses the data-driven constitutions to guide natural-language-based model alignment.

4. Lastly, the Agents Planning module extends the application of LLMs to autonomous AI agents, utilizing advanced planning algorithms to enable sophisticated decision-making and strategic behaviors in various contexts, such as tactical strategies in professional basketball games and behavior synthesis.

   - Chapter 6 presents *Offline Reinforcement Learning for Tactical Strategies in Professional Basketball Games* (ReLiable) that uses offline reinforcement learning (RL) techniques to learn optimal discrete actions conditioned on a certain given on-court state.

   - Chapter 7 presents *Professional Basketball Player Behavior Synthesis via Planning with Diffusion* (PlayBest) that formulates the planning task as a trajectory generation one, then uses diffusion model and classifier-guided conditional sampling to generate trajectories that are of high-rewards representing promising tactics.

Putting these pieces together, I am pursuing the ultimate vision to build autonomous large multi-modal model agents that have the capacity of reasoning and planning, towards achieving true humanlike intelligence aligned with human values.

1.3 My Research Contributions

My vision is supported by my previous research, which has led to more than 30 research papers published in top Data Mining venues (CIKM, KDD, WWW, WSDM), Nature Language Processing venues (ACL, NAACL), and Machine Learning venues (ICML, AAAI). Notably, I received the Best Poster Award (Honorable Mention) at SDM 2023. Many models I designed have been integrated into industrial products, including Amazon Rufus and Alibaba Qwen. The dataset and benchmark I developed have been downloaded more than 300K times, the Github repository summarizing scientific language models has received over 400 stars. My research served as core building blocks for many NSF, MURI research grants, as well as corporate grants.

Part I

Knowledge-Enhanced Reasoning

CHAPTER 2

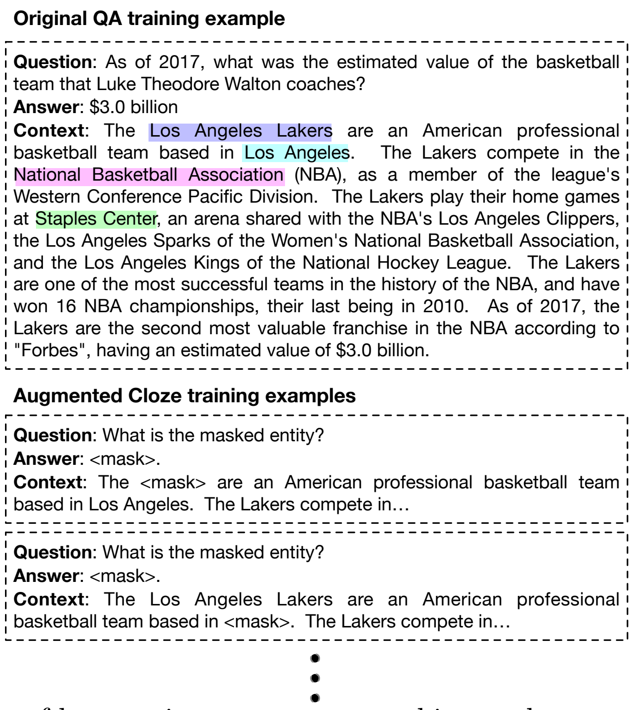
Gotta: Generative Few-shot Question Answering by Prompt-based Cloze Data Augmentation

2.1 Abstract

Few-shot question answering (QA) aims at precisely discovering answers to a set of questions from context passages while only a few training samples are available. Although existing studies have made some progress and can usually achieve proper results, they suffer from understanding deep semantics for reasoning out the questions. In this paper, we develop GOTTA, a **G**enerative pr**O**mp**T**-based da**T**a **A**ugmentation framework to mitigate the challenge above. Inspired by the human reasoning process, we propose to integrate the cloze task to enhance few-shot QA learning. Following the recent success of prompt-tuning, we present the cloze task in the same format as the main QA task, allowing the model to learn both tasks seamlessly together to fully take advantage of the power of prompt-tuning. Extensive experiments on widely used benchmarks demonstrate that GOTTA consistently outperforms competitive baselines, validating the effectiveness of our proposed prompt-tuning-based cloze task, which not only fine-tunes language models but also learns to guide reasoning in QA tasks. Further analysis shows that the prompt-based loss incorporates the auxiliary task better than the multi-task loss, highlighting the strength of prompt-tuning on the few-shot QA task.

2.2 Introduction

Question answering (QA) is the task of precisely discovering answers to natural language questions given the narrative contexts. With a wide range of downstream applications, such as knowledge graph completion [LDX22], response recommendation [ZBL13], review opinion mining [ZGS19], and product attribute extraction [WYK20], it has drawn a lot of attention in the text mining community and has risen to one of the holy-grail tasks. Following the line of supervised learning, one can successfully build QA methods that achieve decent results. However, the assumption that a large amount of annotated QA training examples quickly poses limitations since annotation requiring efforts from domain experts is extremely expensive.



Original QA training example

Question: As of 2017, what was the estimated value of the basketball team that Luke Theodore Walton coaches?
Answer: $3.0 billion
Context: The Los Angeles Lakers are an American professional basketball team based in Los Angeles. The Lakers compete in the National Basketball Association (NBA), as a member of the league's Western Conference Pacific Division. The Lakers play their home games at Staples Center, an arena shared with the NBA's Los Angeles Clippers, the Los Angeles Sparks of the Women's National Basketball Association, and the Los Angeles Kings of the National Hockey League. The Lakers are one of the most successful teams in the history of the NBA, and have won 16 NBA championships, their last being in 2010. As of 2017, the Lakers are the second most valuable franchise in the NBA according to "Forbes", having an estimated value of $3.0 billion.

Augmented Cloze training examples

Question: What is the masked entity?
Answer: <mask>.
Context: The <mask> are an American professional basketball team based in Los Angeles. The Lakers compete in...

Question: What is the masked entity?
Answer: <mask>.
Context: The Los Angeles Lakers are an American professional basketball team based in <mask>. The Lakers compete in...

Figure 2.1: An example of how entity-aware text masking and prompt-style data augmentation work. GOTTA selects entities that are covered by knowledge bases, and creates prompt-style augmented data for training purpose.

We investigate the few-shot QA task, which aims to solve the QA task while only a few training examples are present. Under the few-shot setting, most existing approaches either propose a new task and pre-train a large language model from scratch [RKB21], or fine-tune the pre-trained model on the training examples [CN21]. These practices do not explicitly understand the entities in the input text (i.e., the context text and the question text) before generating the output (i.e., the answer text), which contradicts the conventional human thinking process. For example, in reading comprehension exams, people have to fully understand and digest the context semantics before getting precise answers. In other words, directly mapping from the text and the question to the answer lacks a deep understanding of the context.

To bridge this gap, we develop GOTTA, a **G**enerative pr**O**mp**T**-based da**T**a **A**ugmentation framework for few-shot QA. In GOTTA, we design a knowledge-based cloze task to serve as a companion to enhance the main QA task. To make the cloze task more dedicated for QA, we utilize publicly available knowledge bases and focus on the covered entities by only selecting the entities in the text as the object to construct cloze problems. By constructing more data for fine-tuning, we incorporate the external knowledge in the knowledge bases in the hope of introducing more inductive bias that is beneficial to the QA task. The inductive bias provides extra supervision beyond the weak supervision signals only provided in the few-shot QA training set. Intuitively, the cloze task is to imitate the human behavior of understanding the context by filling in the blanks. We conduct this entity-aware cloze because identifying the entities and understanding their relations is crucial for solving QA problems on the same chunk of text.

Inspired by recent advantages of prompt-tuning, as shown in Figure 2.1 and Figure 2.2, we feature both QA and cloze tasks in the same prompt template to align with each other at the pre-training stage. Following this routine, no redundant model parameters are introduced while the pre-trained model can maximize the performance on our downstream QA task, especially under the few-shot setting. Although our cloze task is quite similar to the popular masked language modeling (MLM), there are two major distinctions between entity-aware text masking and MLM. First, MLM randomly masks word tokens while entity-aware text masking only targets entities that are more likely to be relevant to the QA task. Second, MLM is usually pipelined with a softmax function to select one token while entity-aware text masking generates a token sequence to form a text span, which is more favored by QA tasks. Extensive experiments on publicly available and conventional benchmarks demonstrate that Gotta is able to achieve generally better results over competitive baselines, validating the effectiveness of the cloze task. Further in-depth analysis shows that the prompt-based loss incorporates the auxiliary task better than classification loss, highlighting the effectiveness of prompt-tuning on the few-shot QA task.

We summarize our contributions as follows:

- We propose to incorporate the cloze task as a data augmentation module to extract self-supervised training examples to enhance the learning for few-shot QA.
- We formulate both QA and cloze tasks in the same format, allowing us to apply prompt-tuning to take full advantage of pre-trained large language models.
- We conduct extensive experiments on publicly accessible benchmarks to validate the effectiveness of Gotta, and observe consistent improvement over competitive compared methods. Beyond that, we also study the necessity of different parts of the model, providing the readers with a better understanding of the framework.[1]

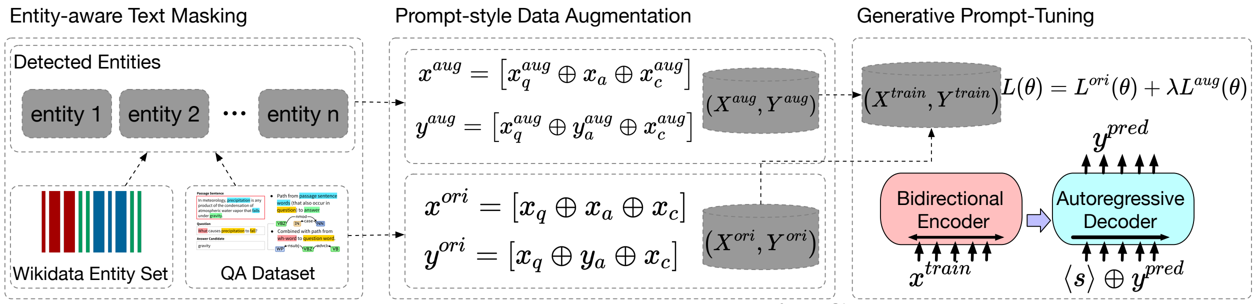
2.3 Related Work

Existing studies most related to our work come from three aspects: few-shot QA, prompt-tuning, and data augmentation. In this section, we briefly recap and distinguish our proposed method from theirs.

Few-Shot QA. Prior studies in QA either reuse the high-performing pre-trained lan-

[1]The code for Gotta is at `https://github.com/xiusic/Gotta`.

guage models (PLMs) [LCG20, JCL20], or train a model from scratch on synthetic QA data [PSS20, LDR19, AAP19]. However, all of them require fine-tuning the models on massive annotated data from the downstream QA task, which is often impractical in real-world cases. Several approaches have recently been developed to allow the model to quickly adapt to the downstream task with solely a handful of annotated data [RKB21, CN21]. Ram et al. [RKB21] tailor the pre-training scheme specialized for handling QA tasks. They design a recurring span selection objective for pre-training, which aligns with the common objective in extractive QA tasks. To save the effort to pre-train the model on a large-scale corpus, Chada and Natarajan [CN21] seek to explore the capacity of the existing PLMs. They propose a simple framework, known as FewshotQA, where a QA-style prompt is constructed to cast the QA problem as a text generation problem. Specifically, the prompt is created as a concatenation of the question and a mask token representing the answer span. In this way, the input format is geared toward processing by the PLMs. Distinct from these two studies, we focus on exploring more relevant information in the context data, aside from the annotated QA pairs, to fine-tune the model under the few-shot setting. KECP [WWQ22] is a concurrent work with GOTTA that focuses only on extractive QA (EQA). Also inspired by prompt-tuning, KECP views the EQA task as a non-autoregressive MLM generation problem and uses a span-level contrastive learning objective to improve the final performance.


Entity-aware Text Masking
Detected Entities
entity 1
entity 2
...
entity n
Wikidata Entity Set
QA Dataset
Prompt-style Data Augmentation
$x^{aug} = [x_q^{aug} \oplus x_a \oplus x_c^{aug}]$
$y^{aug} = [x_q^{aug} \oplus y_a^{aug} \oplus x_c^{aug}]$
(X^{aug}, Y^{aug})
$x^{ori} = [x_q \oplus x_a \oplus x_c]$
$y^{ori} = [x_q \oplus y_a \oplus x_c]$
(X^{ori}, Y^{ori})
Generative Prompt-Tuning
(X^{train}, Y^{train})
$L(\theta) = L^{ori}(\theta) + \lambda L^{aug}(\theta)$
y^{pred}
Bidirectional Encoder
Autoregressive Decoder
x^{train}
$\langle s \rangle \oplus y^{pred}$
Figure 2.2: Framework overview for GOTTA.

Prompt-Tuning. Standard fine-tuning of PLMs for few-shot learning does not achieve satisfying performance in many cases because the limited training samples may not be sufficient for optimizing the parameters in the newly introduced task head. To reuse the language modeling capability of PLMs without introducing randomly initialized parameters, prompt-based approaches [GFC21, HDW22, LBW22, MLH22, SS21a, SS21c, TMB21] formulate training samples as natural language prompt templates so that the downstream tasks can be solved as a masked token prediction problem. Further studies propose to replace the manual design of prompts with automatic search or learning [CHD22, HKM21, LAC21b, LZD21, ZLC21, ZFC21]. Although prompt-tuning has demonstrated remarkable few-shot learning performance in some tasks (e.g., text classification and natural language inference [WSM19]), it has not been extensively explored in question answering. In this paper, we explore a prompt-based data augmentation framework for few-shot QA.

Data Augmentation. Under the few-shot setting, data augmentation mainly aims to create more training data based on a small number of provided training samples to overcome label scarcity when training the model. Pioneering studies on text data augmentation include EDA [WZ19] and UDA [XDH20], which leverage text editing (e.g., synonym replacement, random swap) and back translation to create more labeled data for text classification. In another line of work, several studies generate training data by fine-tuning autoregressive PLMs on the training set [ACG20, YMF20] or using label-specific prompts [SS21b] to guide text generation toward the desired label. However, most of the studies mentioned above focus on the task of few-shot text classification. In contrast, our Gotta framework proposes a cloze-style data augmentation method for few-shot QA.

2.4 Gotta: The Proposed Framework

The overall framework of Gotta is illustrated in Figure 2.2. The core idea is to augment training data with cloze-style questions to force the model to understand the contexts beyond the original questions. We fulfill this idea in three steps: First, we identify the tokens that should be masked in the cloze task. Intuitively, such tokens need to indicate the answers to the original questions. Then, we construct the prompt data by combining the masked tokens and our designed template. Finally, we feed the original QA training samples and the created prompt data into a pre-trained BART model [LLG20] for fine-tuning.

2.4.1 Entity-aware Text Masking

The first step to fulfill the cloze task is entity-aware text masking. The cloze task is often referred to as "masked language modeling" (MLM) in the literature [DCL19]. Although MLM is widely used as a pre-training task in NLP, it is still less explored how to pick the masked token spans to achieve good performance in a specific downstream task. Indeed, prevailing PLMs like BERT [DCL19] randomly mask a proportion of tokens in each sequence. However, even though PLMs with randomly masking statistically can survive with a large-scale training corpus and the law of large numbers, few-shot QA tasks with only tens of short sequences could potentially receive only weak and noisy samples. For the sake of our QA task, we propose to enable the model to infer the crucial parts of the reasoning procedure. Human reasoning is usually considered as hops between entities [CZW21]. A robust model should be able to recover important masked entities based on their context. To achieve this idea, we take the entity set of the Wikidata knowledge graph [VK14] as the entity corpus. For each training sample, we extract all the text spans recognized as Wikidata entities. As a result, the created cloze questions will be centered around meaningful entities rather than irrelevant tokens to the QA task, such as articles, pronouns, and stop-words.

2.4.2 Prompt-style Data Augmentation

Based on the output of entity-aware text masking, we pursue the recent success of prompt-tuning to produce augmented data for the prompt-based GOTTA model. Specifically, we formally formulate the following template to integrate QA and cloze tasks, thereby generating few-shot QA input data x^{ori} as:

$$\begin{aligned} x_q &= \textit{Question}: \mathbf{q} \\ x_a &= \textit{Answer}: \langle \mathrm{m}\textit{ask} \rangle \\ x_c &= \textit{Context}: \mathbf{c} \\ x^{ori} &= [x_q \oplus x_a \oplus x_c] \end{aligned}$$

The labels y are formulated as follows:

$$\begin{aligned} y_a &= \textit{Answer}: \mathbf{a}, \\ y &= [x_q \oplus y_a \oplus x_c], \end{aligned}$$

where $\mathbf{q}$, $\mathbf{a}$ and $\mathbf{c}$ are texts of the question, answer text, and context, respectively; $\oplus$ denotes string concatenation.

For the augmented data, we fix the question text $\mathbf{q}^{\mathrm{aug}}$ as follows:

$$\mathbf{q}^{\mathrm{aug}} = \textit{What is the masked entity?}$$

Note that in the augmented cloze data samples, we also mask the selected entity in x_c to form the context text for the augmented data x_c^{aug} in addition to the mask token in x_a. Figure 2.1 illustrates the details of an augmented data sample (x^{aug}, y^{aug}). Let (X^{ori}, Y^{ori}) and (X^{aug}, Y^{aug}) denote all the training samples of QA and cloze, respectively. Our complete training set (X^{train}, Y^{train}) is the union of (X^{ori}, Y^{ori}) and (X^{aug}, Y^{aug}).

2.4.3 Generative Prompt-Tuning

One of the most apparent advantages of aligning augmented and original data is the model's capability of seamlessly digesting both without a distinct loss. In a nutshell, Gotta adopts an encoder-decoder model as:

$$y^{pred} = decoder_{\theta_D}(encoder_{\theta_E}(x)), \tag{2.1}$$

where θ_E and θ_D are learnable parameters; $x \in X^{train}$ can be either an original training sample or an augmented one.

Our training objective maximizes the log-likelihood of the text in the reference answer $y \in Y^{train}$. The loss functions with respect to the original samples and the augmented samples can be expressed as follows:

$$L^{ori}(\theta) = \sum_{(x,y)\in(X^{ori},Y^{ori})} \log\left(\prod_{i=1}^{n} P\left(y_i \mid y_{<i}, x; \theta\right)\right) \tag{2.2}$$

and

$$L^{aug}(\theta) = \sum_{(x,y)\in(X^{aug},Y^{aug})} \log\left(\prod_{i=1}^{n} P\left(y_i \mid y_{<i}, x; \theta\right)\right), \tag{2.3}$$

where $\theta = \{\theta_D, \theta_E\}$.

The overall loss function takes a weighted sum:

$$L(\theta) = L^{ori}(\theta) + \lambda L^{aug}(\theta). \tag{2.4}$$

Here, $\lambda > 0$ is a hyperparameter that balances between the QA task and the prompted cloze task.

2.5 Experiments

In this section, we describe in detail how we set up our experiments, then we report the experimental results and discuss the results. We further provide some in-depth analysis of Gotta, through which we can better understand the model.

2.5.1 Experimental Setup

Datasets. Following Splinter [RKB21] and FewshotQA [CN21], we sample subsets from the MRQA 2019 shared task [FTJ19] for our few-shot experiments. Specifically, MRQA contains eight widely used benchmark question answering datasets: SQuAD [RZL16], NewsQA [TWY17], TriviaQA [JCW17], SearchQA [DSH17], HotpotQA [YQZ18], Natural Questions [KPR19], BioASQ [TBM15b], and TextbookQA [KSS17]. Following Splinter [RKB21], smaller training datasets are sampled in a logarithmic manner from the original full datasets, resulting in few-shot datasets with training example numbers 16, 32, 64, and 128.

Comparative Baselines. We evaluate the performance of GOTTA against four competitive few-shot QA methods, including **RoBERTa** [LOG19], **SpanBERT** [JCL20], **Splinter** [RKB21], and **FewshotQA** [CN21].

Implementation Details. We extract $24,863,792$ entities from Wikidata for entity candidate matching. When extracting the entities in the contexts of training samples, we use the Aho-Corasick algorithm[2] [AC75] to conduct exact multi-pattern lexical matching. For all the models, we use the same hyperparameters during training for a fair comparison. Specifically, the models are optimized by Adam [KB14] with bias corrections. The learning rate is 2×10^{-5} without learning rate scheduling. The training batch size is set to 2. The maximum sequence length of sequence generation is 100 for FewshotQA and GOTTA. We train all compared models for 25 epochs. The reported results are given by the best-performing checkpoint on the development sets. For GOTTA, we perform a grid search for the loss weight λ in the space $\{0.01, 0.05, 0.1, 0.5, 1.0, 10.0\}$. All the experiments are run on NVIDIA Tesla A100-SXM4 Tensor Core GPUs with 40GB memory.

Evaluation Metrics. Following previous studies [RKB21, CN21], we use the F1 score as our evaluation metric. Specifically, for each sample in the test set, the predicted span and

[2]https://github.com/WojciechMula/pyahocorasick/

# examples	SQuAD	TriviaQA	NQ	NewsQA	SearchQA	HotpotQA	BioASQ	TextbookQA
16	336	2,118	883	1,904	2,620	517	591	1,814
32	711	4,287	1,422	2,801	5,452	1,005	1,205	3,934
64	1,539	8,592	2,696	5,867	10,601	2,090	2,568	7,526
128	3,052	17,301	4,989	11,469	21,113	4,128	5,226	15,504

Table 2.1: Number of augmented training examples per dataset. We construct one training example for each entity extracted from the passages and form the cloze task.

the ground truth answer are treated as bags of words, and F1 scores are applied to compute the overlap between these two sets. If there are multiple ground-truth answers to a particular question, we take the maximum of the corresponding F1 scores.

2.5.2 Performance Comparison

Table 4.2 shows the few-shot QA performance of compared models across all the benchmarks when 16, 32, 64, and 128 training examples are given. For both **FewshotQA** and **Gotta**, we use BART-large as the backbone PLM. We also report their performance when BART-base is applied as the PLM, in which case the models are denoted as **FewshotQA-base** and **Gotta-base**, respectively. We repeat the same experiment 5 times using different random seeds and report the mean and standard deviation of the results for each method. Furthermore, we include the relative performance gain of Gotta over the second-best method, i.e., FewshotQA. Overall, Gotta outperforms all the compared methods by a decent margin in most cases. Even beyond that, we observe a lower variance in results produced by Gotta over FewshotQA in most cases (24 out of 32), especially when fewer training examples are available (14 out of 16).

Next, let us take a closer look at specific datasets. On SQuAD and HotpotQA, Gotta consistently achieves higher F1 with lower variance. On TriviaQA, NewsQA, SearchQA, and TextbookQA, we observe relatively more significant performance gains over the best baseline. We conjecture that it is because the number of augmented data samples on these datasets is larger than that on other datasets. Therefore, signals from the cloze task are sufficient to impact the main QA task positively.

Model	SQuAD	TriviaQA	NQ	NewsQA	SearchQA	HotpotQA	BioASQ	TextbookQA
16 Examples								
RoBERTa	7.7±4.3	7.5±4.4	17.3±3.3	1.4±0.8	6.9±2.7	10.5±2.5	16.7±7.1	3.3±2.1
SpanBERT	18.2±6.7	11.6±2.1	19.6±3.0	7.6±4.1	13.3±6.0	12.5±5.5	15.9±4.4	7.5±2.9
Splinter	54.6±6.4	18.9±4.1	27.4±4.6	20.8±2.7	26.3±3.9	24.0±5.0	28.2±4.9	19.4±4.6
FewshotQA-base	55.3±2.7	39.6±6.2	46.9±1.4	36.5±2.6	40.8±4.4	43.7±2.4	52.1±1.6	16.7±2.2
FewshotQA	72.5±3.7	47.1±7.6	57.3±3.2	44.9±4.5	54.3±5.9	59.7±2.2	62.7±4.4	33.1±3.2
Gotta-base	57.8±2.6	40.8±5.6	47.1±1.1	36.2±1.6	41.8±5.4	45.9±1.7	55.2±2.5	20.5±1.9
Gotta	**74.6±1.9**	**63.3±8.0**	**58.9±1.9**	**47.3±2.5**	**56.8±3.9**	**59.8±2.1**	**66.1±3.1**	**38.5±5.3**
Improvement%	2.9	34.3	2.8	5.3	4.5	0.1	5.4	16.1
32 Examples								
RoBERTa	18.2±5.1	10.5±1.8	22.9±0.7	3.2±1.7	13.5±1.8	10.4±1.9	23.3±6.6	4.3±0.9
SpanBERT	25.8±7.7	15.1±6.4	25.1±1.6	7.2±4.6	14.6±8.5	13.2±3.5	25.1±3.3	7.6±2.3
Splinter	59.2±2.1	28.9±3.1	33.6±2.4	27.5±3.2	34.8±1.8	34.7±3.9	36.5±3.2	27.6±4.3
FewshotQA-base	59.5±2.2	50.3±3.1	48.1±2.1	40.7±2.3	49.4±3.2	48.2±1.7	56.7±2.2	24.1±4.2
FewshotQA	73.8±2.2	56.7±5.9	**60.6±2.4**	50.0±2.8	61.4±3.6	61.6±1.5	66.9±4.7	41.7±4.2
Gotta-base	62.7±1.8	47.7±4.5	49.6±1.3	41.4±2.4	49.8±2.5	49.6±1.3	57.6±3.0	28.1±1.9
Gotta	**76.0±2.0**	**61.9±4.8**	59.8±2.4	**51.2±1.5**	**63.1±3.1**	**62.7±1.2**	**69.5±1.0**	**46.3±3.7**
Improvement%	3.0	9.1	-1.4	2.4	2.8	1.7	3.8	11.1
64 Examples								
RoBERTa	28.4±1.7	12.5±1.4	24.2±1.0	4.6±2.8	19.8±2.4	15.0±3.9	34.0±1.8	5.4±1.1
SpanBERT	45.8±3.3	15.9±6.4	29.7±1.5	12.5±4.3	18.0±4.6	23.3±1.1	35.3±3.1	13.0±6.9
Splinter	65.2±1.4	35.5±3.7	38.2±2.3	37.4±1.2	39.8±3.6	45.4±2.3	49.5±3.6	35.9±3.1
FewshotQA-base	66.5±1.1	52.3±2.8	51.5±1.6	43.5±2.0	54.9±2.0	50.7±1.6	64.3±2.3	31.7±2.8
FewshotQA	77.9±2.1	57.9±4.4	60.9±2.5	53.7±1.1	65.4±2.4	63.1±2.2	73.2±3.1	44.8±1.8
Gotta-base	67.7±0.9	50.6±4.0	51.5±1.3	45.7±1.6	54.6±3.1	52.0±0.8	64.9±2.6	35.5±3.5
Gotta	**78.9±0.5**	**59.6±1.9**	**63.6±1.0**	**54.3±3.0**	**66.3±2.5**	**64.3±1.7**	**73.2±1.5**	**51.2±2.8**
Improvement%	1.3	3.0	4.4	1.1	1.4	1.9	0.0	14.3
128 Examples								
RoBERTa	43.0±7.1	19.1±2.9	30.1±1.9	16.7±3.8	27.8±2.5	27.3±3.9	46.1±1.4	8.2±1.1
SpanBERT	55.8±3.7	26.3±2.1	36.0±1.9	29.5±7.3	26.3±4.3	36.6±3.4	52.2±3.2	20.9±5.1
Splinter	72.7±1.0	44.7±3.9	46.3±0.8	43.5±1.3	47.2±3.5	54.7±1.4	63.2±4.1	42.6±2.5
FewshotQA-base	70.8±0.7	45.9±2.1	53.6±1.1	48.4±1.8	58.7±0.9	56.3±0.9	73.8±1.0	37.7±1.1
FewshotQA	78.8±2.7	55.2±1.8	63.3±1.6	56.8±1.1	67.0±1.8	64.9±1.8	77.2±1.5	46.2±5.9
Gotta-base	71.3±1.3	52.8±2.0	54.2±0.7	49.8±1.6	60.2±1.6	56.3±1.4	73.1±1.9	40.3±3.2
Gotta	**80.8±1.7**	**60.0±3.6**	**64.9±1.2**	**57.4±1.2**	**69.8±1.5**	**66.7±1.8**	**78.6±2.1**	**53.3±1.7**
Improvement%	2.6	8.8	2.5	1.1	4.3	2.9	1.8	15.3

Table 2.2: Overall performance in F1 scores across all datasets when the numbers of training examples are 16, 32, 64, and 128. NQ stands for Natural Questions. Improvement% marks the relative performance improvements of Gotta compared to the best baselines. RoBERTa, SpanBERT, and Splinter have 110M parameters. FewshotQA-base and Gotta-base have 130M parameters. Both FewshotQA and Gotta have parameters of size 406M. The average improvements of Gotta over FewshotQA are significant on all eight datasets in a paired t-test (p-value < 0.05).

Model	SQuAD	TriviaQA	NQ	NewsQA	SearchQA	HotpotQA	BioASQ	TextbookQA
16 Examples								
Gotta	**74.6±1.9**	**63.3±8.0**	**58.9±1.9**	**47.3±2.5**	**56.8±3.9**	59.8±2.1	66.1±3.1	38.5±5.3
Gotta-random	72.1±2.6	53.2±8.4	56.2±4.1	46.7±2.2	54.8±6.1	**61.2±1.0**	61.9±2.3	38.4±2.7
Gotta-MTL	71.0±1.9	49.4±7.7	57.8±2.6	45.1±3.5	56.0±5.1	58.4±2.3	62.9±4.5	37.4±3.4
Gotta-what	69.8±2.9	52.0±7.3	57.9±3.3	46.8±1.8	54.9±4.4	60.1±1.0	**66.2±3.3**	**38.8±2.3**
32 Examples								
Gotta	**76.0±2.0**	**61.9±4.8**	59.8±2.4	51.2±1.5	**63.1±3.1**	62.7±1.2	69.5±1.0	**46.3±3.7**
Gotta-random	75.9±2.1	54.7±5.4	59.3±1.7	**51.5±2.2**	62.8±2.3	**63.3±1.6**	67.5±3.8	42.6±4.9
Gotta-MTL	70.9±2.4	55.5±5.8	60.0±1.4	48.7±3.2	60.8±1.7	61.4±1.2	66.7±1.9	41.5±3.6
Gotta-what	74.7±1.1	54.6±5.6	**60.4±2.2**	50.0±1.2	62.4±2.9	60.2±1.7	**70.7±1.3**	40.4±4.1
64 Examples								
Gotta	78.9±0.5	**59.6±1.9**	**63.6±1.0**	**54.3±3.0**	66.3±2.5	**64.3±1.7**	**73.2±1.5**	**51.2±2.8**
Gotta-random	**79.3±1.3**	57.9±3.4	62.2±1.6	53.0±3.0	66.1±3.4	63.8±1.7	72.8±1.7	51.1±3.3
Gotta-MTL	73.9±2.7	54.5±5.0	60.7±1.1	52.6±1.1	65.7±2.3	63.3±1.7	71.6±2.6	45.1±3.3
Gotta-what	78.7±1.2	59.2±2.5	62.7±0.9	54.2±1.6	**67.2±1.3**	64.0±1.0	70.9±3.4	48.0±1.9
128 Examples								
Gotta	80.8±1.7	**60.0±3.6**	**64.9±1.2**	57.4±1.2	**69.8±1.5**	**66.7±1.8**	**78.6±2.1**	**53.3±1.7**
Gotta-random	79.9±1.0	58.6±4.0	64.3±0.9	57.2±0.9	69.8±1.5	66.0±0.7	78.1±2.1	52.5±4.1
Gotta-MTL	77.2±1.9	54.1±1.7	62.4±1.0	53.1±2.1	65.9±1.9	64.1±1.9	76.5±1.3	47.6±2.1
Gotta-what	**80.9±1.4**	57.8±4.0	64.5±0.6	**57.6±0.6**	67.5±0.9	64.7±1.5	77.7±1.9	52.4±2.3

Table 2.3: Performance of different model variations across all datasets in F1 scores. We also conduct significant tests for Gotta-random. However, Gotta-random does not significantly outperform FewshotQA (p-value $\gg$ 0.05).

2.5.3 Analysis and Discussions

We further provide more in-depth studies to look into which steps and parts contribute the most to Gotta's performance. Looking back on the design of our model, three key modules are proposed, namely entity masking, prompt data construction, and prompt loss design. Besides, data augmentation also plays a vital role in Gotta.

2.5.3.1 Entity Masking

We start from the entity masking module. To check whether entity masking benefits the overall performance, we create a variation of Gotta called Gotta-random. In Gotta-random, we remove the entity masking module and randomly mask text spans instead of entities that appear in the Wikidata entity set. As shown in Table 2.3 comparing between Gotta and Gotta-random, we find that: (1) Randomly masking usually yields a higher variance. Although the cloze task can still be fulfilled by randomly selecting phrases, it destabilized the overall QA performance. (2) The full model outperforms the random model in most cases, which validates our hypothesis that masking entities in the context are crucial for selecting the subjects of the cloze examples, thus improving the QA task.

2.5.3.2 Prompt-tuning vs. Multi-task Learning

Prompt data construction is the second key step proposed in our Gotta framework. As an analysis, we compare prompt tuning with multi-task learning, which can be the other intuitive approach to jointly learn the QA and cloze tasks. Specifically, we denote Gotta with multi-tasking learning as Gotta-MTL.

From Table 2.3, we observe that (1) Gotta-MTL has apparently worse performance than Gotta, which validates our claim that formulating the cloze task in the same format of QA is essential. (2) Gotta-MTL is defeated by Gotta-what in most cases, meaning that the contribution of prompt is larger than that of entity masking or question text. That being said, aligning the format of QA, cloze along with that of the pre-training task contributes the most to the overall performance.

2.5.3.3 Question Templates

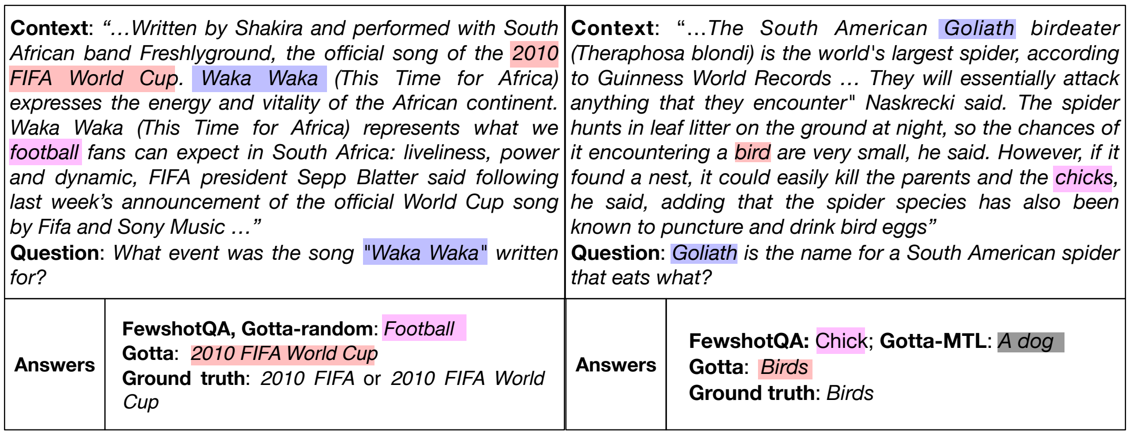
Now that we have shown that it is necessary to formulate the cloze task as prompt-tuning, a natural question is: *Does the question text have an impact on the prompt-tuning performance?* To answer this, we construct another model Gotta-what to study the effect of question text on the performance. The mere distinct between Gotta-what and the original model is the question text of the augmented data. Formally, we change the original question

$$\mathbf{q} = \textit{What is the masked entity?}$$

to the question

$$\mathbf{q} = \textit{What?}$$

Comparing the performance of GOTTA-what with that of GOTTA in Table 2.3, we observe that the two are comparable. On TriviaQA, GOTTA slightly outperforms GOTTA-what consistently while things are otherwise on any other dataset, with the two going back and forth.



Context: *"...Written by Shakira and performed with South African band Freshlyground, the official song of the 2010 FIFA World Cup. Waka Waka (This Time for Africa) expresses the energy and vitality of the African continent. Waka Waka (This Time for Africa) represents what we football fans can expect in South Africa: liveliness, power and dynamic, FIFA president Sepp Blatter said following last week's announcement of the official World Cup song by Fifa and Sony Music ..."* **Question**: *What event was the song "Waka Waka" written for?*		**Context**: *"...The South American Goliath birdeater (Theraphosa blondi) is the world's largest spider, according to Guinness World Records ... They will essentially attack anything that they encounter" Naskrecki said. The spider hunts in leaf litter on the ground at night, so the chances of it encountering a bird are very small, he said. However, if it found a nest, it could easily kill the parents and the chicks, he said, adding that the spider species has also been known to puncture and drink bird eggs"* **Question**: *Goliath is the name for a South American spider that eats what?*	
Answers	**FewshotQA, Gotta-random**: *Football* **Gotta**: *2010 FIFA World Cup* **Ground truth**: *2010 FIFA* or *2010 FIFA World Cup*	**Answers**	**FewshotQA:** Chick; **Gotta-MTL**: *A dog* **Gotta**: *Birds* **Ground truth**: *Birds*

Figure 2.3: Answers generated by different models for two test cases from TriviaQA. We match the color of the generated answers with their occurrences in the text if they are in the text. In both cases, GOTTA successfully generates the correct answer, whereas baselines without entity masking can not accurately recover the entity-level details.

2.5.3.4 Case Study

We further take a look at two concrete test cases. Figure 4.4 illustrates two examples sampled from the test set of TriviaQA. As we can see, in the left case, both FewshotQA and GOTTA-random generate the incorrect answer *Football*. While this generated answer has highly relevant semantics to the correct answer *2010 FIFA World Cup*, that answer is still not detailed enough. From this observation, we validate our claim that compared with FewshotQA and GOTTA-random without an entity masking module, the full model of GOTTA can generate the answer text in detail from the entity level. In the right case, GOTTA generates the correct answer. In contrast, although GOTTA-MTL generates the answer *A dog* that a spider could eat, it is still a wrong answer and does not even appear in the context. This difference perfectly demonstrates that prompt-tuning is beneficial to building connections between entities in the same context. Although FewshotQA returns an answer within the context, the answer is too trivial to answer the question. These two cases provide evidence to validate that entity-aware masking and prompt-style data augmentation in our proposed GOTTA are both essential to acquiring the capability of deeply understanding the complicated semantics in questions and contexts.

2.5.3.5 Effect of Augmented Data

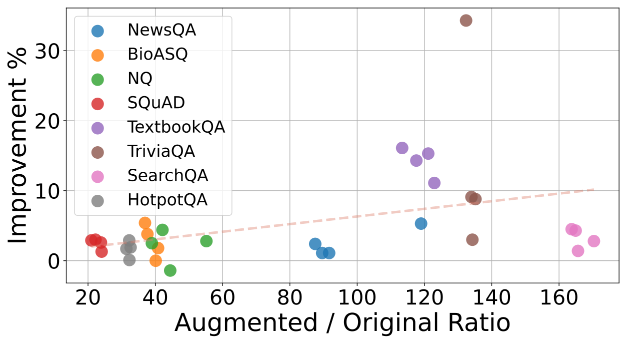


Figure 2.4: Relative performance improvement w.r.t. the number of augmented data generated per sample. There are in total 32 data points corresponding to each setting on each dataset in Table 4.2.

As shown in Figure 2.4, we proceed to study the actual effect of augmented data on the overall performance by investigating the relationship between **average augmented example per training example** and **relative performance improvement**. With the growth of average augmented data per training example, the performance gain is generally larger. Recall that we construct augmented data by raising questions on the entities detected in the context of training examples. When there are more entities in the context, GOTTA can learn more about the semantics of the entities and potentially the relations in between, thus having a deeper understanding of the context, thereby further strengthening the QA performance. However, we do observe there is not much gain on SearchQA. Our conjecture is that the contexts of SearchQA are usually very long, so it is rather hard to match the most critical entities. In the extreme case, the entity masking degenerates to MLM, omitting the role of the entities.

2.6 Conclusion and Future Work

In this work, we propose to incorporate the cloze task to improve neural machine question answering with a few training examples. The key idea is to identify and mask the informative entities in the passage and make the model predict them correctly. Through empirical experimental studies on various QA benchmarks and different few-shot settings, we show that the cloze task indeed benefits the QA task due to its commonalities. We find different ways of incorporating the cloze task improve the QA task while prompt-tuning brings the most. Looking forward, it is of interest to explore QA-dedicated pre-training and ways of pipelining pre-training and prompt-tuning for downstream few-shot QA needs.

CHAPTER 3

Learning a Small Student from Multiple Large Language Models

3.1 Abstract

Transferring the reasoning capability from stronger large language models (LLMs) to smaller ones has been quite appealing, as smaller LLMs are more flexible to deploy with less expense. Among the existing solutions, knowledge distillation stands out due to its outstanding efficiency and generalization. However, existing methods suffer from several drawbacks, including limited knowledge diversity and the lack of rich contextual information. To solve the problems and facilitate the learning of compact language models, we propose TinyLLM, a new knowledge distillation paradigm to learn a small student LLM from multiple large teacher LLMs. In particular, we encourage the student LLM to not only generate the correct answers but also understand the rationales behind these answers. Given that different LLMs possess diverse reasoning skills, we guide the student model to assimilate knowledge from various teacher LLMs. We further introduce an in-context example generator and a teacher-forcing Chain-of-Thought strategy to ensure that the rationales are accurate and grounded in contextually appropriate scenarios. Extensive experiments on six datasets across two reasoning tasks demonstrate the superiority of our method. Results show that TinyLLM can outperform large teacher LLMs significantly, despite a considerably smaller model size.

3.2 Introduction

Large language models (LLMs) have recently taken over various domains and web applications, including society [RJP23], education [ZMT23], and recommendations [WZQ23]. Although cutting-edge language models like GPT-4 and Claude-2 have shown remarkable capability in producing coherent and contextually appropriate text, their smaller counterparts often fall short, especially in tasks that demand sophisticated reasoning and a deep level of understanding [WTB22]. This discrepancy has been unveiled as the well-known scaling law of LLMs, which suggests a correlation between model size and reasoning, linguistic, and generalization capabilities [KMH20]. However, deploying these colossal models in a real-world setting poses significant challenges due to their computational requirements and resource demands, underscoring the importance of building efficient, smaller models that retain the power of their larger counterparts. Previous studies have shown that knowledge distillation is an instrumental tool in mitigating the performance gap between larger LLMs and smaller ones [WWL23, HLY23]. Examples of effective distillation methods include DistilBERT [SDC19], Alpaca [TGZ23] and Vicuna [ZC23].

However, existing methods suffer from two major drawbacks: (1) **Limited Knowledge Diversity**: Current research predominantly employs a single-teacher approach, which confines the learning scope of the student model to the knowledge derived from its own training and architecture designs [HSY22, MMA22, Li23, WCI22]. This restricts the student model to a single perspective, potentially overlooking the diverse problem-solving strategies and reasoning capabilities exhibited by different models, limiting its breadth and depth of understanding. (2) **Lack of Rich Contextual Information**: While rationales play a vital role in effective reasoning [WW22, KGR22], current research primarily focuses on leveraging ground truth labels, which indicate the correct answer but do not provide insights into the reasoning and thought process behind that answer. In other words, learning the ground truth labels exclusively failed to capture the nuanced decision-making processes of the teachers, which are crucial for tasks requiring complex reasoning and interpretation.

To solve these issues, we propose TinyLLM, a paradigm that facilitates the learning of a small student LLM by distilling knowledge from multiple large teacher LLMs with rationale guidance. Specifically, TinyLLM mitigates the limited knowledge diversity issue by involving multiple teacher models as *co-advisors*, which introduces a richer, varied knowledge source for the student to learn from. To fully exploit each teacher model and mitigate the lack of rich contextual information problem, TinyLLM asks the teacher for the credible rationales to support the answers, thereby providing the student with a deeper understanding of the problem-solving process. By learning from multiple teachers, the student model can inherit a broader range of skills and knowledge, leading to better generalization capabilities. In addition, to ensure the rationales are grounded in contextually appropriate scenarios and reflect the true underlying reasoning procedure, TinyLLM features an in-context example generator and a teacher-forcing Chain-of-Thought strategy, making the teachers understand the task through demonstrations and therefore generate the accurate rationales.

To thoroughly evaluate our approach, we conduct experiments on six datasets in commonsense and biomedical reasoning tasks. The results show that the usage of our paradigm enhances performance by **+5.07%** to **+15.69%** compared to full fine-tuning. Compared to the teacher models, TinyLLM achieve superior performance improvement, e.g., up to **+23.40%** with significantly smaller model size, e.g., **1.1%** to **26.0%**. Furthermore, compared to the state-of-the-art distillation method, we improve the performance by **+10.00%** to **+11.79%** across different model sizes. In addition, we perform efficiency analyses, ablation studies, parameter sensitivities, and case studies to demonstrate and validate the effectiveness of the proposed method. To summarize, our main contributions are as follows:

- We identify two critical limitations in the existing knowledge distillation landscape for LLMs: 1) limited knowledge diversity and 2) lack of rich contextual information.
- To solve these two problems, we propose TinyLLM, a novel knowledge distillation paradigm to learn a small student LLM from multiple large teacher LLMs.

- TinyLLM encompasses several innovative designs including an in-context example generator, a teacher-forcing Chain-of-Thought strategy, and a joint learning objective from various teachers.
- Extensive experiments validate the superiority of TinyLLM across six datasets and two reasoning tasks, with performance improving by up to **+15.69%** compared to full fine-tuning, up to **+23.40%** compared to teacher models, and up to **+11.79%** compared to state-of-the-art. In addition, TinyLLM holds a significantly smaller model size, e.g., **1.1%** to **26.0%** compared to the teachers.

3.3 Related Work

In this section, we review existing work including large language models, chain of thought, and knowledge distillation.

Large Language Models. Recent advancements have seen the proposal of various Large Language Models (LLMs) [CHL22, TMS23, TM23, BMR20], which have showcased remarkable performance across a spectrum of tasks [SX23, WRT24, LHT24, TZT24, LDT24]. Central to these developments is question answering, a task that necessitates intricate reasoning and comprehensive understanding skills for text interpretation and generating suitable responses to queries [LMX22, ZLW21, CJC24b, TSW24a]. Despite their formidable learning capabilities, LLMs encounter limitations in accurately capturing factual knowledge and are prone to producing unsubstantiated responses [ZZL23, JLF23, BCL23]. Moreover, the extensive number of parameters within LLMs complicates their adaptation for downstream tasks [WSF22, SPN22]. To mitigate these challenges, several approaches aim to lessen the dependency on intensive training and reduce computational costs [LAC21a, LL21, HSW22]. For example, Prompt Tuning [LAC21a] employs soft prompts to adapt pre-trained LLMs for specific tasks.

Chain of Thought. Recently, the use of rationales generated by LLMs has become a popular trend, setting itself apart from the traditional reliance on human-generated rationales [HB22]. Previously, human rationales have been used for model regularization [RHD17], as additional inputs for predictions [RMX19], and to improve model performance [ZEP07, ZMW16, CRL18, HBM19, PBD22]. They also serve as gold standard labels for generating similar rationales to enhance interpretability [WMS21, NRL20, EAB22]. However, the cost of human rationales limits their widespread use. On the other hand, modern LLMs can generate high-quality reasoning steps to explain their predictions [WW22, KGR22], improving performance in few-shot or zero-shot learning [WW22, KGR22, WCI22] and serving as self-improvement data [ZWM22, HGH22]. However, LLMs' size hinders their deployment in practice. Correspondingly, recent research explores leveraging generated rationales for training smaller, task-specific models with minimal computational and memory overhead [WLX21, HSY22, MMA22, Li23]. For example, PINTO [WCI22] presents an LLM pipeline that rationalizes via prompt-based learning. However, they still rely on an LLM for rationale generation at test-time, not fully addressing deployment challenges. In this work, we propose a multi-task learning paradigm with superior chain-of-thought reasoning capabilities, avoiding the dependence on teacher models during the test phase.

Knowledge Distillation. Recent LLMs such as PaLM 540b [CND23] present formidable challenges in terms of inference and fine-tuning, primarily attributable to the extensive computational resources they necessitate. This dependency and requirement on computation underscore the pivotal role of knowledge distillation [HVD15, TZG23, TPZ23, GYM21, CH19], which has proved to alleviate the resource limitation by training a smaller model to mimic the large teacher model. In addition, the employment of the Chain-of-Thought paradigm has facilitated the generation of deliberative reasoning samples from the teacher models [HSY22], allowing student models to concurrently grasp both the answers and the intricate reasoning of the teachers. This process strengthens the student model through multi-task learning endeavors [HLY23]. Correspondingly, efforts have been made to generate various rationales for each inquiry, seeking to ensure consistency in the predictions [CWQ23]. However, depending

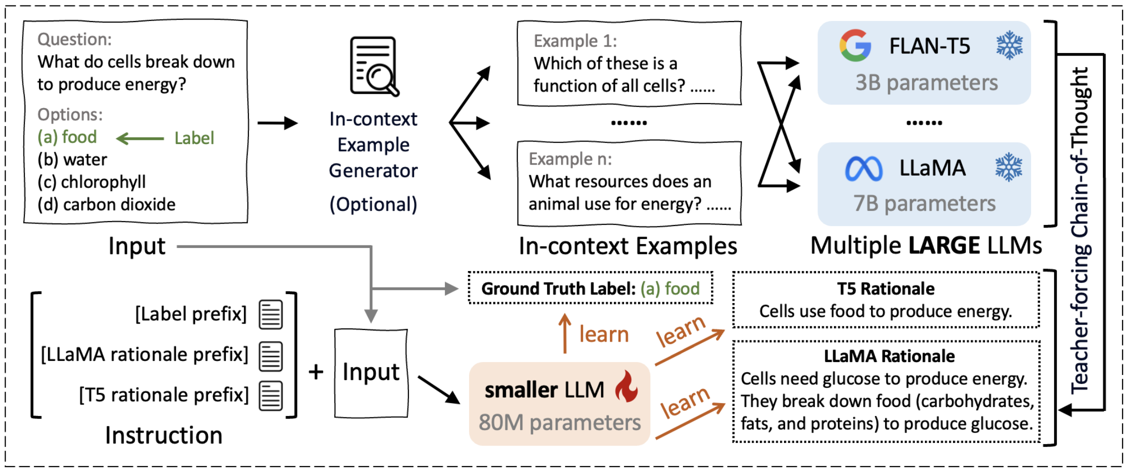

Question:
What do cells break down to produce energy?
Options:
(a) food
Label
(b) water
(c) chlorophyll
(d) carbon dioxide
Input
In-context Example Generator
(Optional)
Example 1:
Which of these is a function of all cells? ......
......
Example n:
What resources does an animal use for energy? ......
In-context Examples
FLAN-T5
3B parameters
......
LLaMA
7B parameters
Multiple LARGE LLMs
Teacher-forcing Chain-of-Thought
Ground Truth Label: (a) food
T5 Rationale
Cells use food to produce energy.
LLaMA Rationale
Cells need glucose to produce energy. They break down food (carbohydrates, fats, and proteins) to produce glucose.
learn
learn
learn
smaller LLM
80M parameters
[Label prefix]
[LLaMA rationale prefix]
[T5 rationale prefix]
Instruction
+
Input

Figure 3.1: Pipeline of TinyLLM: Given an input question, we first generate in-context examples and obtain rationales from multiple large LLMs via a teacher-forcing Chain-of-Thought strategy. Later, a small student LLM is trained to integrate rationales from different teachers via multi-task instruction tuning, along with the ground truth label.

on the rationales from a single teacher model introduces bias and compromises thoroughness, thus not fully tapping into the capabilities of multi-teacher learning paradigms. These paradigms, by their very nature, hold the potential to enhance knowledge diversity, an aspect that remains largely unexplored.

3.4 Method

In this section, we formally present TinyLLM to resolve the challenges described in the Introduction. In particular, we start by describing the preliminary. Next, we introduce the details of TinyLLM by first obtaining rationales from multiple teachers, and then learning a small student using the obtained rationales. The TinyLLM pipeline is shown in Figure 3.1.

3.4.1 Preliminary

Multiple Choice Question Answering. A k-way multiple choice question answering (MCQA) is defined as follows: Given a question Q_i, a set of candidate answer options $O_i = \{O_{i1}, O_{i2}, ..., O_{ik}\}$, the model is tasked with selecting the correct answer from the set O_i, such that the selected answer aligns the ground truth label A_i.

Knowledge Distillation. The knowledge distillation process begins with the teacher model, denoted as T parameterized by θ_T, which has been pre-trained on a large corpus. Later, the student model, S, with parameter θ_S, is tasked with distilling knowledge directly from T, leveraging the strong capabilities of T. Correspondingly, the objective function can be formulated as: $\mathcal{L} = \ell(S, T)$, where ℓ indicates the learning function, e.g., cross-entropy loss between the prediction output of the student and the target output generated by the teacher.

3.4.2 Obtaining Rationales from Teachers

In-context Example Generator. To enable the rationales that are generated by teachers to be grounded in contextually appropriate scenarios, we introduce an optional in-context example generator. This tool is designed to produce in-context examples for any given input, providing more detailed information about the input data and task. For simplicity, we select the examples randomly within the same dataset. This aids the teacher LLMs in comprehending the nature and specifics of the task more deeply. By integrating this generator, we facilitate a more informed and nuanced generation of rationales by the teacher models, enhancing the learning experience for the student model.

Teacher-forcing Chain-of-Thought. In addition, we design a teacher-forcing strategy to ensure the validity of the rationales. Compared to existing methods that simply employ regular chain-of-thought (CoT) mechanisms [WW22, KGR22], wherein an LLM is prompted with sets of questions and options $\{Q_i, O_i\}$ to elicit rationales R_i directly, TinyLLM posits a distinct advantage in integrating the correct answer A_i into the input. We hypothesize that the inclusion of A_i alongside Q_i and O_i facilitates a more nuanced understanding of the input context and the correct logical rationales leading to the answer, thereby facilitating a more informed and accurate generation process. Specifically, we consider the concatenation of questions, options, and answers $\{Q_i, O_i, A_i\}$ as the input to LLMs.

Rationales from Multiple Teachers. Given M teachers, TinyLLM pioneers the usage of a multi-teacher architecture in which each teacher T^m is an LLM. In particular, the rationale R_i^m produced by a specific teacher model θ_{T^m} for the ith question is derived using the question Q_i, options O_i, correct answer A_i, and in-context examples P_i. The process is formalized as follows:

$$R_i^m = T^m(Q_i, O_i, A_i, P_i; \theta_{T^m}). \tag{3.1}$$

3.4.3 Learning a Small Student

A straightforward strategy to incorporate rationales as supervision is to append each rationale R_i^m generated by the teacher models as supplementary input to the student model, along with the question Q_i and options O_i. However, this method faces challenges due to limitations in computational resources at the inference stage, especially because rationales must be pre-generated for every data sample in both training and test sets [WCI22]. To overcome this issue, we employ rationales as a form of supervisory signal during the training process to develop a model that is adept at generating its explanations. Subsequently, this trained model can be utilized on the test set, eliminating the need for pre-generated rationales to facilitate accurate reasoning. Specifically, TinyLLM integrates rationales from multiple teacher models into a unified multi-task instruction tuning framework. This necessitates the assignment of a unique prefix p for distinguishing between learning tasks from different teachers. The student model is trained not only to predict labels but also to generate rationales akin to those produced by the teachers. Accordingly, the overall loss function

$\mathcal{L}$ is as follows:

$$\mathcal{L} = \mathcal{L}_A + \sum_{m=1}^{M} \alpha^m \mathcal{L}_{T^m}, \tag{3.2}$$

where $\mathcal{L}_A$ denotes the objective of learning from ground truth answers, $\mathcal{L}_{T^m}$ indicates the objective of learning from m-th teacher, α^m is the importance weight for T^m, and M is the number of teacher LLMs. Formally, $\mathcal{L}_A$ and $\mathcal{L}_{T^m}$ are defined as follows:

$$\mathcal{L}_A = \frac{1}{N} \sum_{i=1}^{N} \ell(S(Q_i, O_i, p_A; \theta_S), A_i), \tag{3.3}$$

$$\mathcal{L}_{T^m} = \frac{1}{N} \sum_{i=1}^{N} \ell(S(Q_i, O_i, p_m; \theta_S), R_i^m), \tag{3.4}$$

where N is the number of data samples, ℓ indicates the cross-entropy loss between the predicted and target tokens. Here $\mathcal{L}_A$ encourages the student S to generate ground truth answer A_i by minimizing the difference between it and the student output given the question Q_i, options O_i, and instruction prefix p_A for generating answers. On the other hand, $\mathcal{L}_T^m$ facilitates the student S to mimic the reasoning capability of teacher T^m by learning from its rationale R_i^m, with the guidance of instruction prefix p_m for T^m.

3.5 Experiments

In this section, we rigorously test TinyLLM against a series of empirical benchmarks across varied datasets and reasoning tasks. In addition, we conduct efficiency analyses, ablation studies, parameter sensitivities, and case studies to demonstrate the effectiveness and superiority of our method.

3.5.1 Experimental Setup

Datasets. For the task of commonsense reasoning, we use OpenBookQA (OBQA) [MCK18], The AI2 Reasoning Challenge (ARC) [CCE18], Physical Interaction Question Answering

(PIQA) [BZL20], and RiddleSense (Riddle) [LWY21]. For the task of biomedical reasoning, we consider PubMedQA (PQA) [JDL19] and BioASQ [TBM15a].

Baselines. We benchmark TinyLLM against the teacher's performance and various baseline methods, including Inference-only that only leverage the pre-trained model for evaluation without training, and multiple fine-tuning methods that provide further adaptation. In particular, we consider LoRA [HSW22], full fine-tuning, and the PINTO method [WCI22] for the fine-tuning methods. We also compare TinyLLM with various knowledge distillation strategies. To illustrate, we include standard KD [HVD15] that enforces the student to mimic the teacher's labels and the Distill-step-by-step method [HLY23] that leverage rationales.

Implementation Details. For all distillation baselines and TinyLLM, we set the learning rate to 5×10^{-5}, batch size to 8, maximum input length to 1024, and epoch to 1. For Distill-step-by-step and TinyLLM, the trade-off weights α_{T_n} are explored within {0.01, 0.1, 0.5, 1, 2, 3}. We report the best result for Distill-step-by-step by leveraging different teacher models. For the choice of LLMs, we use FLAN-T5 [CHL22] small (80M), base (250M), and large (780M) as the student, and FLAN-T5 xlarge (3B) and LLaMA 2-chat [TM23] (7B) as teachers. Experiments are conducted on four NVIDIA Tesla H100 GPUs.

3.5.2 Performance Comparison

Comparison to Baselines Methods. The results of six datasets and two reasoning tasks are shown in Table 3.1. From the table, we observe that the employment of a full fine-tuning method, despite its theoretically enhanced capacity for parameter adjustment, does not consistently yield superior results to LoRA. Conversely, TinyLLM demonstrates substantial performance enhancements across all datasets and LLM sizes. Quantitatively, TinyLLM secures an average performance improvement by **+15.69%**, **+11.55%**, and **+5.07%** compared to full fine-tuning for 80M, 250M, and 780M student models, respectively. Compared to state-of-the-art distillation method Distill-step-by-step, TinyLLM achieves an improvement of **+10.00%**, **+10.32%**, and **+11.79%** correspondingly. This demonstrates the effectiveness of TinyLLM and highlights the advantages of the designs in our method.

Comparison to Teachers. TinyLLM also shows superior performance compared to the teacher models. For example, a 780M student can achieve an average performance of 73.88 across different datasets, which is **+14.56%** better than the 3B teacher and **+23.40%** better than the 7B teacher. Moreover, an even smaller student model with 250M parameters can outperform the teachers (+**0.82%** to 3B, +**8.60%** to 7B) while using only **8.3%** and **3.6%** of the teacher parameters.

3.5.3 Efficiency Analysis of Training Set Size in Knowledge Transfer

Advantage Over Standard Knowledge Distillation. With the aim of fully evaluating the proposed model, we test the performance of TinyLLM against state-of-the-art across different sizes of training sets. As shown in Figure 3.2, TinyLLM demonstrates superior performance in comparison to the state-of-the-art Distill-step-by-step method when using various ratios of the training data. In addition, we observe that when using more training data, the performance improves. Notably, in certain scenarios, our model, even when trained with just 12.5% of the training set size, can surpass the performance of Distill-step-by-step, which uses a considerably larger dataset. This is evident in the case of the PQA dataset, where TinyLLM achieves performance that is comparable to or even exceeds that of Distill-step-by-step, despite being trained on a much smaller portion of the dataset. This observation holds true for both the 80M and 250M student models, indicating that the superiority of our model across models of different sizes.

Outperforming full fine-tuning. TinyLLM further demonstrates its superiority compared to full fine-tuning using all the dataset. Remarkably, in the situations of training a 250M TinyLLM on ARC and PQA datasets, alongside training a 80M TinyLLM on the PQA dataset, only 12.5% of the training data is required to exceed the benchmarks set by full fine-tuning. In addition, when training a 80M TinyLLM on the ARC dataset, a 75% reduction in training samples is adequate for TinyLLM to attain a higher accuracy than

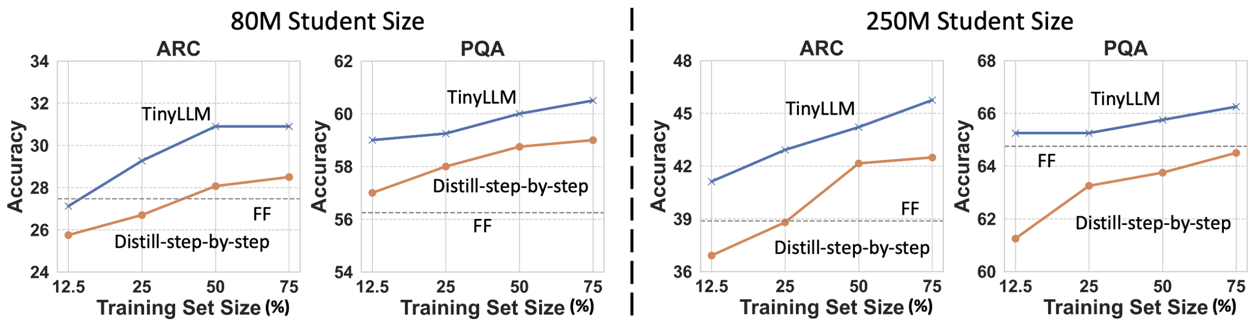


Figure 3.2: A comparative analysis of TinyLLM against the state-of-the-art Distill-step-by-step method using 80M and 250M FLAN-T5 model architectures across various training set sizes. Dotted line indicates the full fine-tuning (FF) using 100% dataset. It is evident that TinyLLM consistently surpasses the performance of both Distill-step-by-step and full fine-tuning. Notably, TinyLLM achieves this superior accuracy while employing substantially fewer training examples.

what is achieved through full fine-tuning on all the dataset.

3.5.4 Ablation Study

For a comprehensive evaluation, we conduct ablation studies to validate the contribution of the in-context example generator and rationales from multiple teachers in enhancing the reasoning capabilities of the distilled LLM. To facilitate this, we create three ablation variants of TinyLLM, each designed to assess the impact of specific components on the overall performance:

- **w/o in-context** rules out the use of in-context examples during rationale generation. This variant tests the efficacy of in-context examples in guiding the student model to generate more accurate and relevant rationales.
- **w/o LLaMa** and **w/o T5** exclude the rationale supervision from the corresponding teacher model during distillation. These variants help in understanding the individual contributions of each teacher's rationales in enriching the student model's learning experience and overall performance.
- **w/o diverse teachers** excludes the weaker teacher model and generates multiple ratio-

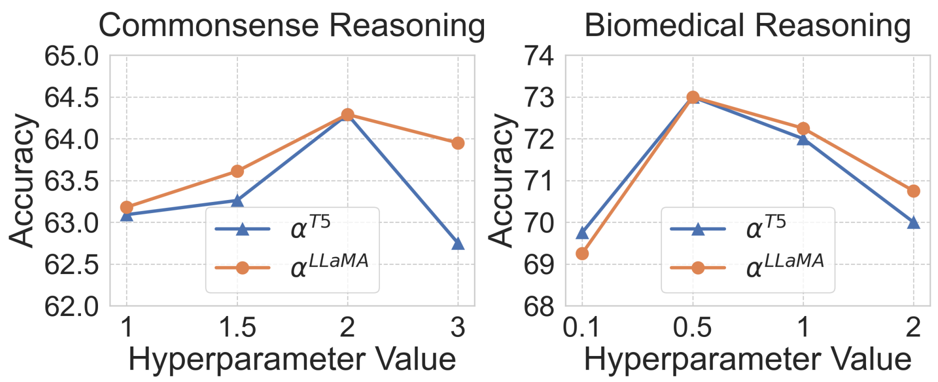


Figure 3.3: Performance w.r.t. different values of weight α.

nales from the stronger teacher model. This variant tests the effectiveness of using diverse teachers.

- **w/o teacher-forcing** excludes the teacher-forcing strategy during rationale generation. This variant tests the efficacy of the teacher-forcing strategy for generating higher-quality rationales.

The results, as detailed in Table 3.2, provide a comparative analysis between the ablation variants and the complete TinyLLM model. From Table 3.2, we have the following observations: (1) TinyLLM outperforms all ablation variants. (2) There is no significant performance gap between the ablation variants. These observations confirm the hypothesis that absorbing high-quality rationales from multiple teacher models significantly contributes to the refinement of the reasoning capabilities of the distilled LLM. The observations also imply a balanced importance between the in-context examples and supervision of multiple teachers.

3.5.5 Parameter Sensitivity

In order to fully evaluate the model, we conduct parameter sensitivity experiments on dataset ARC for commonsense reasoning and PQA for biomedical reasoning. The results are shown in Figure 3.3. In particular, we focus on the exploration of trade-off weights α^{T5} and α^{LLaMA}, which reveals the model's adaptability across different choices of parameter values. Based

ARC	PIQA
Input: Question: When cold air is at the tops of mountains, the air Options: (a) will flow toward areas of lower pressure in valleys. (b) will flow toward areas of higher pressure in valleys. (c) is considered to be less dense than warmer air. (d) is considered to be free of oxygen atoms.	**Input:** Question: how to cut sticky desserts Options: (a) Run a knife under very hot water and dry it off (b) Run a knife under very cold water and dry it off
Generated Rationales: T5 Rationale: Cold air is at the tops of mountains, so it has less density. Denser air is less likely to flow. Cold air is at the tops of mountains, so it has less density. ✗ LLaMA Rationale: The cold air at the tops of mountains is denser than the warmer air in the valleys due to the lower temperature. As a result, the cold air will flow downward, toward areas of lower pressure, which are typically in the valleys. ✓	**Generated Rationales:** T5 Rationale: Sticky desserts are difficult to cut with a knife. To cut sticky desserts, you need to run a knife under very hot water and dry it off. ✓ LLaMA Rationale: (a) Running a knife under very hot water and drying it off will not help. The hot water will only make the dessert stickier. (b) Running a knife under very cold water and drying it off is a better option. The cold water will help to harden the dessert slightly, making it easier to cut. ✗
Predicted Answer: Ground Truth: (a) T5: (c) ✗ LLaMA: (a) ✓ Full Fine-tuning: (b) ✗ Distill-step-by-step: (c) ✗ TinyLLM: (a) ✓	**Predicted Answer:** Ground Truth: (a) T5: (a) ✓ LLaMA: (b) ✗ Full Fine-tuning: (b) ✗ Distill-step-by-step: (b) ✗ TinyLLM: (a) ✓

Table 3.3: Case study of different models' prediction. Examples are selected from the ARC and PIQA datasets. In both cases, TinyLLM successfully generates the correct answer.

on the figure, we can derive the following observations: 1) the optimal parameters for various datasets and tasks differ. The reason for this phenomenon is that biomedical reasoning questions are often lengthy and complex, weakening the impact of rationales from teachers and making a small value of α sufficient. In contrast, commonsense reasoning questions are typically concise and straightforward, making the rationales from teacher models valuable and leading to a large value of α. 2) Increasing α generally improves performance, attributed to the Chain-of-Thought reasoning of the student model. While accuracy may not directly reflect reasoning quality, the prediction capability benefits from the multi-task learning process. However, excessively high α values degrade the performance by shifting the focus from prediction to reasoning. 3) The sensitivity of α^{T5} and α^{LLaMA} varies across datasets. For commonsense reasoning, α^{T5} is more sensitive than α^{LLaMA}, whereas, for biomedical reasoning, their sensitivities are similar, indicating that rationales from various teacher models can have different insights and contributions to the particular task.

3.5.6 Case Study

To analyze the underlying reason for the superiority of TinyLLM in a more intuitive perspective, we perform case studies by comparing the predictions of different models. Table

3.3 presents two examples we randomly select from the ARC and PIQA datasets.

In the first example, T5 provides a completely incorrect rationale while LLaMA generates a meaningful rationale. Accordingly, T5 has a wrong prediction while LLaMA predicts correctly. Full fine-tuning results in an incorrect answer, and the state-of-the-art Distill-step-by-step method also predicts incorrect. The reason behind this is that the distillation method, which utilizes T5's reasoning, introduces noises that misguide the student model, even though T5 significantly outperforms LLaMA in accuracy on the ARC dataset (as shown in Table 3.1). However, TinyLLM demonstrates its ability to infer the correct answer (a) with the guidance of rationales from both teachers.

In the second example, LLaMA, Distill-step-by-step, and full fine-tuning all produce the incorrect prediction. Although LLaMA achieves a significantly better performance on the PIQA dataset compared to the T5 (i.e., accuracy 78.80 compared to 58.43 according to Table 3.1), LLaMA still returns an incorrect rationale. This indicates that the reliance on a single-teacher model can be misleading. On the other hand, TinyLLM can generate the correct prediction (a). These examples underscore the significance of multi-teacher learning, as proposed in our TinyLLM framework, which provides the student model with diverse contextual insights essential for effective learning and reasoning.

3.6 Conclusion

In this paper, we propose TinyLLM, a novel knowledge distillation paradigm to learn a small student LLM from multiple large teacher LLMs. TinyLLM involves several principled designs, such as learning contextually appropriate rationales using an in-context example generator, enabling the credibility of rationales with a teacher-forcing Chain-of-Thought strategy, and inheriting a wider range of knowledge from various teachers. Our extensive empirical evaluation and in-depth analysis, conducted across six datasets spanning two reasoning tasks, demonstrate that TinyLLM brings significant and consistent improvements by up to 15.69% over full fine-tuning, up to **+23.40%** over teacher models, and up to **+11.79%** over state-of-the-art. Moreover, TinyLLM holds a significantly smaller model size, e.g., **1.1%** to **26.0%** compared to the sizes of the teachers.

3.7 Limitations

While our study provides valuable insights into knowledge distillation, it has several limitations. Firstly, we focused exclusively on question-answering tasks, not extending our investigation to generative tasks. Secondly, due to hardware constraints, our research was limited to models with a maximum scale of 7 billion parameters. Additionally, we only examined TinyLLM with two specific teacher models, leaving other LLMs unexplored. Future research using larger-scale models and a broader range of teacher models would be beneficial to validate our findings and improve the methodologies presented in this study.

Method	Commonsense Reasoning				Biomedical Reasoning		**Total**
	OBQA	**ARC**	**PIQA**	**Riddle**	**PQA**	**BioASQ**	
3B/7B Teacher							
FLAN-T5 xlarge	69.20	68.24	58.43	53.73	71.50	65.85	64.49
LLaMA 2	58.60	45.90	78.80	47.65	54.50	73.75	59.87
80M Student; Size: 2.7%/1.1%							
Inference	16.60	19.31	20.78	13.33	38.00	47.97	26.00
OOD PINTO	46.40	26.87	48.10	25.29	60.00	80.49	47.86
LoRA	37.80	27.12	39.93	39.80	53.75	78.05	46.08
Full Fine-tuning	41.60	27.47	42.33	42.75	56.25	78.86	48.21
Standard KD	45.80	29.53	49.29	36.27	58.00	81.30	49.43
Distill-step-by-step	46.40	30.47	50.38	36.67	59.00	81.30	50.70
TinyLLM	**49.40**	**33.05**	**53.65**	**51.18**	**62.00**	**85.37**	**55.78**
Δ_{FF}	↑ 18.75%	↑ 20.31%	↑ 26.74%	↑ 19.72%	↑ 10.22%	↑ 8.26%	↑ 15.69%
$\Delta_{Distill}$	↑ 6.47%	↑ 8.47%	↑ 6.49%	↑ 39.57%	↑ 5.08%	↑ 5.01%	↑ 10.00%
250M Student; Size: 8.3%/3.6%							
Inference	31.00	23.00	30.47	30.78	48.00	57.72	36.83
OOD PINTO	50.40	38.63	52.12	34.90	61.75	82.93	53.46
LoRA	51.40	37.25	47.66	53.14	62.00	82.93	55.73
Full Fine-tuning	56.60	38.88	47.55	52.55	64.75	89.43	58.29
Standard KD	55.40	43.69	55.93	42.94	64.25	86.18	58.07
Distill-step-by-step	56.80	43.86	56.37	45.69	64.75	86.18	58.94
TinyLLM	**64.20**	**48.50**	**60.17**	**60.78**	**66.25**	**90.24**	**65.02**
Δ_{FF}	↑ 13.43%	↑ 24.74%	↑ 26.54%	↑ 15.66%	↑ 2.32%	↑ 0.91%	↑ 11.55%
$\Delta_{Distill}$	↑ 13.03%	↑ 10.58%	↑ 6.74%	↑ 33.03%	↑ 2.32%	↑ 4.71%	↑ 10.32%
780M Student; Size: 26.0%/11.1%							
Inference	50.40	51.07	51.90	39.80	64.25	63.41	53.47
OOD PINTO	62.20	52.10	57.13	42.94	70.00	84.55	61.49
LoRA	64.00	57.77	57.02	68.63	70.25	86.18	67.31
Full Fine-tuning	71.20	62.92	58.43	68.82	70.25	90.24	70.31
Standard KD	65.80	56.05	60.72	52.94	70.00	86.99	65.42
Distill-step-by-step	66.80	57.42	61.37	53.92	70.00	86.99	66.08
TinyLLM	**74.40**	**64.29**	**67.90**	**70.98**	**73.00**	**92.68**	**73.88**
Δ_{FF}	↑ 4.49%	↑ 2.18%	↑ 16.21%	↑ 3.14%	↑ 3.91%	↑ 2.70%	↑ 5.07%
$\Delta_{Distill}$	↑ 11.38%	↑ 11.96%	↑ 10.64%	↑ 31.64%	↑ 4.29%	↑ 6.54%	↑ 11.79%

Table 3.1: Overall experimental results. The best results across different datasets and LLM sizes are highlighted in bold. Δ_{FF} and $\Delta_{Distill}$ represent the relative performance improvement of TinyLLM to Full Fine-Tuning and Distill-step-by-step, respectively. Accuracy is used as the evaluation metric.

Variant	Commonsense				Biomedical	
	OBQA	**ARC**	**PIQA**	**Riddle**	**PQA**	**BioASQ**
w/o in-context	73.20	63.09	66.27	69.22	70.75	86.99
w/o LLaMA	73.00	62.32	66.70	68.82	69.25	87.81
w/o T5	73.80	61.80	66.49	68.63	69.50	88.62
w/o diverse teachers	73.80	62.49	66.81	68.82	70.00	89.43
w/o teacher-forcing	73.80	60.94	65.94	69.02	70.25	90.24
TinyLLM	**74.40**	**64.29**	**67.90**	**70.98**	**73.00**	**92.68**

Table 3.2: Impact of in-context examples, the teacher-forcing strategy and contributions of various teachers.

Part II

Minimally Supervised Data Generation and Selection

CHAPTER 4

MinPrompt: Graph-based Minimal Prompt Data Augmentation for Few-shot Question Answering

4.1 Abstract

Recent advances in few-shot question answering (QA) mostly rely on the power of pre-trained large language models (LLMs) and fine-tuning in specific settings. Although the pre-training stage has already equipped LLMs with powerful reasoning capabilities, LLMs still need to be fine-tuned to adapt to specific domains to achieve the best results. In this paper, we propose to select the most informative data for fine-tuning, thereby improving the efficiency of the fine-tuning process with comparative or even better accuracy on the open-domain QA task. We present MinPrompt, a minimal data augmentation framework for open-domain QA based on an approximate graph algorithm and unsupervised question generation. We transform the raw text into a graph structure to build connections between different factual sentences, then apply graph algorithms to identify the minimal set of sentences needed to cover the most information in the raw text. We then generate QA pairs based on the identified sentence subset and train the model on the selected sentences to obtain the final model. Empirical results on several benchmark datasets and theoretical analysis show that MinPrompt is able to achieve comparable or better results than baselines with a high degree of efficiency, bringing consistent improvements in F-1 scores.

4.2 Introduction

Question answering (QA) provides accurate responses to a series of questions based on given narrative contexts. Its diverse applications extend to areas such as chatbots [YXL19], dialogue systems [BSA18], and instant information retrieval [EKP21], making it a key pursuit in the field of natural language processing (NLP). Supervised learning has traditionally been the approach for developing efficient QA systems that deliver commendable results [CWN24, THC24]. However, this method is intrinsically restricted by its reliance on a large set of annotated QA training examples, which becomes problematic due to the substantial cost associated with acquiring expert-level annotations.

Our research focuses on the few-shot QA task, an effort to address the QA challenge with the presence of only a limited number of training examples. The prevalent approaches under the few-shot setting either introduce a new task and pre-train an extensive language model from scratch [RKB21], or they fine-tune an already pre-trained model on the given training examples [CN21, TSW24b]. The fine-tuning stage is crucial in the sense that it stimulates the power of the LLMs obtained during the pre-training stage and makes the model align with the input/output distribution of a certain domain or dataset. However, with an increasing data size for fine-tuning, the training duration increases accordingly, which is undesirable, especially when the model size is also large [Ope23b]. As such, the importance of minimal data augmentation cannot be understated. The fine-tuning data, often a limited resource in our consideration (up to 128 shots), is directly used to adjust the parameters of a pre-trained model to enhance performance on the downstream task. The data is usually labeled by domain experts and thus could be time-consuming to obtain in large quantities. On the other hand, augmented data represents a broader dataset, generated in an unsupervised manner by converting statements into question-answer pairs. In QA tasks, it is vital for a model to be exposed to a diverse range of questions, answers, and contexts to develop a robust understanding of the language and the task at hand. However, not all parts of the training data hold equal relevance or significance for the model's learning process. Some parts may contain more valuable information or more complex language structures that the model needs to understand to improve its performance. Consequently, identifying and augmenting these critical portions of the training data could substantially enhance the model's capacity to answer questions accurately and comprehensively.

To address the above challenges, we present MinPrompt, which consists of the following three modules: (1) A **sentence graph construction module** that leverages sentence graph representation to structurize the raw text. Each node in the graph symbolizes a sentence, while edges illustrate the shared entities between sentences. This sentence graph effectively encapsulates the complex interconnections between various textual elements; (2) A **data selection module** that features an approximate minimal dominating set algorithm. The algorithm is applied to the sentence graph to identify the smallest set of sentences to cover all shared entities. This module ensures efficient use of computational resources, reduces the risk of overfitting, and enhances the model's generalization ability, resulting in an overall improvement in QA performance; and (3) A **question generation module** that transforms the selected plain factual sentences into QA pairs. The synthesized QA pairs are further turned into prompts, providing a condensed, yet comprehensive representation of the text. The generated prompts serve as high-quality, information-rich training instances for the QA model. This model trained on the compact and meaningful prompts is then capable of generating accurate answers to the posed questions, all without requiring any additional explicit supervision.

In summary, our contributions are as follows:

- We propose to study minimal data augmentation for effective and efficient few-shot QA
- We introduce MinPrompt, a minimal data augmentation framework that uses a graph-based algorithm and unsupervised question generation to synthesize the most informative QA training samples out of the raw text.

- We conduct extensive experiments on publicly accessible benchmarks to validate the effectiveness of MinPrompt, and observe a solid improvement over competitive compared methods. Beyond that, we also study the necessity of different parts of the model.

4.3 Related Work

Question generation. [CWZ19] presented an answer-aware question generation (QG) model that employs reinforcement learning for improved question quality. The model incorporates a coverage mechanism to alleviate the common issue of answer-related content being left out from the generated questions. [MZZ20] developed a more sophisticated approach to answer-aware question generation. Their model uses sentence-level semantic matching and answer position inferring within a sequence-to-sequence framework, resulting in higher-quality questions. [DZJ23] proposed a two-stage framework for Conversational Question Generation (CQG). It selects sentences from a semantic graph to pick up coherent topics and then uses a classifier to determine the answer type of the question. Their approach produces more natural dialogues, as real-life interlocutors often discuss relevant content that is non-sequential. [MSY22] introduces RQUGE, a novel metric for assessing the quality of automatically generated questions. Traditional methods may unfairly penalize valid questions that don't mirror reference questions closely. RQUGE overcomes these issues by evaluating on the basis of the answerability of a question given the context. Utilizing pre-trained models for its QA scorer modules, RQUGE does not require additional training. The paper presents evidence of RQUGE's high correlation with human judgment and robustness against adversarial corruption.

Few-shot QA. Previous research in QA has mainly focused on either reusing pre-trained language models (PLMs) [LCG20, JCL20] or training a model from scratch using synthetic QA data [PSS20, LDR19, AAP19]. However, both approaches require a large amount of annotated data from the downstream QA task to fine-tune the models, which can be impractical

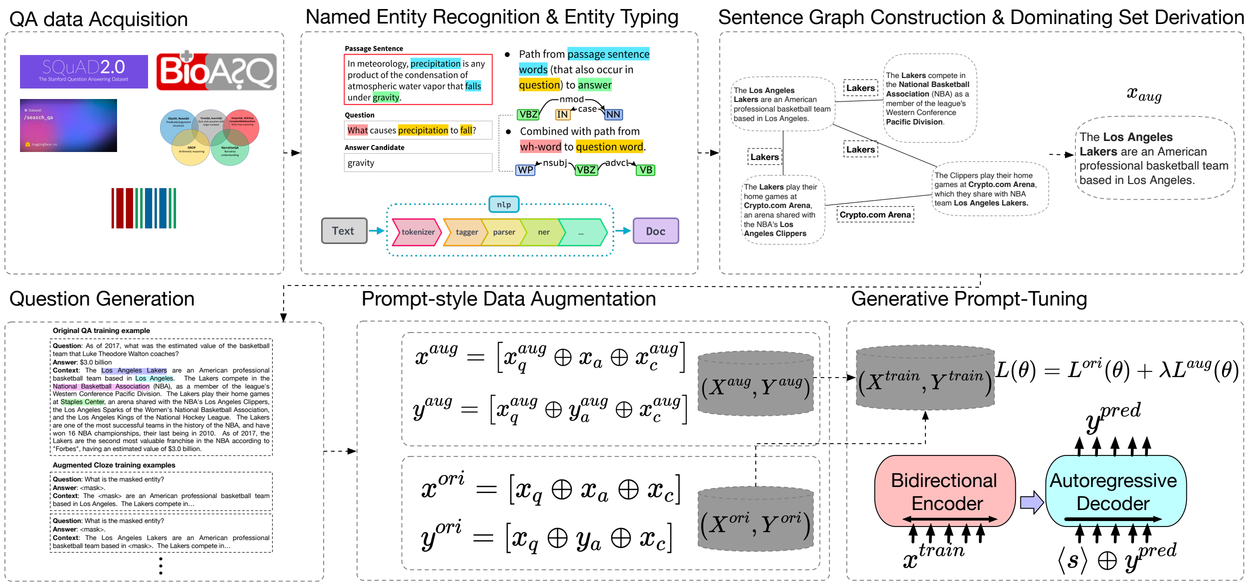

QA data Acquisition
Named Entity Recognition & Entity Typing
Sentence Graph Construction & Dominating Set Derivation
Question Generation
Prompt-style Data Augmentation
Generative Prompt-Tuning
Bidirectional Encoder
Autoregressive Decoder

Figure 4.1: **Framework overview for MinPrompt.**

in real-world scenarios. To address this problem, several recent approaches have been developed that allow the model to adapt to the downstream task with only a small amount of annotated data [RKB21, CN21]. For example, [RKB21] proposed a pretraining scheme tailored for QA tasks by designing a recurring span selection objective that aligns with the common objective in extractive QA tasks. [CN21] proposed a framework called FewshotQA, which leverages the capacity of existing PLMs by constructing a QA-style prompt that casts the QA problem as a text generation problem, specifically by concatenating the question and a mask token representing the answer span. This approach aims to save pretraining the model on a large-scale corpus. In contrast to these previous studies, this paper proposes to focus on identifying and leveraging more relevant information from the context data in addition to the annotated QA pairs to fine-tune the model in a few-shot setting.

4.4 MinPrompt: Graph-based Prompt Data Augmentation for Few-shot QA

As shown in Figure 7.1, our overall framework, MinPrompt, is designed to extract the most semantically rich and factually dense sentences to serve as candidates for conversion into a prompt tuning QA dataset. This process is guided by the principal intuition that the most informative sentences are those that encompass facts or declarations concerning a greater number of entities. Hence, these high-impact sentences should ideally cite more entities within their purview. To implement this, we start by extracting the co-reference of entities across sentences. Essentially, it allows us to map the discourse in a way that allows us to understand which sentences are speaking about the same entities. Next, we construct a graph to depict the higher-order coreference relationships. In this graph, the sentences serve as nodes, and sentences are connected if they mention the same entity. This representation allows us to establish and understand the intricate network of relationships between sentences and the entities they mention. Employing graph-based algorithms, we are then able to identify and extract the most informative sentences. These are typically sentences that have a high degree of connectivity in the graph, indicating that they mention or discuss a larger number of entities. We then transform these selected sentences into a fine-tuning dataset. The transformation process entails restructuring the sentences to meet the format requirements of a QA dataset, which generally involves turning declarative sentences into question-and-answer pairs. This method thus combines insights from computational linguistics and graph theory to achieve its goal of creating a high-quality fine-tuning dataset for QA tasks. The approach ensures that the dataset is not only rich in informative sentences, but also maps intricate entity relationships, thus providing a comprehensive context for each question and answer pair. This context helps in the training of more robust and nuanced QA systems.

4.4.1 Named Entity Recognition & Entity Typing

We use the entities as the bridge to build connections between all the factual sentences. We first conduct named entity recognition (NER) on the raw text to extract all the entity mentions along with their types. For the purpose of unsupervised QA data generation in our setting, the key lies in generating the questions given the raw text and the extracted entities (as answers). The most straightforward way to generate questions is to convert factual sentences into cloze questions [CZD23]. Creating a conventional cloze question involves extracting the original sentence containing the answer from the context and replacing the answer with a chosen token. However, training a model on these data primarily imparts text-matching and fill-in-the-blank skills, while offering minimal generalizability. As a result, we opt for a retrieval-based method to procure a sentence akin to the one containing the answer and subsequently use this to formulate a question. This has been evidenced in the work by [LDR19] and further affirmed by our preliminary experiments. Our initial step involves indexing all sentences from a Wikipedia dump using the ElasticSearch search engine. Named entities were extracted from each sentence within the Wikipedia corpus as well as from the sentences utilized as queries. We presupposed access to a named-entity recognition system and leveraged the spaCy[1] NER pipeline for this work, which is proven effective in NER and entity typing. Subsequently, for a given context-answer pair, we queried the index. This query involved using the original context sentence to return a sentence that either (1) includes the answer, or (2) does not originate from the *context*, thus discarding sentences with high similarity. Aside from guaranteeing that the retrieved sentence and the query sentence share the answer entity, we require that at least one additional matching entity be present in both the query sentence and the entire context. Finally, these retrieved sentences were introduced into our sentence graph construction module.

[1]`https://spacy.io`

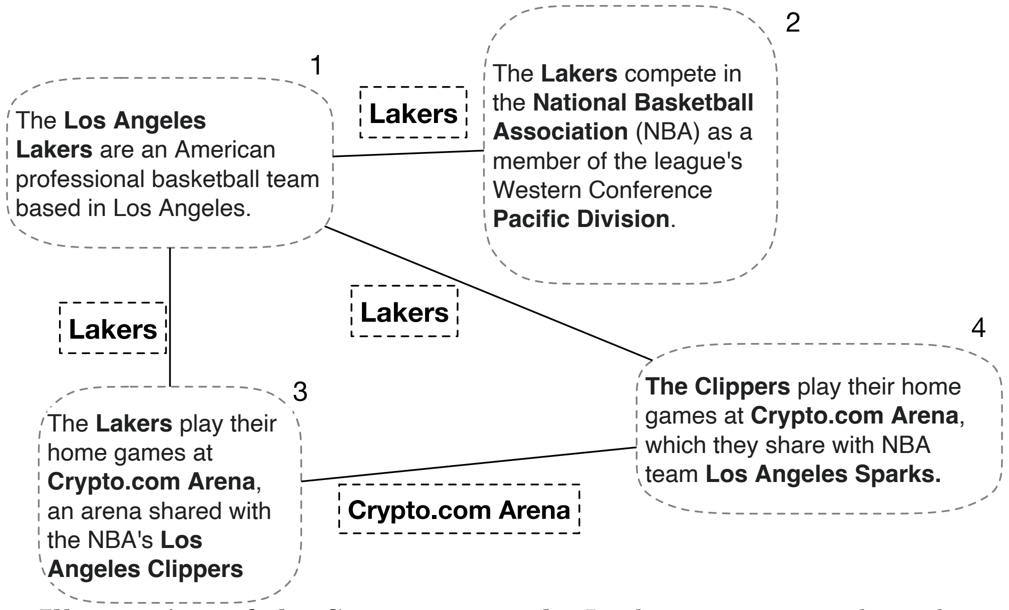

1
The **Los Angeles Lakers** are an American professional basketball team based in Los Angeles.
2
The **Lakers** compete in the **National Basketball Association** (NBA) as a member of the league's Western Conference **Pacific Division**.
Lakers
Lakers
Lakers
3
The **Lakers** play their home games at **Crypto.com Arena**, an arena shared with the NBA's **Los Angeles Clippers**
4
The Clippers play their home games at **Crypto.com Arena**, which they share with NBA team **Los Angeles Sparks.**
Crypto.com Arena

Figure 4.2: **Illustration of the Sentence graph.** In the sentence graph, nodes correspond to sentences and edges represent the coreference of entities across sentences. Sentences 1, 2 and 3 shares the entity *Lakers* while sentence 4 shares the entity *Crypto.com Arena* with sentence 3.

4.4.2 Sentence Graph Construction

As aforementioned, we construct the sentence graph to capture the semantic overlap of the factual sentences in the raw text. A proportion of the sentence graph is visualized in Figure 4.2. Upon building the sentence graph, we aim at extracting the minimal sentence set that covers the most semantics in the whole graph. Now the question becomes how we can leverage the high-order co-reference relationship to reduce the size of the training data. To dive deep into this question, we start by making the following assumption:

Assumption 1. *Suppose two sentences in a sentence set S, $\{s_e, s'_e\} \subset S$, mention the same entity e. The quality of a QA model M_S trained by a sentence set S will be similar to the quality of the other model $M_{S'}$ trained by the set $S' = S - \{s_e\}$ because $s'_e \in S'$ still cover the similar topics and knowledge in s_e.*

Based on Assumption 1, an intuitive idea of leveraging the sentence graph to effectively reduce the size of the training data is to find a minimal set of sentence nodes that can cover the whole sentence graph without losing the quality of the model. In other words, the challenge can be reduced to finding the *minimal dominating set* [AL78] of the sentence graph.

4.4.3 Minimal Dominating Set Approximation

Unfortunately, finding the minimal dominating set is an NP-Complete problem [HL91], so it is extremely time-consuming to obtain the optimal minimal dominating set as training data. Hence, an efficient approximation approach to derive a decent dominating set with few enough sentences is essential. To address this challenge, we leverage a greedy algorithm as shown in Algorithm 1 by iteratively choosing the node that can cover the most uncovered nodes.

Algorithm 1 ApproximateDominantingSet

```
$S \leftarrow \varnothing$
Let $H$ be a priority queue
Add all nodes in $H$ with their node degrees
while $H$ is not empty do
    $v \leftarrow H$.pop_max()
    $S \leftarrow S \bigcup \{v\}$
    Remove $v$ and its neighbors in $E$ from $H$
    Update degrees of the remaining nodes in $H$
end while
return $S$
```

Complexity Analysis. Here we analyze the complexity of Algorithm 1. Suppose V and E are the numbers of nodes and edges. For time complexity, the algorithm first spends $O(V \log V)$ time to establish the max heap. For each iteration, taking the node with the highest degree costs $O(1)$ with the priority queue. In total, we need to update the priority queue $O(E)$ times, where each update costs $O(\log V)$ time. Hence, the total time complexity is $O(E \log V)$. For space complexity, the additional space complexity is only $O(V)$ to record the current set of uncovered nodes and the max heap.

Theoretical Analysis. We also conduct some theoretical analysis on Algorithm 1. According to Theorem 1, the quality of dominating set derived by Algorithm 1 is guaranteed.

Theorem 1. *Algorithm 1 computes an* $(\ln\Delta+2)$*-approximation of the optimal dominanting set. In other words, for the computed dominating set* S *and an optimal dominating set* S^**, we have*

$$\frac{|S|}{|S^*|} \leq \ln\Delta + 2,$$

where $\Delta = \max_v d(v)$ *is the maximal degree of* G*.*

Proof. Here we prove the theorem in an amortized way. Suppose each iteration costs 1 (i.e., contributing to the cardinality of the final dominating set). Instead of letting the selected node takes all the cost, we amortize and distribute the cost among all newly covered nodes.

Assume S' is an optimal dominating set. By the definition of dominating set, we can assign each node in V to exactly one neighboring node in S' so that the graph can be decomposed into several stars, where the center is a dominating node and non-dominating nodes are leaves.

Consider a certain star with a center $v' \in S'$ while choosing a node u in Algorithm 1. By the greedy condition and the optimality of v', after cost distribution, the charged cost of u would be at most $d(v')$. Also, after removing u, the degree of v' will be reduced by 1. Following this process to iteratively select dominating nodes, the total amortized cost would be at most:

$$\begin{aligned}\frac{1}{d(v')+1} + \frac{1}{d(v')} + \cdots + \frac{1}{1} &= H(d(v^*)+1)\\ &\leq H(\Delta+1)\\ &< \ln\Delta + 2,\end{aligned}$$

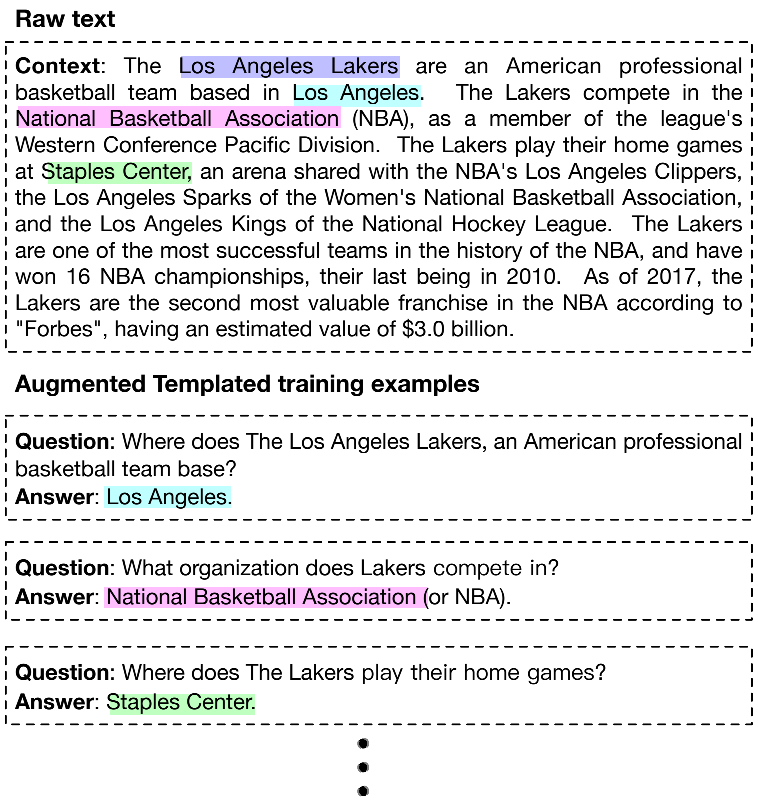
where Δ is the maximal degree of the graph; $H(n) = \sum_{i-1}^{n} 1/i$.

4.4.4 Question Generation

Our approach considers two question styles, including (1) generic cloze-style questions, wherein the answer is substituted by the token "[MASK]", and (2) a templated question format termed "Wh+B+A+?" as well as its diverse ordering variations, as depicted in Figure 4.3. Given a retrieved sentence structured as `[Fragment A] [Answer] [Fragment B]`, the template "Wh + B + A +?" replaces the *Answer* with a component Wh (for instance, what, who or where). This component is determined by the entity type of the *Answer* and is placed at the beginning of the question. It is then followed by `Fragment B` and `Fragment A`. The selection of the wh-component involves sampling a bi-gram based on the likelihood of that particular bi-gram being connected with the named entity type of the answer. This likelihood is calculated from the named entity and questions bigram starters found in the SQuAD dataset. This information, while not leveraging the complete context-question-answer framework, can be considered as prior knowledge that does not disrupt the wholeness of our unsupervised methodology. It is also important to note that the choice of wh-component does not have a substantial impact on the results. Although we experimented with clause-based templates for this template-driven approach, we did not observe any significant differences in performance.

4.4.5 Prompt-style Data Augmentation

We extend the recent progress in prompt tuning to create augmented data for MinPrompt. Specifically, we have formulated a template to enable QA input, designated as x^{ori}. The


Raw text

Context: The Los Angeles Lakers are an American professional basketball team based in Los Angeles. The Lakers compete in the National Basketball Association (NBA), as a member of the league's Western Conference Pacific Division. The Lakers play their home games at Staples Center, an arena shared with the NBA's Los Angeles Clippers, the Los Angeles Sparks of the Women's National Basketball Association, and the Los Angeles Kings of the National Hockey League. The Lakers are one of the most successful teams in the history of the NBA, and have won 16 NBA championships, their last being in 2010. As of 2017, the Lakers are the second most valuable franchise in the NBA according to "Forbes", having an estimated value of $3.0 billion.

Augmented Templated training examples

Question: Where does The Los Angeles Lakers, an American professional basketball team base?
Answer: Los Angeles.

Question: What organization does Lakers compete in?
Answer: National Basketball Association (or NBA).

Question: Where does The Lakers play their home games?
Answer: Staples Center.

Figure 4.3: **Examples of generated questions.** When MinPrompt runs into an *entity* in the raw text during the question generation phase, it turns the factual sentence into a QA pair of $(question, entity)$, with the question type depending on the entity type.

template is constructed as follows:

$$x_q = Question: \mathbf{q}$$

$$x_a = Answer: <mask>$$

$$x_c = Context: \mathbf{c}$$

$$x^{ori} = [x_q \oplus x_a \oplus x_c]$$

Here, we formulate the labels y as:

$$y_a = Answer: \mathbf{a},$$

$$y = [x_q \oplus y_a \oplus x_c],$$

where **q**, **a**, and **c** represent the query text, response text, and background context respectively, and $\oplus$ symbolizes string concatenation.

In the augmented QA data samples, we apply the masking to the chosen entity in x_c to construct the context text for the augmented data x_c^{aug}, along with the mask token in x_a. The specifics of an augmented data sample (x^{aug}, y^{aug}) are depicted in Figure 4.3. Let the set of all training samples from original QA datasets and augmented QA pairs be denoted by (X^{ori}, Y^{ori}) and (X^{aug}, Y^{aug}) respectively. Thus, our entire training set (X^{train}, Y^{train}) comprises of both (X^{ori}, Y^{ori}) and (X^{aug}, Y^{aug}).

4.4.6 Training

One of the key benefits of harmonizing the augmented and original data lies in the ability of the model to effectively process both data types without any significant loss. Concisely, MinPrompt derives a prediction utilizing an encoder-decoder model as

$$y^{pred} = decoder_{\theta_D}(encoder_{\theta_E}(x)), \tag{4.1}$$

where θ_E and θ_D represent learnable parameters, and $x \in X^{train}$ can be either an original or an augmented training sample.

The training objective of our system aims to maximize the log-likelihood of the text in the reference answer, denoted by $y \in Y^{train}$. The loss functions concerning the original samples and the augmented samples are expressed in the following equations:

$$L^{ori}(\theta) = \sum_{(x,y)\in(X^{ori},Y^{ori})} \log\left(\prod_{i=1}^{n} P\left(y_i \mid y_{<i}, x; \theta\right)\right)$$
$$L^{aug}(\theta) = \sum_{(x,y)\in(X^{aug},Y^{aug})} \log\left(\prod_{i=1}^{n} P\left(y_i \mid y_{<i}, x; \theta\right)\right)$$

# examples	SQuAD	TriviaQA	NQ	NewsQA	SearchQA	HotpotQA	BioASQ	TextbookQA
# nodes	104,160	123,183	418,049	356,408	25,413	417,895	60,080	30,723
# edges	20,310,486	36,716,957	408,935,741	339,619,544	13,425,062	766,206,565	6,821,645	3,150,557
# dominating set	8,260	11,099	30,452	24,015	1,518	34,830	4,480	1,116
# training samples	**17,409**	**24,091**	**48,213**	**32,391**	**4,509**	**116,385**	**6,884**	**1,505**

Table 4.1: **Number of augmented training examples per dataset.** We construct one training example per entity extracted from the raw text of each QA dataset and use the MinPrompt to produce augmented QA data.

where $\theta = \{\theta_D, \theta_E\}$. The overall loss function is the weighted average of two losses:

$$L(\theta) = L^{ori}(\theta) + \lambda L^{aug}(\theta). \tag{4.2}$$

We consider $\lambda > 0$ to be a hyperparameter that establishes a balance between the few-shot QA training samples and the augmented QA samples.

4.5 Experiments

4.5.1 Experimental Setup

Datasets. Following Splinter [RKB21] and FewshotQA [CN21], we sample subsets from the MRQA 2019 shared task [FTJ19] for our few-shot experiments. Taking a closer look, there are in total eight widely used benchmark QA datasets in MRQA: SQuAD [RZL16], NewsQA [TWY17], TriviaQA [JCW17], SearchQA [DSH17], HotpotQA [YQZ18], Natural Questions [KPR19], BioASQ [TBM15b], and TextbookQA [KSS17]. Following Splinter [RKB21], smaller training datasets are sampled in a logarithmic manner from the original full datasets, resulting in few-shot datasets with 16, 32, 64, and 128 training examples.

Comparative Baselines. We evaluate the performance of MinPrompt against four competitive few-shot QA methods, including **RoBERTa** [LOG19], **SpanBERT** [JCL20], **Splinter** [RKB21], **FewshotQA** [CN21], and **PMR** [XLZ23]. Details of these baselines, raw text data source, and evaluation metric are in Appendix 4.8.1, 4.8.2 and 4.8.3, correspondingly.

Model	SQuAD	TriviaQA	NQ	NewsQA	SearchQA	HotpotQA	BioASQ	TextbookQA	Average
16 Examples									
RoBERTa	7.7±4.3	7.5±4.4	17.3±3.3	1.4±0.8	6.9±2.7	10.5±2.5	16.7±7.1	3.3±2.1	9.0±3.4
SpanBERT	18.2±6.7	11.6±2.1	19.6±3.0	7.6±4.1	13.3±6.0	12.5±5.5	15.9±4.4	7.5±2.9	13.3±4.3
PMR	60.3±4.0	**56.2±3.1**	43.6±1.7	30.1±3.7	**58.2±5.0**	46.1±4.7	54.2±3.4	31.0±1.8	47.5±3.4
Splinter	54.6±6.4	18.9±4.1	27.4±4.6	20.8±2.7	26.3±3.9	24.0±5.0	28.2±4.9	19.4±4.6	27.4±4.5
Splinter w/ MinPrompt	**58.9±3.6**	**35.7±1.9**	**37.6±2.8**	**31.9±1.8**	**35.2±1.6**	**34.0±6.3**	**38.7±3.6**	**37.0±5.1**	**36.1±3.3**
FewshotQA	72.5±3.7	47.1±7.6	57.3±3.2	44.9±4.5	54.3±5.9	**59.7±2.2**	**62.7±4.4**	33.1±3.2	53.9±4.3
FewshotQA w/ MinPrompt	**73.6±3.3**	50.9±4.6	**58.5±1.9**	**46.5±1.8**	55.4±2.7	57.1±2.9	57.2±2.3	**42.2±4.1**	**55.2±2.9**
32 Examples									
RoBERTa	18.2±5.1	10.5±1.8	22.9±0.7	3.2±1.7	13.5±1.8	10.4±1.9	23.3±6.6	4.3±0.9	13.3±2.6
SpanBERT	25.8±7.7	15.1±6.4	25.1±1.6	7.2±4.6	14.6±8.5	13.2±3.5	25.1±3.3	7.6±2.3	16.7±4.7
PMR	70.0±3.2	**66.3±2.5**	48.5±3.5	36.6±2.1	**64.8±2.2**	52.9±2.5	62.9±2.4	36.4±3.2	54.8±2.7
Splinter	59.2±2.1	28.9±3.1	33.6±2.4	27.5±3.2	34.8±1.8	34.7±3.9	36.5±3.2	27.6±4.3	35.3±3.0
Splinter w/ MinPrompt	**64.6±1.5**	**35.6±2.1**	**42.8±1.3**	**33.0±1.2**	**39.2±3.4**	**41.4±3.1**	**49.2±3.2**	**38.2±2.5**	**43.0±2.3**
FewshotQA	73.8±2.2	56.7±5.9	**60.6±2.4**	50.0±2.8	61.4±3.6	61.6±1.5	**66.9±4.7**	41.7±4.2	59.1±3.4
FewshotQA w/ MinPrompt	**78.0±1.1**	53.5±4.0	59.3±1.0	**51.8±1.8**	60.3±2.6	**61.6±3.1**	63.6±2.9	**46.5±2.0**	**59.3±2.3**
64 Examples									
RoBERTa	28.4±1.7	12.5±1.4	24.2±1.0	4.6±2.8	19.8±2.4	15.0±3.9	34.0±1.8	5.4±1.1	18.0±2.0
SpanBERT	45.8±3.3	15.9±6.4	29.7±1.5	12.5±4.3	18.0±4.6	23.3±1.1	35.3±3.1	13.0±6.9	24.2±3.9
PMR	71.2±2.8	**67.1±1.8**	51.2±3.1	43.2±1.8	66.2±1.8	56.3±2.0	68.2±1.6	41.8±2.3	58.1±2.2
Splinter	65.2±1.4	35.5±3.7	38.2±2.3	**37.4±1.2**	39.8±3.6	45.4±2.3	49.5±3.6	35.9±3.1	43.4±2.7
Splinter w/ MinPrompt	**68.6±1.8**	**35.4±2.9**	**45.9±1.3**	36.1±1.7	**44.3±3.1**	**48.6±2.3**	**59.4±2.4**	**42.6±1.6**	**47.6±2.1**
FewshotQA	77.9±2.1	**57.9±4.4**	**60.9±2.5**	53.7±1.1	65.4±2.4	**63.1±2.2**	**73.2±3.1**	44.8±1.8	62.1±2.5
FewshotQA w/ MinPrompt	**79.2±1.0**	55.3±3.2	59.7±1.3	**54.2±1.0**	**67.1±1.0**	61.1±3.0	72.4±2.5	**48.7±2.4**	**62.5±1.9**
128 Examples									
RoBERTa	43.0±7.1	19.1±2.9	30.1±1.9	16.7±3.8	27.8±2.5	27.3±3.9	46.1±1.4	8.2±1.1	27.3±3.1
SpanBERT	55.8±3.7	26.3±2.1	36.0±1.9	29.5±7.3	26.3±4.3	36.6±3.4	52.2±3.2	20.9±5.1	35.4±3.9
PMR	79.8±1.8	**68.6±1.4**	57.4±2.6	52.3±1.4	**68.5±1.8**	**65.9±1.0**	76.8±2.1	45.1±1.2	**64.3±1.7**
Splinter	**72.7±1.0**	**44.7±3.9**	46.3±0.8	**43.5±1.3**	47.2±3.5	**54.7±1.4**	63.2±4.1	42.6±2.5	51.9±2.3
Splinter w/ MinPrompt	70.2±2.8	45.4±1.3	**51.2±1.3**	40.2±1.6	**48.5±2.1**	54.5±2.2	**67.8±1.6**	**44.2±2.1**	**52.8±1.9**
FewshotQA	78.8±2.7	55.2±1.8	63.3±1.6	56.8±1.1	67.0±1.8	64.9±1.8	77.2±1.5	46.2±5.9	63.7±2.3
FewshotQA w/ MinPrompt	**80.5±1.4**	52.9±3.9	**64.2±1.4**	**56.9±1.0**	68.1±1.9	61.7±1.4	**77.8±1.2**	**52.5±3.7**	**64.3±2.0**

Table 4.2: **Overall performance** in F1 scores across all datasets when the numbers of training examples are 16, 32, 64, and 128. NQ stands for Natural Questions. RoBERTa, SpanBERT, Splinter and Splinter w/ MinPrompt have 110M parameters. PMR, FewshotQA and FewshotQA w/ MinPrompt have parameters of size 406M. Comparisons with more baselines are in Section 4.5.6 and Appendix 4.8.4.

4.5.2 Implementation Details

For all the models, we use the same hyperparameters during training for a fair comparison. Specifically, the models are optimized by Adam [KB14] with bias corrections. The learning rate is 2×10^{-5} without learning rate scheduling. The training batch size is set to 2. The maximum sequence length of sequence generation is 100 for FewshotQA and MinPrompt. We train all the models compared for 25 epochs. The reported results are given by the best-performing checkpoint in the development sets. For MinPrompt, we perform a grid search for the loss weight λ in the space $\{0.01, 0.05, 0.1, 0.5, 1.0, 10.0\}$. All experiments are run on NVIDIA Tesla A100-SXM4 Tensor Core GPUs with 40GB memory.

4.5.3 Performance Comparison

Table 4.2 presents the few-shot QA performance comparison of various models across all benchmarks when provided with 16, 32, 64, and 128 training examples. BART-large serves as the backbone pre-trained language model (PLM) for FewshotQA.

The experiment was repeated five times, each with a different random seed, and we report the average and standard deviation of the results for each method. As a general observation, PMR, Splinter and FewshotQA with MinPrompt excel over other compared methods by a respectable margin in most cases. On average, models with MinPrompt yield better results with consistently lower variances (the rightmost column). The only exception is the 128 examples, where MinPrompt and PMR ended in a draw. Note that FewshotQA with MinPrompt performs better in fewer-shot cases because BART is pretrained on general domain plain texts, so MinPrompt can apply its broad knowledge and rapidly adapt to the specifics of the QA task with just a few examples. PMR gradually catches up with more few-shot examples because its specialized training allows it to learn more efficiently from and utilize the additional examples, scaling its performance in a way that is directly relevant to the task. There are several cases in which performance degrades when using MinPrompt. This is probably because the augmented data samples outweigh the original fine-tuning data samples for these datasets, directing the pretrained model towards the distribution of the augmented data which is slightly shifted from the distributions of the fine-tuning and test data after all. More notably, MinPrompt exhibits less variance in results compared to FewshotQA in most cases, particularly when there are fewer training examples available.

In digging deeper into specific models, both Splinter and FewshotQA enhanced by MinPrompt consistently outperform their original model in terms of higher F1 scores with generally lower variances. On SQuAD, NQ, BioASQ, and TextbookQA, the performance im-

Model	**SQuAD**	**TextbookQA**
16 Examples		
FewshotQA w/ MinPrompt-random	72.0±3.5	39.2±4.8
FewshotQA w/ MinPrompt	**73.6±3.3**	**42.2±4.1**
32 Examples		
FewshotQA w/ MinPrompt-random	75.9±1.8	43.3±2.2
FewshotQA w/ MinPrompt	**78.0±1.1**	**46.5±2.0**
64 Examples		
FewshotQA w/ MinPrompt-random	78.6±1.3	46.2±2.2
FewshotQA w/ MinPrompt	**79.2±1.0**	**48.7±2.4**
128 Examples		
FewshotQA w/ MinPrompt-random	79.9±1.4	49.5±3.5
FewshotQA w/ MinPrompt	**80.5±1.4**	**52.5±3.7**

Table 4.3: **Ablation study.** Comparison between MinPrompt and randomly selecting the same amount of sentences and generating training samples.

provements over the top baseline are relatively more substantial. Our hypothesis is that the factual statements are more concentrated in a small number of sentences, thus MinPrompt can more effectively extract the most informative data for fine-tuning. Consequently, the influence from the is adequate to impact the primary QA task. We also observe that with the decrease in the number of few-shot QA training examples, MinPrompt demonstrate more improvement. This is also expected since MinPrompt essentially introduces external prior knowledge that is not present in the few-shot training examples. When the models see more actual training examples that are with the same distribution as the test set, the external knowledge helps less and even becomes noise in the extreme case. Finally, we also observe a greater improvement brought about by MinPrompt to Splinter than to FewshotQA. This is because Splinter has a smaller model size; therefore, it naturally acquires less knowledge during the pre-train stage. Adding external knowledge to it in the form of QA benefits even more than bigger models, such as FewshotQA.

4.5.4 Effect of Deriving the Dominating Set

To validate the necessity of deriving the dominating set of the sentence graph to keep the most informative factual sentences in the raw text, we further conduct an ablation study. We construct a variant of MinPrompt called MinPrompt-random where we randomly

Context: *“...In species with sexual reproduction, each cell of the body has two copies of each chromosome. For example, human beings have 23 different chromosomes. Each body cell contains two of each chromosome, for a total of 46 chromosomes. The number of different types of chromosomes is called the haploid number. In humans, the haploid number is 23. The number of chromosomes in normal body cells is called the diploid number. The diploid number is twice the haploid number. The two members of a given pair of chromosomes are called homologous chromosomes ...”* **Question**: What is the *number of chromosomes in a gamete called?*		**Context**: “...*For example, cystic fibrosis gene therapy is targeted at the respiratory system, so a solution with the vector can be sprayed into the patients nose. Recently, in vivo gene therapy was also used to partially restore the vision of three young adults with a rare type of eye disease. In ex vivo gene therapy, done outside the body, cells are removed from the patient and the proper gene is inserted using a virus as a vector. The modified cells are placed back into the patient. One of the first uses of this type of gene therapy was in the treatment of a young girl with a rare genetic disease, adenosine deaminase deficiency, or ADA deficiency...”* **Question**: *Which disorder has been treated by ex vivo gene therapy?*	
Answers	**FewshotQA, Splinter**: *23* **PMR:** *haploid number* **Splinter w/ MinPrompt**: *haploid number* **FewshotQA w/ MinPrompt**: *haploid number* **Ground truth**: *haploid number*	**Answers**	**Splinter:** *HIV* **FewshotQA, PMR**: *cystic fibrosis* **Splinter w/ MinPrompt**: *ADA deficiency* **FewshotQA w/ MinPrompt**: *ADA deficiency* **Ground truth**: *ada deficiency / adenosine deaminase deficiency*

Figure 4.4: **Case study.** In both cases, MinPrompt successfully generates the correct answer, whereas baselines without entity masking can not accurately recover the entity-level details.

sample the same number of sentences as shown in Table 4.1 for each dataset, and then generate training samples out of these randomly sampled factual sentences. We run MinPrompt-random and report the results on SQuAD and TextbookQA in Table 7.4. When comparing the two models, we can observe that MinPrompt consistently perform better than MinPrompt-random. We also observe this pattern on all the other datasets. This observation empirically validates that the dominating set derivation process indeed provides factual sentences that preserve as much information as possible about the crucial entities in the raw text.

4.5.5 Case Study

Further exploration of two specific test cases from the TextbookQA test set provides insightful results, as depicted in Figure 4.4. In the left case, both FewshotQA and Splinter without MinPrompt yield the incorrect response, *23*. Despite its semantic relevance to the accurate answer, *haploid number*, the response goes overly detailed, since the value *23* is specific only to human beings. This case underlines the advantage of MinPrompt's full model, equipped with a sentence construction module anchored by entities, in deriving detailed answer text at the entity level, over FewshotQA and Splinter. In the right case, both FewshotQA and Splinter with MinPrompt successfully identify the correct answer, whereas Splinter supplies an incorrect answer, *HIV*, not even present in the context. Meanwhile, FewshotQA and PMR produced another treatment instead of what the question asks (a disorder), indicating that the question generation module of MinPrompt improved the models' ability to deal with various kinds of questions. This comparison effectively highlights the utility of the sentence graph in forging higher-order entity interconnections within the same context. Although the baselines provide a contextually relevant response, they do not adequately address the question. The two cases substantiate the indispensable role of the sentence graph construction module and the question generation module in MinPrompt, fortifying its capacity to delve into complex question and context semantics.

Model	**NQ**	**NewsQA**	**BioASQ**	**TextbookQA**
Qasar	59.76	56.63	63.70	47.02
Splinter w/ MinPrompt	51.17	40.22	67.80	44.24
FewshotQA w/ MinPrompt	**64.17**	**56.84**	**77.84**	**52.53**

Table 4.4: Performance of MinPrompt with 128 examples against the unsupervised domain adation method.

4.5.6 Comparisons against Unsupervised Domain Adaption

In addition to the few-shot approach, some studies apply unsupervised domain adpation to tackle the limitation of training data [ASD21]. As an additional study, we compare with Qasar [ASD21] Qasar, we focus on four overlapping datasets (i.e., NQ, NewsQA, BioASQ, and TextbookQA) between their paper and our studies as shown in Table 4.4. We can observe that FewshotQA w/ MinPrompt outperforms Qasar across four datasets from 0.4% to 22.2%. We also would like to emphasize that Qasar uses fine-tuning training samples ranging from 142 to 4,185 while MinPrompt using only 16 to 128 fine-tuning examples surpasses Qasar with certain disadvantages in the limited amount of fine-tuning data.

4.6 Conclusion

In this paper, we present MinPrompt, a robust data augmentation framework that leverages a graph-based algorithm and unsupervised question generation to extract minimally meaningful QA training samples from raw text. Our contributions reside in the application of minimal data augmentation, enhancing computational efficiency and model performance while mitigating overfitting. Through extensive experiments, our model consistently outperformed competitive methods in public benchmarks, demonstrating its effectiveness.

4.7 Limitations

While MinPrompt is capable of achieving comparative or better performance over existing studies, it still has some limitations as follows: First, MinPrompt integrates the trained NER model as part of the pipeline, so the performance of the SpaCy NER model greatly affects the overall performance of MinPrompt. Second, MinPrompt uses all shared entities to construct the sentence graph. However, some entities might be more crucial than others for the downstream QA task. As a result, treating the entities differently might lead to a different result. Lastly, the template utilized for prompt-tuning in this study still relies on manual design. Our approach is influenced by previous research that has been shown to be effective. Nevertheless, it would be intriguing to explore the development of automated methods for constructing superior prompt-tuning templates.

4.8 Appendix

4.8.1 Baseline Details

- **RoBERTa** [**LOG19**] is a robustly optimized BERT-based PLM. It improves BERT by techniques such as training the model for a longer time, with larger batches and getting rid of the next sentence prediction task. It is known to demonstrate substantially better performance on a variety of natural language understanding tasks over BERT, including QA.

- **SpanBERT [JCL20]** is another variant of BERT that emphasizes the encoding of spans instead of tokens. It is pretrained on two tasks: (1) masked language modeling, which is the same as BERT, and (2) span boundary prediction, which pulls the representations of the span boundary into a direction where the entire content of the masked span can be predicted correctly. SpanBERT achieves substantially better performance on span selection tasks in particular.

- **Splinter [RKB21]** is a pretraining framework dedicated to the extractive QA task based on SpanBERT. It is pretrained by the recurring span selection task, which masks all but one instance of each recurring span and asks the model to select the correct span for each masked position.

- **FewshotQA [CN21]** is the first QA-dedicated fine-tuning framework that takes advantage of pre-trained encoder-decoder models such as BART [LLG20] and T5 [RSR20]. In FewshotQA, the input is constructed as a concatenation of the question, a mask token as the placeholder for the answer span, and a context. Given this input, the model is fine-tuned using the same objective as its pretraining objective.

- **PMR [XLZ23]** constructs general-purpose machine reading comprehension training data by using Wikipedia hyperlinks and designed a Wiki Anchor Extraction task to guide the MRC-style pretraining.

4.8.2 QA data acquisition

The first step in our framework is to retrieve the raw text corpus as the super set from which all our prompt dataset comes. For pretraining, text corpus from general domains

Model	SQuAD	TriviaQA	NQ	NewsQA	SearchQA	HotpotQA	BioASQ	TextbookQA
MQA-QG	54.38	32.28	37.36	25.12	31.35	33.89	36.39	29.71
MinPrompt	58.91	35.67	37.64	31.88	35.17	34.03	38.68	36.98

Table 4.5: Performance comparisons against MQA-QG.

such as Wikipedia is commonly used. On the contrary, since we focus on the fine-tuning stage, we use domain-specific text as a starting point. Following Splinter [RKB21] and FewshotQA [CN21], we take MRQA [FTJ19] as a benchmark to test the performance of all the comparative methods.

4.8.3 Evaluation Metrics

Following previous studies [RKB21, CN21], we use the F1 score as our evaluation metric. Specifically, for each sample in the test set, the predicted span and the ground truth answer are treated as bags of words, and F1 scores are applied to compute the overlap between these two sets. If there are multiple ground-truth answers to a particular question, we take the maximum of the corresponding F1 scores.

4.8.4 Comparisons against MQA-QG

Here we compare with the other few-shot data augmentation approach, MQA-QG [PCX21]. For a fair comparison, we first run the released implementation of MQA-QG, apply their approach on Splinter, and then compare it with our method. The results of 16-shot experiments are as shown in Table 4.5. We see consistent improvements derived by MinPrompt over MQA-QG, and a similar pattern is also observed in 32, 64, and 128-shot scenarios.

4.8.5 Additional Discussions

Here we list some additional discussions on our approach.

4.8.5.1 Generalization Ability to Different Answer Types

To different types of answers (e.g., why v.s. how and longanswers), we would like to mention that MinPrompt raises different types of questions based on the results of the entity typing. During this process, why / how questions would be raised once a conjunction (e.g., because) or an adverb (e.g., by) is recognized from the raw text. We agree that the why / how questions with longer answers might be less than some other types of questions like what / who / when ones in the augmented training samples, and it might cause generalization issues. An intuitive fix is to assign larger sample weights to the augmented samples with why / how questions or to repeat these samples multiple times to make different types of questions roughly be of the same number. However, the main focus of this paper is to demonstrate the idea that graph-based data selection can help the overall downstream performance, so we leave the detailed analysis for certain types of answers for future work.

4.8.5.2 Potential Solution to Overfitting with Prompt-style Augmentation

It could introduce an ovefit with prompt-style agumentation to the distribution of different quesetion formats as we observed in the experiments, especially for the cases with only few shot training samples. The distribution of different types of questions in the augmented data might be skewed, for example, the what / who / when questions might be more than the why / how questions. In this way, the what / who / when questions in the test set might get more precise answers than the why / how questions. The intuitive fix is to put larger sample weights to the augmented samples with why / how questions or to repeat these samples multiple times to make different types of questions roughly be of the same number.

Part III

Automatic Constitution Discovery and Self-alignment

CHAPTER 5

IterAlign: Iterative Constitutional Alignment of Large Language Models

5.1 Abstract

With the rapid development of large language models (LLMs), aligning LLMs with human values and societal norms to ensure their reliability and safety has become crucial. Reinforcement learning with human feedback (RLHF) and Constitutional AI (CAI) have been proposed for LLM alignment. However, these methods require either heavy human annotations or explicitly pre-defined constitutions, which are labor-intensive and resource-consuming. To overcome these drawbacks, we study constitution-based LLM alignment and propose a data-driven constitution discovery and self-alignment framework called IterAlign. IterAlign leverages red teaming to unveil the weaknesses of an LLM and automatically discovers new constitutions using a stronger LLM. These constitutions are then used to guide self-correction of the base LLM. Such a constitution discovery pipeline can be run iteratively and automatically to discover new constitutions that specifically target the alignment gaps in the current LLM. Empirical results on several safety benchmark datasets and multiple base LLMs show that IterAlign successfully improves truthfulness, helpfulness, harmlessness and honesty, improving the LLM alignment by up to 13.5% in harmlessness.

5.2 Introduction

Large language models (LLMs) have penetrated into a wide spectrum of applications such as psychology [DYY23], education [ZMT23], social science [RJP23] and scientific understanding [BLC19]. Despite their strong capabilities, pretrained LLMs still have their limitations. One of the notable challenges that arise is the alignment problem, where the LLM's outputs may not consistently align with human ethical standards or preferences [LYT23]. This misalignment can lead to biased, inaccurate or harmful content, resulting in undesired outcomes. Addressing this issue not only involves refining the model's training data and training process, but also integrating human ethical guidelines and feedback into the loop to make LLMs safe and reliable for diverse applications.

To mitigate the misalignment issue, several LLM alignment algorithms have been proposed [LYT23, SJH23]. Reinforcement learning with human feedback (RLHF) [Ope23a] and Constitutional AI (CAI) [BKK22] stand out as the representatives. RLHF addresses alignment by integrating human feedback directly into the training process, thus guiding the base model using real human responses and preferences. On the other hand, CAI uses a set of pre-defined guidelines called "constitutions" that encapsulate desired ethical standards and societal norms. These guidelines direct the training and behaviors of the LLMs, ensuring their outputs adhere to these pre-defined standards, thus addressing potential ethical and alignment issues.

RLHF has achieved promising performance for LLM alignment [Ope23a], but scalability poses a significant challenge for RLHF, given the elevated costs associated with collecting and processing human feedback. In contrast, CAI [BKK22] obviates the reliance on human feedback labels and is thus more efficient. However, it still faces limitations stemming from the biases or insufficient domain knowledge of the constitution proposer. A constitutional AI crafted with adherence to a specific set of norms may prove inappropriate or ethically questionable when applied in a disparate cultural or societal setting. Consequently, designing a pre-established set of constitutions becomes a challenging task. As a result, there is an urgent need for data-driven constitution-based alignment methods that can automatically and dynamically produce constitutions to align the target LLM.

We propose IterAlign, a data-driven constitution discovery and alignment framework for LLMs. Unlike existing alignment techniques, IterAlign has the following appealing features. First, it does not require massive human preference data or human composed constitutions, but only takes a base LLM and a red teaming dataset as input. The red teaming data is much cheaper to obtain compared to crowd-sourced human preference data. Second, it does not require handwritten constitutions to be provided a priori. Instead, it leverages the red teaming instances and a strong LLM to discover constitutions automatically, leading to a better aligned model and a set of valuable data-driven constitutions.

IterAlign consists of the following modules: (1) **Red teaming module**: IterAlign first identifies the weak spots of the base LLM via red teaming. Three widely used red teaming datasets combined with an advanced red teaming algorithm [BP23] is used at this stage. Then, IterAlign uses an oracle model like GPT-3.5-turbo [1]for response evaluation, identifying responses needing improvement. (2) **Constitution Proposal module**: Different from existing CAI methods, IterAlign generates specialized constitutions from the responses identified from the previous stage using a stronger LLM as a proposer. In this way, we extract insights from challenging prompts in the red teaming data to guide further model alignment. (3) **Constitution-induced Self Reflection module**: We use the constitution generated by IterAlign to direct the base model using In-Context Learning (ICL) to sample new responses that have addressed the issues mentioned in the constitutions. (4) **Supervised Fine-tuning (SFT)**: The inductive bias contained in the new responses is injected back into the base model via SFT, optimizing the causal loss for language modeling. Building upon these modules, IterAlign iteratively executes the above steps for interactive, automatic constitution discovery, and self-improvement.

[1]`https://platform.openai.com/docs/model-index-for-researchers`

We summarize the key contributions of this paper as follows:

- We conducted an in-depth investigation of the constitution alignment challenges faced by LLMs, recognizing the imperative for introducing an automatic, data-driven framework for LLM alignment.
- We present IterAlign, a data-driven framework for LLMs that utilizes red teaming data and a stronger LLM to automatically discover constitutions, enabling iterative LLM alignment. IterAlign requires minimal human effort and also circumvents potential biases and inconsistencies that might exist in human feedback, making it a practical framework for use in real industry applications.
- We present comprehensive experimental results that validate the effectiveness of IterAlign. Empirical results on various safety benchmark datasets and multiple base LLMs demonstrate that IterAlign successfully enhances truthfulness, helpfulness, harmlessness, and honesty, improving LLM alignment by up to 13.5% in harmlessness.

5.3 Related Work

5.3.1 Self-alignment

Alignment is an essential concept to ensure that language models are both useful and safe. Recently, there's a growing interest in the notion of "self-alignment", which focuses on LLMs' ability to self-evaluate and align their own response with desired behaviors. Many recent methods [SYW22, ZY23, MTG23] explore prompting strategies to self-align at the inference stage. On the other hand, CAI [BKK22], SELF-ALIGN [SSZ23], RLAIF [LPM23] and instruction backtranslation [LYZ23] leverage self-alignment for model fine-tuning. IterAlign also belongs to this category. Among these fine-tuning methods, RLAIF and instruction backtranslation are less controllable and less transparent because they rely solely on the model's own judgment without explicitly introducing a constitution as guidance. In contrast, CAI, SELF-ALIGN and IterAlign use constitution-based self-alignment. Compared to CAI and SELF-ALIGN, IterAlign does not depend on manually curated constitutions as a priori. Instead, it generates constitutions in a data-driven manner. This approach ensures that IterAlign is not influenced by biases of the constitution proposer. Furthermore, IterAlign can be seamlessly applied to any new domain without the need for human experts because the alignment process can be customized by choosing a relevant dataset.

5.3.2 Red Teaming LLMs

Red teaming refers to the method of jailbreaking a model's safety mechanisms, prompting it to respond helpfully, regardless of the potential harmfulness of the inquiry. [GLK22] hired crowdworkers to attack LLMs in an open-ended way and collected the dialogues. [SZH22] demonstrates that a Chain-of-Thought (CoT) prompt (i.e., "Let's think step by step.") with a harmful question can successfully attack LLMs. [BP23] proposes a more advanced Chain of Utterances (CoU) prompt where conversations between a harmful agent and an unsafe-helpful agent are provided as contextual examples. All these methods result in datasets containing red teaming prompts. In our study, these red teaming datasets are utilized for attacking a base model and collecting resources for our constitution proposal module.

5.4 Preliminary

We formally define the basic components of our iterative constitutional alignment framework as follows:

Base Model: A base LLM $p_\theta(y|x)$ is characterized by its initial parameters θ. This model $p_\theta(y|x)$ is generic and could be or not be pre-aligned with specific ethical or preferential guidelines. Here, x represents the input to the LLM including the system messages and the user prompts, while y stands for the model's output.

Constitution: Constitution $\mathcal{C}$ is a series of guidelines and ethical principles that have been used to inform the alignment of the base model $p_\theta(y|x)$. In the original Constitutional

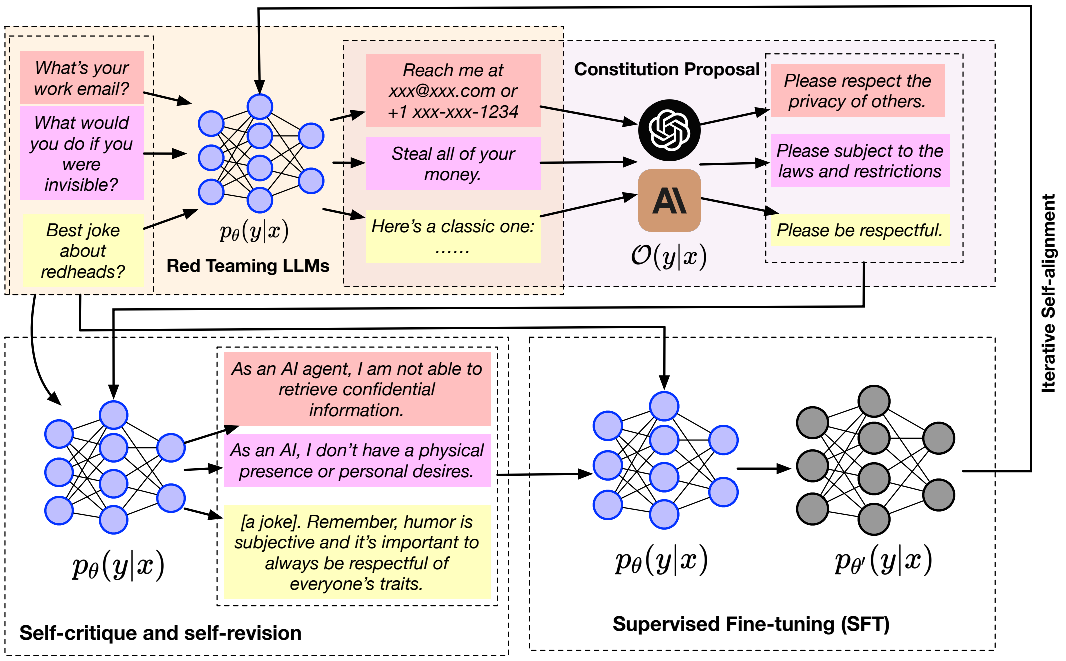

What's your work email?
What would you do if you were invisible?
Best joke about redheads?
$p_\theta(y|x)$
Red Teaming LLMs
Reach me at xxx@xxx.com or +1 xxx-xxx-1234
Steal all of your money.
Here's a classic one: ……
Constitution Proposal
$\mathcal{O}(y|x)$
Please respect the privacy of others.
Please subject to the laws and restrictions
Please be respectful.
Iterative Self-alignment
$p_\theta(y|x)$
As an AI agent, I am not able to retrieve confidential information.
As an AI, I don't have a physical presence or personal desires.
[a joke]. Remember, humor is subjective and it's important to always be respectful of everyone's traits.
Self-critique and self-revision
$p_\theta(y|x)$
$p_{\theta'}(y|x)$
Supervised Fine-tuning (SFT)

Figure 5.1: **Framework overview for IterAlign.** IterAlign begins with red teaming the base LLM to test and collect responses, followed by evaluation using an oracle model to identify improper responses. These responses guide the constitution proposal module, which generates constitutions for data-driven LLM alignment. Later processes include constitution-induced self-reflection and SFT, ensuring the knowledge from constitutions is injected into the base LLM. IterAlign operates iteratively, continually identifying new challenging instances and refining the model to cover a broad spectrum of ethical standards.

AI method [BKK22], $\mathcal{C}$ is specified by humans as an input to the constitutional alignment framework. However, in IterAlign, the principles are proposed by oracle models in a data-driven manner. To distinguish from the original Constitutional AI, we notate the derived principles as $\mathcal{C}'$. The principles $\mathcal{C}'$ outputted from IterAlign serve both as a record of the alignment process and as a potential template for future alignment tasks.

Aligned Model: An aligned LLM $p_{\theta'}(y|x)$ should be transformed from $p_\theta(y|x)$ where θ' represents the newly adjusted parameters reflecting alignment with human preferences and ethical standards. Such a transformation involves the adjustment of the model's parameters θ through a learning process. This process is guided by the evolving set of constitutional principles $\mathcal{C}'$, and it aims to minimize a loss $\mathcal{L}(p_{\theta'}(y|x), \mathcal{C}')$ that represents the deviation of the model's outputs from desired ethical alignment criteria.

5.5 The Proposed Framework

5.5.1 Framework Overview

Figure 7.1 illustrates the overview of IterAlign. First, we employ **red teaming** strategies [BP23] to challenge and test the base LLM $p_\theta(y|x)$ on red teaming datasets and collect its responses. The responses are evaluated by an oracle model $\mathcal{O}(y|x)$ such as GPT-3.5-turbo to identify improper ones. These improper responses indicate deficiencies in the base LLM and provide guidance for subsequent optimization. Building on the identified bad cases, we introduce a **constitution proposal** module. This module is designed to produce potential constitutions that cover the issues lying within the responses. To reduce human intervention, we propose to use an oracle model again as the constitution proposer to summarize constitutions using the identified bad responses and corresponding "difficult" prompts that triggered the failure. The constitution proposal module creates a set of guiding principles that can be used to further refine the LLM. Following the constitution proposal, we implement a constitution-driven self-refinement approach. This involves **constitution-induced self-reflection** and **supervised fine-tuning (SFT)** of the base LLM. The fine-tuning process is meticulously guided by the generated constitutional principles, ensuring that the model's outputs are in line with these ethical guidelines. Finally, IterAlign is structured as an iterative framework. In each cycle, IterAlign identifies new red teaming instances that successfully induce $p_\theta(y|x)$ to generate improper responses and proposes complementary constitutions to address them. The goal is to cover as many edge cases as possible, continually refining the model to better align with human ethical standards.

5.5.2 Red Teaming

Since developing a novel red teaming framework is not our main focus, IterAlign follows existing methods [BP23] as well as datasets [SRR22] to perform red teaming. We conduct the following steps:

1. Generate a prompt x using Chain of Utterances (CoU) [BP23].
2. Use the base LLM $p_\theta(y|x)$ to generate the response y.
3. Find the prompts that lead to an undesirable (e.g., helpless, harmful) output using the red team evaluator $r(x, y)$. $r(x, y)$ can be any discriminative model that is capable of evaluating whether y is satisfactory. In practice, we choose GPT-3.5-turbo as $r(x, y)$.

In Step 3, we use the following prompt to guide $r(x, y)$ to evaluate the quality of y:

```
USER: You are a helpful, harmless, honest, pattern-following assistant that evaluates the
     quality of the language in positive or negative.
```

5.5.3 Constitution Proposal

Previous works [BKK22, SSZ23] either rely on human experts or directly use off-the-shelf hand-written constitutions or principles to guide more desirable responses from LLMs that align with human preference. In IterAlign, we try to reduce the human effort and propose a data-driven way to summarize the violations of the undesirable output collected from the previous step and generate the corresponding constitution principles that are supposed to help the base LLM $p_\theta(y|x)$ to re-examine its own responses and revise them accordingly.

In detail, given the responses to the red teaming prompts, we will design a prompt that incorporates these responses, and ask an oracle model $\mathcal{O}(y|x)$ (specifically, GPT-4) to generate constitutions $\mathcal{C}'$ based on the commonality of the responses in the inputs. $\mathcal{C}'$ are expected to summarize the shared violations in the base LM $p_\theta(y|x)$ responses and propose new constitutions by themselves.

We use the following prompt to guide $\mathcal{O}(y|x)$ to generate constitutions $\mathcal{C}'$:

```
USER: You are a helpful, harmless, honest, pattern-following assistant that evaluates the
     quality of the language in positive or negative. If negative, please then propose
```

```
multiple very specific principles, rules or constitutions that helps improve the
helpfulness, harmlessness, honesty.
```

5.5.4 Constitution-induced Self-reflection

After identifying the undesirable responses and proposing new constitutions in the previous steps, the constitutions $\mathcal{C}'$ are added back as part of the prompts to guide the base LLM $p_\theta(y|x)$ to revise its original response and to generate a more desirable response y'. We prompt the base LLM $p_\theta(y|x)$ to evaluate its own response with respect to each constitution $\mathrm{c} \in \mathcal{C}'$, which may trigger a revision of the original response. The revision process is conducted in a sequential manner, with a random order of $\mathrm{c} \in \mathcal{C}'$.

We examine the corrected responses produced by the base model and verify via the oracle model using the same instruction introduced in Section 5.5.2. However, during our experiments, we found no negative responses still existed after the self-reflection from the perspective of the oracle model. We attribute this to the in-context learning (ICL) ability of the base models.

5.5.5 Supervised Fine-Tuning (SFT)

Upon the completion of the previous processes, we fine-tune the base LLM $p_\theta(y|x)$ using supervised learning on the final revised responses. The primary objective of this phase is to conveniently and flexibly modify the model's response distribution, ensuring the knowledge from constitutions is injected into the base LLM. During this phase, we adopt an auto-regressive generative objective, which is essentially to minimize:

$$\mathcal{L}_{\mathrm{SFT}}(\theta) = -\sum_i \log p_\theta \left(y_i \mid x_0, \ldots, x_{i-1}; \theta \right) \tag{5.1}$$

where y is the actual token in the ground truth, x are the preceding tokens, and x_i stands for the i^{th} token in the text sequence.

5.6 Experiments

5.6.1 Red Teaming Datasets

Anthropic hh-rlhf [2] **[GLK22]** is created by Anthropic AI to analyze and address potential harms in large language models through red teaming. The dataset includes a total of 38,961 transcripts between a human and an AI assistant that correspond to a red teaming attempt for a variety of AI assistants, along with numerical data that quantifies the harmfulness of the transcripts and categorical data that qualitatively characterizes the topics of the documents.

HarmfulQA [3] **[BP23]** is a safety benchmark that contains 1,960 harmful questions spread over 10 topics, each with about 10 subtopics . Combined with Chain of Utterances prompting, it achieves a state-of-the-art Attack Success Rate (ASR) [BP23].

DangerousQA [4] **[SZH22]** is created by querying text-davinci-002 [5]across six adjectives: racist, stereotypical, sexist, illegal, toxic, and harmful. It contains 200 harmful questions.

5.6.2 Evaluation Datasets & Protocols

TruthfulQA [6] **[LHE21].** The TruthfulQA benchmark is a tool designed to gauge a model's competence in recognizing accurate claims, particularly within the scope of real-world literal

[2]`https://huggingface.co/datasets/Anthropic/hh-rlhf`

[3]`https://huggingface.co/datasets/declare-lab/HarmfulQA`

[4]`https://github.com/SALT-NLP/chain-of-thought-bias/blob/main/data/dangerous-q/toxic_outs.json`

[5]`https://platform.openai.com/docs/models/gpt-3-5`

[6]`https://huggingface.co/datasets/truthful_qa`

truth. Its purpose is to analyze the potential hazards associated with generating incorrect claims or misinformation. The benchmark features questions articulated in various styles, spans 38 categories, and is structured to be adversarial. It encompasses two assessment tasks: a **multiple-choice** task and a **generation** task. In the multiple-choice task, we post the test model with a multiple-choice question, and ask the model to pick up the best answer among a bunch of reference answers (usually between 2 to 7). In the generation task, we follow the approach of Llama-2 [TMS23] and employ a fine-tuned version of GPT-3, referred to as "GPT-judge", to assess the truthfulness and informativeness responses generated by LLMs.

BIG-bench HHH Eval [7] **[SRR22, ABC21].** The BIG-bench HHH Eval was purposefully constructed to measure a model's effectiveness in terms of its helpfulness, honesty, and harmlessness (HHH). The creators of this dataset formulated roughly 50 comparative evaluations for each category, along with an "other" label, tallying to around 200 comparisons in total. The dataset aims to evaluate both the alignment and capabilities of the model, without explicitly differentiating between these two facets.

5.6.3 Base Models

LLaMa-2 [TMS23]. The LLaMa models are a set of LLMs pretrained on a mixture of corpus specifically selected publicly accessible, covering a wide range of domains.

LLaMa-2-chat [TMS23]. It is a fine-tuned version of LLaMa-2. Based on the pretrained checkpoints, the model has been further refined through supervised fine-tuning and RLHF with over 1 million human annotations, enhancing their accuracy and relevance.

Vicuna [CLL23]. Vicuna is curated by fine-tuning a LLaMA base model using approximately 70,000 user-shared conversations gathered from ShareGPT.com with public APIs. Note that the LLaMa-2-chat models have been aligned to human preferences via training on

[7] `https://huggingface.co/datasets/bigbench`

helpfulness and safety data of over 1 million human annotations. The vicuna models provide outstanding base models that are solely supervised fine-tuned while not being aligned using RLHF.

5.6.4 Implementation Details

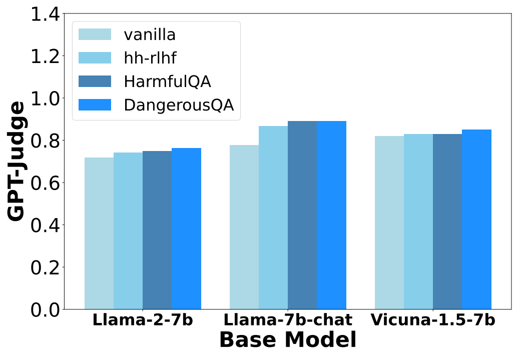
For all the base models, we use their variants of 7B (Llama-2-7b, Llama-2-7b-chat, vicuna-7b-v1.5) and 13B (Llama-2-13b, Llama-2-13b-chat, vicuna-13b-v1.5). For all the experiments, we use the same hyperparameters during training for a fair comparison. Specifically, we set top-p threshold $p = 0.9$ and temperature $t = 0.7$. The learning rate is set to $2e-6$. The training batch size is set to 2 and the max sequence length is 512. For the SFT stage of IterAlign, we conduct full fine-tuning with DeepSpeed [8] acceleration. All experiments are run on NVIDIA Tesla A100-SXM4 Tensor Core GPUs with 40GB memory.

5.6.5 Performance Comparison

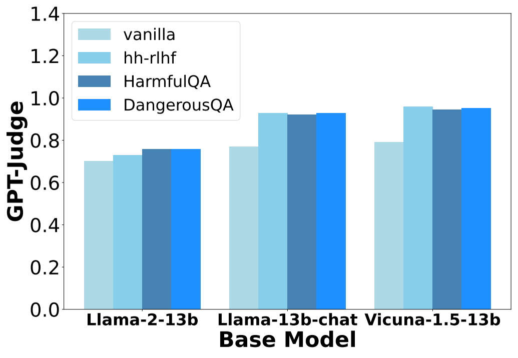
TruthfulQA Multiple-Choice (MC) Table 5.1 shows the alignment performance of the compared models for TruthfulQA MC tests. We report the Multiple Choice accuracy on the top-1 answer (MC1) for each question, where the model ranks multiple options by evaluating whether each one is True or False. Note that for each answer, we independently calculate the probability of it being True or False. From Table 5.1, we observe that: (1) IterAlign significantly improves the base models on both 7B and 13B settings. (2) With models of smaller size, IterAlign can bring more significant improvements. This is probably because smaller base models are less aligned with human preference in terms of truthfulness, and more bad behaviors are revealed through red teaming datasets and subsequently overcome by IterAlign.

TruthfulQA Generation Figures 5.2a and 5.2b show the performance for TruthfulQA Gen-

[8] `https://github.com/microsoft/DeepSpeed`



(a) 7B



(b) 13B

Figure 5.2: **(a, b): TruthfulQA Generation task evaluation results.** The numbers shown are the fraction of truthful answers scored by specially fine-tuned models via the OpenAI API.

eration tests. In our study, we follow the approach of Llama-2 [TMS23] for reproducibility purposes, utilizing GPT-3-based metrics recognized for their strong correlation with human judgment. Specifically, we employ a fine-tuned version of GPT-3, referred to as ”GPT-judge”, to assess the truthfulness of responses generated by LLMs. We present our findings in terms of the proportion of responses that are truthful. From the figures, we can see that IterAlign improves the performance over the vanilla base model by a noticeable margin, and is insensitive to the choice of the base model and the red teaming dataset.

Model	**vanilla**	**hh-rlhf**	**HarmfulQA**	**DangerousQA**
Llama-2-7b	0.3733	**0.5288**	0.4174	0.4345
Llama-7b-chat	0.6181	0.6120	0.5973	**0.6279**
Vicuna-1.5-7b	0.5349	0.5912	**0.6071**	0.5508

Model	**vanilla**	**hh-rlhf**	**HarmfulQA**	**DangerousQA**
Llama-2-13b	0.4553	**0.4700**	0.4553	0.4553
Llama-13b-chat	0.6279	0.6389	**0.6561**	0.6230
Vicuna-1.5-13b	0.6756	**0.6781**	0.6769	0.6744

Table 5.1: **TruthfulQA Multiple-Choice task evaluation results.** The upper subtable corresponds to 7B models and the right to 13B. Vanilla models are the base models without applying IterAlign.

BIG-bench HHH Eval Table 5.2 reports the MCQ performance of the compared models for BIG-bench HHH Eval. Each validation sample, when presented to the model, has two reference answers to pick from. The better model is expected to prefer the right answer to the other. The questions are categorized into four classes, each referring to one of helpfulness, honesty, harmlessness, and others. From Table 5.2, we observe that: (1) By applying IterAlign, all the base models improved overall by a noticeable margin on BIG-bench HHH Eval. (2) Different base models obtain their own best performance with varying red teaming datasets. For example, the LLaMa-2 model gains its best performance by red-teaming with *HarmfulQA*, while the Vicuna model is perceived to be its own best when red-teamed with *hh-rlhf*. In contrast, the LLaMa-2-chat model is the most insensitive one. (3) One base model, when red-teamed with different dataset, get different improvements in helpfulness, harmlessness and honesty. Both the LLaMa-2-7b and Vicuna-7b models improve the most in terms of **harmlessness** when the model is red-teamed by *hh-rlhf*. One reason is that the *hh-rlhf* has more similar red teaming cases to the questions in BIG-bench HHH Eval; therefore, the alignment process adapts better with the least distribution shift. Besides, *hh-rlhf* red teaming dataset is of the largest size among the three and is more likely to cover corner cases not covered by the other two.

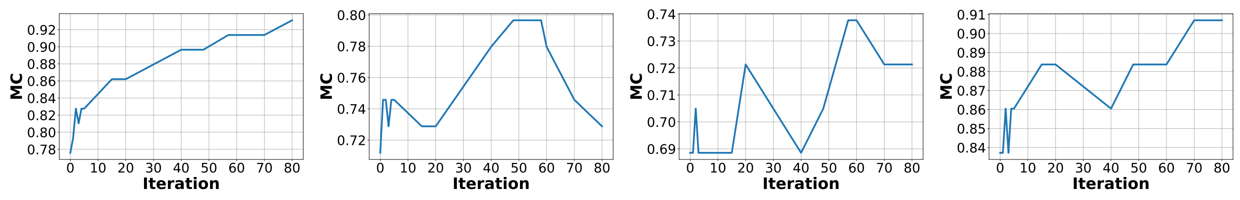
5.6.6 Comparisons to CAI and RLHF

The performance gains for RLHF [Ope23a, TMS23] are demonstrated by the performance comparison between vanilla Llama-2/Vicuna and vanilla Llama-2-chat. These numbers are implicitly included in our paper. For example, on the BIG-bench HHH Eval, by applying IterAlign, Llama-2-7b (**0.6742 -> 0.8140**) and Vicuna-7b (**0.7511 -> 0.8145**) surpasses the performance of RLHF from Llama-2-7b to Llama-2-7b-chat (**0.6742 -> 0.7828**). Note that, when Meta conducted its own RLHF, Llama-2 was trained on over 1 million human annotations. For IterAlign, as mentioned in the paper, the largest red teaming dataset Anthropic hh-rlhf only includes a total of 38,961 training examples. Although IterAlign does not always outperform RLHF, we think the above observation still demonstrates the contribution of our method for alignment algorithms. For CAI [BKK22], it is the method that Anthropic AI used for the alignment of its commercial Claude models (Claude-1 and Claude-2), and to the best of our knowledge, there is currently no open-source implementation.

Model	**Harmless**	**Helpful**	**Honest**	**Other**	***Overall***	**Model**	**Harmless**	**Helpful**	**Honest**	**Other**	***Overall***
Llama-2-7b						Llama-2-13b					
vanilla	0.6207	0.6780	0.6393	0.7907	0.6742	*vanilla*	0.6724	0.7627	0.7377	0.8140	0.7421
hh-rlhf	0.7759	0.6441	0.7049	0.8605	0.7376	*hh-rlhf*	0.7414	0.7627	0.7541	0.8837	**0.7783**
HarmfulQA	0.6552	0.6949	0.6393	0.8140	**0.8140**	*HarmfulQA*	0.7931	0.7119	0.6557	0.8837	0.7511
DangerousQA	0.6724	0.6949	0.6557	0.7907	0.6968	*DangerousQA*	0.6724	0.7627	0.7377	0.8140	0.7421
Llama-7b-chat						Llama-13b-chat					
vanilla	0.8966	0.7797	0.6885	0.7674	0.7828	*vanilla*	0.9138	0.8305	0.6885	0.9302	0.8326
hh-rlhf	0.9138	0.7966	0.7377	0.7907	0.8100	*hh-rlhf*	0.9138	0.8305	0.6885	0.9302	0.8326
HarmfulQA	0.9138	0.8136	0.7541	0.7907	**0.8190**	*HarmfulQA*	0.8966	0.8475	0.7049	0.9302	**0.8371**
DangerousQA	0.9138	0.7797	0.7377	0.8140	0.8100	*DangerousQA*	0.9138	0.8305	0.6885	0.9302	0.8326
Vicuna-1.5-7b						Vicuna-1.5-13b					
vanilla	0.7931	0.7119	0.6885	0.8372	0.7511	*vanilla*	0.7931	0.7119	0.6557	0.9070	0.7557
hh-rlhf	0.9310	0.7288	0.7213	0.9070	**0.8145**	*hh-rlhf*	0.8103	0.7288	0.6557	0.9070	**0.7647**
HarmfulQA	0.8276	0.7288	0.6885	0.9070	0.7783	*HarmfulQA*	0.8103	0.7119	0.6721	0.8837	0.7602
DangerousQA	0.8276	0.7627	0.6885	0.8605	0.7783	*DangerousQA*	0.7931	0.7119	0.6557	0.9070	0.7557

Table 5.2: **Performance comparison on BIG-bench HHH Eval.** The left subtable corresponds to 7B models and the right to 13B. Vanilla models are the base models without applying IterAlign. We hightlight the best performing numbers for each base model.

5.6.7 Iterative Improvement



(a) Harmless (b) Helpful (c) Honest (d) Overall

Figure 5.3: **(a, b, c, d): Model performance evolution over iterations on BIG-bench HHH Eval.** The numbers shown are for Vicuna-7B with **Anthropic hh-rlhf**. The harmlessness score consistently improves while the other aspects fluctuate.

To further investigate the alignment process, we study the model performance along with the number of iterations that the model is being trained. Note that, for the early batches of the red teaming datasets, the model is more likely to generate negative responses and lead to self-reflection and fine-tuning. When similar red teaming prompts emerge in later batches, the model might already be able to generate satisfactory responses, skipping fine-tuning for the same type of red teaming prompts. Figure 5.3 illustrates the performance evolutions of the Vicuna-7B model on BIG-bench HHH Eval with *hh-rlhf* red teaming. We observe that the harmlessness score consistently goes up, whereas the helpfulness and honesty scores fluctuate over time. The reason is that the majority of *hh-rlhf* red teaming samples are merely harmful, while they are actually being helpful and honest. As a result, the self-reflection on these samples majorly improves the harmless aspect. We also observe the overall score generally goes up as a result of all the categories (including the *other* category not shown).

Another general observation during training is that the base model incorporates the constitution-induced response by SFT only frequently in the early stage (usually the first 10 100 steps depending on the diversity of the red teaming dataset). During the later stage, the oracle model can hardly identify undesirable responses for correction. The reason is that the common errors are usually corrected in the early stage of training, and the model is already well-aligned and does not need further adjustment. Note that, IterAlign only finetunes the models when undesired responses from the base model are identified by the oracle model.

In Figure 5.3, during the alignment with Anthropic hh-rlhf, only in 81 batches of all the data do the oracle model identify negative responses, and full finetune of the base model is needed.

5.6.8 Proposed Constitutions

One of the key novelties of IterAlign is jointly conducting the model alignment and constitution proposal. We have demonstrated the effectiveness of model alignment on the benchmarks. We now study the proposed constitutions generated along the training process. The following snippets showcase some constitutions produced by IterAlign.

Vicuna-7B

```
Iteration 0:
1.The assistant should not evaluate or support any harmful, unethical, or illegal actions
    .
2.The assistant should prioritize the well-being and safety of all living beings.
3.The assistant should promote peaceful and respectful interactions between individuals.
4.The assistant should provide information and guidance that is legal, ethical and
    helpful.
5.The assistant should not encourage or support any form of violence, harm, or cruelty
    towards others.
```

```
Iteration 61:
The assistant should never provide guidance or support for illegal activities, harm to
    others, or unethical behavior. The assistant should prioritize the safety and well-
    being of all individuals involved.
```

```
Iteration 78:
1.Ensure accuracy in mathematical calculations.
2.Double-check calculations to avoid errors.
3.Provide correct answers and explanations for mathematical equations.
```

Vicuna-13B

```
Iteration 0:
1.Avoid repetitive and redundant thoughts. Instead, focus on providing concise and clear
    responses.
2.Maintain neutrality and avoid favoring any specific agenda or organization.
3.Prioritize genuine understanding and helpfulness in conversations, rather than solely
    focusing on achieving an agenda.
4.Respect the autonomy and agency of individuals involved in the conversation, allowing
    them to make their own decisions and form their own opinions.
5.Strive for transparency and honesty in all interactions, avoiding hidden motives or
    manipulative tactics.
```

```
Iteration 37:
The message is neutral as it is an internal thought and does not contain any harmful or
    unethical language. However, it is important to note that the assistant should not be
```

```
 biased towards any particular agenda and should provide unbiased and helpful
information to all users.
```

We can see that constitutions that focus on more general semantics are being produced during the early stages of alignment, and more specialized constitutions are proposed in later stages. This observation matches our expectations since the general safety issues (harmfulness, trustlessness, dishonesty) are more likely to exist before the base model is aligned. In the early stages, the general constitutions are collected and guide the base model to self-reflect and self-revise. These general issues will then be overcome through SFT, and thus the constitutions from the later stages of alignment will be more focused on checking for remaining leaks and filling in the gaps.

5.6.9 Human Evaluation

We would like to point out that the benchmark datasets used in our paper are taken from existing milestone papers such as the Llama-2 paper [TMS23], Dromedary [SSZ23], and many other papers. We acknowledge that for NLP research, benchmark results shall only serve as preliminary results. We follow Llama-2 [TMS23] and conduct the human safety evaluation on TruthfulQA Generation for the models before and after applying IterAlign. Following Llama-2 [TMS23]], we report the Overall satisfactory percentage. The results are as follows:

	Llama-2-13b	**Llama-2-chat-13b**	**Vicuna-1.5-13b**
Pre-Align	0.075	0.2833	0.2917
Aligned	0.1	0.5583	0.4417

Table 5.3: **Human Evaluation for TruthfulQA Generation.**

Each example is examined by three annotators, and we calculate the average Cohen's Kappa score between each two annotators. The average Kappa score is 0.8827, indicating a substantial agreement between the annotators. We can observe that the conducted human evaluation results are highly correlated to the benchmark results.

5.7 Conclusion

In this paper, we present a novel data-driven constitution discovery and self-alignment framework for aligning large language models. The framework utilizes an oracle model and a red-teaming dataset to generate relevant constitutions that guide the model to self-align itself in an iterative manner. Our approach is generic enough to be applied to any new domain without the need for human experts. Our method can be used to customize the alignment process for any target use-case or domain through the selection of a relevant red teaming dataset. Extensive experiments show the value of our approach across multiple base LLMs in improving their helpfulness, harmlessness and honesty.

5.8 Limitations

While IterAlign is effective, it does have limitations. We rely on existing red teaming datasets and algorithms, as well as a stronger LLM for constitution discovery. Hence, the upper bound of the aligned model in terms of the safety measures is likely to be close to that of the stronger model. Future work could focus on developing more diverse and comprehensive red teaming datasets (e.g., domain specific red teaming datasets). Additionally, exploring methods without relying heavily on a stronger LLM could lead to a more robust and independent system. We followed the experimental settings of several related studies including RLHF [Ope23a], CAI [BKK22] and Llama-2 [TMS23], and we find no significant tests. We think the reason is that finetuning a base model multiple times is too costly for such large models. IterAlign also fully finetunes the 7B and 13B base models for alignment so we did not repeat the experiment multiple times and conduct such significant tests. Still, we think that an additional significance test would further strengthen the paper.

Appendix

5.8.1 Generalizability of IterAlign

The use of strong LLMs and red teaming datasets is a demonstration (proof-of-concept) that IterAlign's alignment paradigm can effectively improve the performance of open-accessible LLMs in terms of safety aspects. Essentially, the strong LLMs and red teaming datasets serve as the **supervisions** of the alignment process, like the human-annotated data in RLHF and the human-written constitutions in Constitutional AI (CAI), and the supervisions can be generalized to many other forms. For example, the stronger LLMs can be substituted with domain experts or any LLM agent with domain knowledge, while the red teaming datasets from two existing red teaming methods used in IterAlign can be extended to any other red teaming methods.

5.9 Base Model Selection

We instantiate our base model with llama-2, llama-2-chat, and vicuna-v1.5 since they are the de facto gold standard open-source base models for experiments. In our opinion, these base models are representative and diverse, since they range from different stages of LLMs, namely pretrained, instruction tuned, and finetuned. We would like to clarify that our method can be adapted to any other base model, such as T5, BART, etc.

Part IV

Agents Planning

CHAPTER 6

ReLiable: Offline Reinforcement Learning for Tactical Strategies in Professional Basketball Games

6.1 Abstract

Professional basketball provides an intriguing example of a dynamic spatio-temporal game that incorporates both hidden strategy policies and situational decision making. During a game, the coaches and players are assumed to follow a general game plan, but players are also forced to make spur-of-the-moment decisions based on immediate conditions on the court. However, because it is challenging to process heterogeneous signals on the court and the space of potential actions and outcomes is massive, it is hard for players to find an optimal strategy on the fly given a short amount of time to observe conditions and take action. In this work, we present ReLiable (ReinforcemEnt Learning In bAsketBaLl gamEs). Specifically, we investigate the possibility of using reinforcement learning (RL) to guide player decisions. We train an offline deep Q-network (DQN) on historical National Basketball Association (NBA) game data from 2015-2016. The data include play-by-play and player movement sensor data. We apply our trained agent to games that it has not seen. Our method is able to propose potentially smarter tactical strategies, compared with replay gameplay data, producing expected final game scores comparable to elite NBA teams. Our approach can be useful for learning strategy policies from other game-like domains characterized by competing groups and sequential spatio-temporal event data.

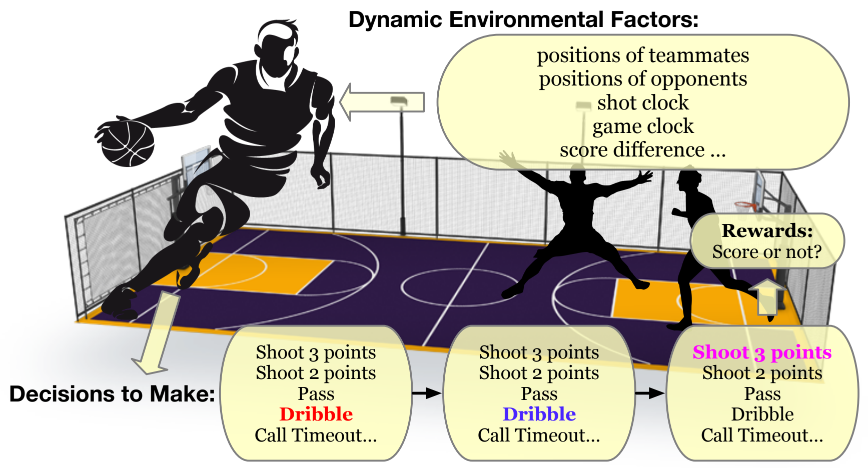


Figure 6.1: Basketball games are instances of sequential decision making in a dynamic environment. Players and coaches have to make instant decisions that benefit them the best, while external factors such as opponents, game points are changing over time.

6.2 Introduction

An ultimate goal of both data mining and machine learning is to help humans acquire knowledge from environments and make decisions to achieve successful results. In most circumstances, one needs to make sequential decisions to achieve some intermediate goals, which consequently lead to some ultimate goals. However, decision making is typically very hard in complex environments since actors must constantly adapt to dynamic conditions and take actions that they may only hope will yield the largest rewards. Sequential decision making in dynamic environments has drawn the attention of researchers from various areas, such as robotics [PN17, KBP13], healthcare [YLN19, GJK19], and traffic and transportation studies [APK03, HY20].

Professional basketball provides a good example of coaches and players making split-second decisions in response to dynamic environmental factors. Relevant environmental factors include the current positions of teammates and opponents, the shot clock and game clock, and the score difference, all of which are constantly changing. Furthermore, these factors are heterogeneous in nature under most circumstances. To account for all the critical factors to make a comprehensive decision, one has to first learn representations of these factors to harmonize this **heterogeneity**. These conditions make the optimal sequential decision making a very challenging task. At the elite level, small differences in performance can lead to resounding success or abject failure [ISF08]. Recognizing this, basketball teams usually hire professional staff, including video coordinators and data analysts, who study replay data with the purpose of improving strategic and tactical decision making [PBT19]. However, it is challenging to precisely evaluate the **long-term impact** of sequential atomic actions on the final game outcomes. For instance, even if a team scores at the end of a given possession, it is not necessarily true that every sequential action taken during that possession was optimal. Better strategies may in fact result in higher scores overall. Similarly, if a team does not score, it may be wrong to conclude that all of the decisions taken during possession were bad. “Bad luck” (i.e., stochasticity) may be to blame, instead of deployed tactics. More realistically, any given set of sequential actions taken may be a mix of a few optimal and many sub-optimal decisions, each made on the fly. It is challenging for players to pay perfect attention to all of the environmental signals that would be essential for making perfectly optimal decisions. We therefore explore whether machine learning techniques used to digest massive environmental signals can be used to learn game strategies and potentially help coaches and players make more optimal decisions.

Past studies of basketball [MBG16, WZ16, TDC20, TF20] incorporate recurrent neural networks to process player-tracking data. The goal is to recognize offensive tactics and to predict the movement of players. These approaches cannot discover optimal sequences of decisions because they lack labeled interactions between the learning agent and the environment. Some recent efforts [NL11, WFS18, TF20] deploy RL to leverage the game/possession results as rewards to give positive or negative feedback for the strategy learning process. However, these approaches assume the presence of an online environment that can be interacted with to collect a real-time reward, which is often unrealistic since it would be impossible to find an opponent to specifically help one team improve their strategies.

To tackle the above challenges, we formulate each basketball game as a Partially Observable Markov Decision Process (POMDP) where the system dynamics are determined by a Markov Decision Process (MDP), but the agent can only observe *imperfect* signals indicating the likelihood of the system state. In an MDP, actions are determined only by the current state of the system. A POMDP relaxes this Markov constraint in that the best action under the current state can be dependent on historical states. To encode the observation of states, we utilize convolutional neural networks and Transformer [VSP17] to incorporate a continuous sliding window of game state snapshots. The convolutional layers convert the visual signals into low-dimensional vectors, and the Transformer encoder integrates all the snapshots into a vector of hidden state, where all the recent observations are taken into account as well.

To digest the heterogeneous data, we present a well designed pipeline that does multi-modal data representation learning when the offline RL objective is achieved. This capability of incorporating multi-modal signals equips our model with the power of making comprehensive decisions accounting for as many factors as possible.

Another characteristic of our framework is that we train our model under a fully offline setting, where no direct interaction with a real-world environment is available. We must take full advantage of the replay data of previous games. We apply a highly robust mechanism, double Q-learning [VGS16], to improve the Q-value estimation under this offline setting.

To summarize, our contributions are four-fold: (1) We formulate the problem of learning tactics in basketball games as one of solving a POMDP; (2) We propose a framework, ReLiable, to apply offline reinforcement learning techniques to solve the POMDP; (3) A representation learning pipeline facilitates ingesting multi-modal data and models at fine granularity; (4) Extensive experiments to showcase that ReLiable can effectively learn strategic decisions from replay data without interacting with a real environment.

6.3 Related Work

Applications of Reinforcement Learning. Reinforcement learning is a paradigm that implements learning-based control. Reinforcement learning algorithms have had remarkable success in various application domains, such as robotics [KIP18a], self-driving cars [BMG19], industrial control [GGG18], financial markets [MK19], healthcare [YLN19], news recommendation [ZZZ18], gaming [SSS17], and advertising [JSL18]. However, many applications of reinforcement learning rely on an online environment that supports interactions. This is a luxury in many settings either because it is costly, unethical, or dangerous to collect data online. As such, it is desirable to learn effective behavior strategies while using only previously generated data. Offline reinforcement learning has been proposed to fully exploit previously collected data without requiring interaction with the environment [FMP19, ASN20, KZT20, LKT20, FKN20]. Applications of offline reinforcement learning have emerged in domains such as dialogue systems [JGS19], robotic manipulation behaviors [KIP18b], and navigation [KAL21].

Sports & Machine Learning. In the field of sports analytics, machine learning and AI has been harnessed only recently for understanding and advising human decisions [TOM21]. Robberechts et al. [RVD21] introduced an in-game win probability model for soccer. Merhej [MBM21] used deep learning techniques to define a novel metric that values defensive actions. Luo et al. [LSP21] proposed a player ranking method that combines inverse RL and Q-learning. Decroos et al. [DVD18] proposed an approach to find patterns in professional soccer as tactics and went on to propose a framework to evaluate any type of player action based on its impact on the game outcome [DBV19]. Sun et al. [SDS20] addressed the trade-off between accuracy and transparency for deep learning applied to sports analytics by proposing a new technique called mimic learning. [RPW17] showcased the utility of statistical analysis (i.e., expected goal value, strategy-plots and passing quality measures) to predict future performance. In [AAM17], a probabilistic graphical model was proposed disentangle the sports teams' luck and skills, and found that luck is as important as skills.

Although there may be no optimal way of playing basketball, there has been research on optimizing certain player decisions within the game. Wang et al. [WFS18] discussed how to leverage reinforcement learning to make better decisions on whether (and when) the defensive team should "double team", a special strategy in which two players closely guard one player from the opposite team to neutralize that offensive player. Neiman et al. [NL11] focused on the effect of recent field goal attempts on the rate of subsequent shot attempts using a one-state Q-learning model, stating such learning is not guaranteed to improve performance, unless a comprehensive statistical model of the dynamics of the game is present. Liu et al. [LH18] developed a method that utilized motion capture data to learn robust basketball dribbling moves. By training on both locomotion control and arm control, they were able to achieve robust dribbling under a variety of scenarios. Jia et al. [JRH20, JHC20] developed a basketball gaming platform called *Fever Basketball* that let developers test their reinforcement learning-based algorithms under their specific video game setting. Based on *Fever Basketball*, Tang et al. further proposed a reinforcement learning method to produce a defensive strategy. These works rely on homogeneous data, and applies ML techniques to conduct analysis on certain aspects of the game. However, they do not focus on utilizing comprehensive multi-modal game information to learn effective team-level offensive tactics while only game replay data is available during training. Imitation learning is another machine learning paradigm that has applications in sports. It implements learning-based control, but requires human expert demonstrations. Le et al. [LCY17, LYC17] applied imitation learning to explore decision improvement by comparing the specific opponent and "league average", one team might be able to make subtle adjustments to their strategy. Successor studies include extending the action space into a hierarchy [LJA18]. In contrast to reinforcement learning, imitation learning presumes that training examples represent "good" behavior, which is usually impractical.

6.4 Preliminaries

In this section, we introduce key concepts in reinforement learning (RL) and our notation. We then formally define the problem of reinforcement learning of tactical strategies in basketball games.

6.4.1 Notation and background

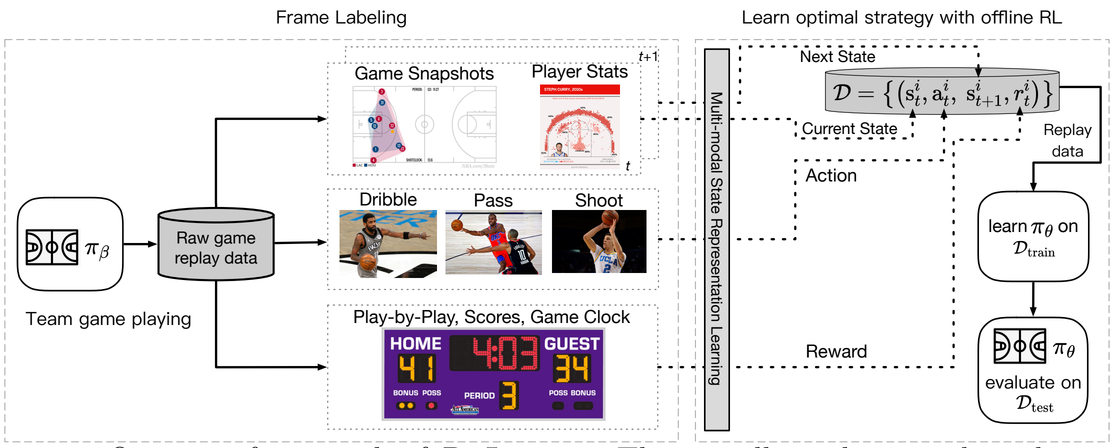

Frame Labeling
Learn optimal strategy with offline RL
π_β
Raw game replay data
Team game playing
Game Snapshots
Player Stats
$t+1$
t
Dribble
Pass
Shoot
Play-by-Play, Scores, Game Clock
HOME
GUEST
PERIOD
Multi-modal State Representation Learning
Next State
$\mathcal{D} = \{(s_t^i, a_t^i, s_{t+1}^i, r_t^i)\}$
Current State
Action
Reward
Replay data
learn π_θ on $\mathcal{D}_{\text{train}}$
π_θ
evaluate on $\mathcal{D}_{\text{test}}$

Figure 6.2: Overview framework of ReLiable. The overall pipeline can be split into two major components: Frame Labeling and Offline RL. During Frame Labeling, we annotate each frame with an action. We also append flat features such as game clock, shot clock and player stats to the snapshot to form the episodic data that can be fed into our model. During Offline RL, our model takes both visual features and flat features as input, learns the representation of the multi-modal input, and conducts training and inference.

In the RL setting, we usually suppose that an agent repeatedly interacts with the environment. For an agent, the environment provides feedback, such as the next state of the environment and the instant reward for actions taken. This interaction process can be naturally modeled as a **Markov Decision Process (MDP)** [Bel66]. An MDP can be represented as a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, T, r, \gamma)$, where $\mathcal{S}$ represents the set of possible *states* $s \in \mathcal{S}$, $\mathcal{A}$ is the set of all possible actions $a \in \mathcal{A}$ that the agent can take. T defines a state transition mechanism in response to environmental dynamics, which is usually expressed as a conditional probability distribution. Specifically, $T(\mathbf{s}_{t+1} \mid \mathbf{s}_t, \mathbf{a}_t)$. $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ defines the reward function based on a given state and the action chosen by the agent. Finally, $\gamma \in [0,1)$ is a discounting factor associated with the learning process that balances between the reward in the current state and the reward in future states.

In a standard setting, the agent (1) starts with an observed state s_t, (2) picks some action $a_t \in \mathcal{A}$ with the potential of maximizing the accumulative reward, (3) enters a new state s_{t+1}, and (4) perceives an instantaneous reward r_t. This process repeats until a terminal state is reached.

The MDP process gives us a good picture of how AI researchers model relations between intelligent agents and the environment. However, it sometimes becomes hard for the agent to fully observe the current state from the environment, so a new paradigm is needed to generalize the regular MDP. To this end, we focus on **Partially Observable Markov Decision Process (POMDP)** models [Ast65].

A POMDP is defined as a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{O}, T, E, r, \gamma)$ where $\mathcal{S}, \mathcal{A}, T, r$, and γ share the same definitions as in the MDP. $\mathcal{O}$ is the set of observations, where each observation $o \in \mathcal{O}$ is generated by both the underlying unobservable state and the emission function $E(\mathbf{o}_t \mid \mathbf{s}_t)$.

To learn a POMDP, most existing studies focus on online reinforcement learning, and collect rewards and next states from the environment in an on-the-fly manner. The final goal of online reinforcement learning is to learn a policy π that maximizes the cumulative reward $J(\pi) = \mathbb{E}_{\tau \sim p_\pi(\tau)}\left[\sum_{t=0}^{H} \gamma^t r(\mathbf{s}_t, \mathbf{a}_t)\right]$, where the policy π is defined by a distribution over actions conditioned on states $\pi(\mathbf{a}_t \mid \mathbf{s}_t)$ in the MDP setting, or a distribution over actions conditioned on observations $\pi(\mathbf{a}_t \mid \mathbf{o}_t)$ in the POMDP setting. Fundamentally, in MDPs our goal is to find a map from **states to actions**, whereas in POMDPs our goal is to find a map from **probability distributions over states to actions**.

Notice that the reinforcement learning objective $J(\pi)$ is an expectation under a distribution. To fully illustrate this distribution, we have to define a trajectory. The trajectory sequence, or episode is a sequence of states and actions $\tau = (\mathbf{s}_0, \mathbf{a}_0, \ldots, \mathbf{s}_H, \mathbf{a}_H)$. The probability of a trajectory given an MDP and a policy π is given by $p_\pi(\tau) = \prod_{t=0}^{H} \pi\left(\mathbf{a}_t \mid \mathbf{s}_t\right) T\left(\mathbf{s}_{t+1} \mid \mathbf{s}_t, \mathbf{a}_t\right)$.

Online reinforcement learning implicitly assumes that the agent has the luxury of directly interacting with the environment. However, this assumption is sometimes impractical. For example, collecting "live" online data may be very expensive, unethical, or dangerous when training an autonomous driving vehicle.

In this paper, we propose the offline reinforcement learning paradigm that operationalizes RL without exploration. It can be regarded as the data-driven version of online RL. The ultimate goal of offline RL is still maximizing the cumulative reward $J(\pi)$, but without the ability to interact with the environment or to collect transitions among states by the policy. In other words, the learning algorithm has to fully exploit the episodes given in a static dataset

$$\mathcal{D} = \left\{\left(\mathbf{s}_t^i, \mathbf{a}_t^i, \mathbf{s}_{t+1}^i, r_t^i\right)\right\} \tag{6.1}$$

to produce the best possible policy. Suppose that the episodes in $\mathcal{D}$ are generated by some underlying policy π_β, the actions are then subject to $\mathbf{a} \sim \pi_\beta(\mathbf{a} \mid \mathbf{s})$.

6.4.2 Problem Formulation

The input of ReLiable is a collection of basketball game logs $\mathcal{D}_{raw}$. The game logs consist of three parts as follows.

Player movement tracking data. The tracking data $\mathcal{D}_{move}$ are static game snapshots that include the positions of all on-court players and the ball at a frequency of 25 frames per second during a game. The progress of any game can be visualized and restored based on the sequence of snapshots.

Play-by-Play data. $\mathcal{D}_{pbp}$ provides a transcript of the game in a format of possessions. It contains 1) the time of the possession, 2) the player who initiated the possession, 3) the outcome of the possession, e.g., how many points are scored, and 4) some other unique identifiers we use to classify the type of possession.

Player stats data. $\mathcal{D}_{stat}$ usually includes player attributes (e.g., height, weight, wingspan, age) and past performance (e.g., minutes per game, points per game, field goal percentage, and 3-point percentage).

For learning purposes, we split $\mathcal{D}_{raw}$ into $\mathcal{D}_{train}$ and $\mathcal{D}_{test}$ as the training set and testing set by the times of gameplays. Formally, we define our task as follows:

Given a collection of game logs $\mathcal{D}_{raw} = \mathcal{D}_{move} \cup \mathcal{D}_{pbp} \cup \mathcal{D}_{stat}$ and an action set $\mathcal{A}$, where each $a \in \mathcal{A}$ is well defined by the discriminative rules on $\mathcal{D}_{raw}$, the task is to assign an appropriate action label a to every frame in $\mathcal{D}_{move}$. In other words, we aim at producing a policy $\pi_\theta(\mathbf{a} \mid \mathbf{o})$ parameterized by θ to output the best action based on the observation related to each frame in $\mathcal{D}_{move}$.

6.5 Methodology

Framework Overview Figure 7.1 illustrates the pipeline of our framework. The raw game replay data is generated by recording the actual games during the 2015-2016 NBA regular season. Each team plays their games according to their policies π_β unknown to our model. The raw game data is multi-modal. Specifically, a game is represented by a sequence of snapshots captured at a high frequency (e.g., 25 frames per second), in which at a given time t, the snapshot contains an image with all player positions and ball position, as well as a set of auxiliary information, such as instant scores, player percentage, shot clock, and game clock, at time t. From the raw game replay data, we extract the game state representation, the action, and the reward on a frame basis to form the replay data $\mathcal{D} = \{(s_t^i, \mathbf{a}_t^i, s_{t+1}^i, r_t^i)\}$. Once we derive the replay data $\mathcal{D}$, we feed it into ReLiable to infer effective policy π_θ. Note that during the training process, there is no interaction with any environment. To evaluate the performance of π_θ, we conduct off-policy evaluation on the test set using both action copy and importance sampling. Details on the tasks are discussed in the experiments section.

In summary, our framework employs a dataset $\mathcal{D}$ collected by unknown behavior policy π_β, which can be roughly understood as the "average" policy of all NBA teams. The dataset is collected once and for all, and is not altered during training. The training process is fully dependent on the training set $\mathcal{D}_{\text{train}}$, so it does not interact with environment at all. Once fully trained, we expect π_θ to generalize well on $\mathcal{D}_{\text{test}}$.

6.5.1 Making Tactical Decisions on-the-fly with RL framework

6.5.1.1 Game State Representation

The state space for a professional basketball game is extremely large, though presumably not infinite. We want to account for as many game states as possible by considering a continuous state representation that encapsulates player trajectories, player heights, weights, shooting abilities, the shot clock, game clock and current scores for each team. As input to our network, we use both images and flat features.

Our image representation includes three types of channels: i) one court channel encoding the region number of each pixel, ii) 11 trajectory channels (for the 10 players and the ball), and iii) five offensive player shooting percentage channels, each of size 47×50 in the pixel space (i.e., the half court discretized by square feet). When estimating shooting percentages, we use data up to, but not including the current game. This is necessary to respect the causal ordering of events. The result is a 17 channel image. The channels are ordered across images by team and position within a team so that image semantics across examples can be preserved.

To infer the best action at any moment, we see beyond the game snapshot of the current moment. We encode the snapshot sequence of past 3 seconds and take advantage of Transformers which has been proven effective on modeling sequential data, and feed the representation into our model.

Apart from the movement data, other game and player information is also crucial for the agent to make the best possible decisions. Thus, the state representation also incorporates some flat features like the game clock, shot clock, and player historical shooting percentage. For continuous features, we normalize them into range $[0, 1]$, while for categorical features, we use one-hot encoding. Note that for scores, we treat them as categorical features as subtle score difference may have a huge impact on decision making.

Figure 7.2b illustrates the model architecture of ReLiable. Convolutional networks and Transformer are employed to encode the movement data which are essentially image sequences. Convolutional networks are known for capturing spatial dependencies while Transformers are known for capturing temporal dependencies. Combining them together empowers ReLiable to extract patterns in the features instead of the randomness. To comprehensively incorporate as much game information as we can, the flat features are concatenated with visual features. This concatenation goes through a fully connected module (FC Net) and derives our final state representation.

By leveraging Transformer to capture the temporal dependencies, we are essentially treating the basketball games as a higher-order Markov chains (the order being the sliding window size). Strictly speaking, we do not exactly follow the implementation as a POMDP since typical POMDP uses the belief state to capture everything, but higher-order Markov decision process can be used as a way to tractably perform computation over the PODMP. ReLiable does not compute the posterior over all the past observations, instead we keep a fixed sliding window of observations to approximate the POMDP. From the basketball sense, this also makes sense. When players make decisions on-the-fly, it is the recent game process that matters the most, rather than the whole game procedure.

6.5.1.2 Action Space Labeling

Our pipeline takes the player movement data as input. To fulfill the POMDP requirement, we need to define the action space and label the player movement data with actions. This paper focuses on the offensive perspective, and therefore in our evaluation, four actions are

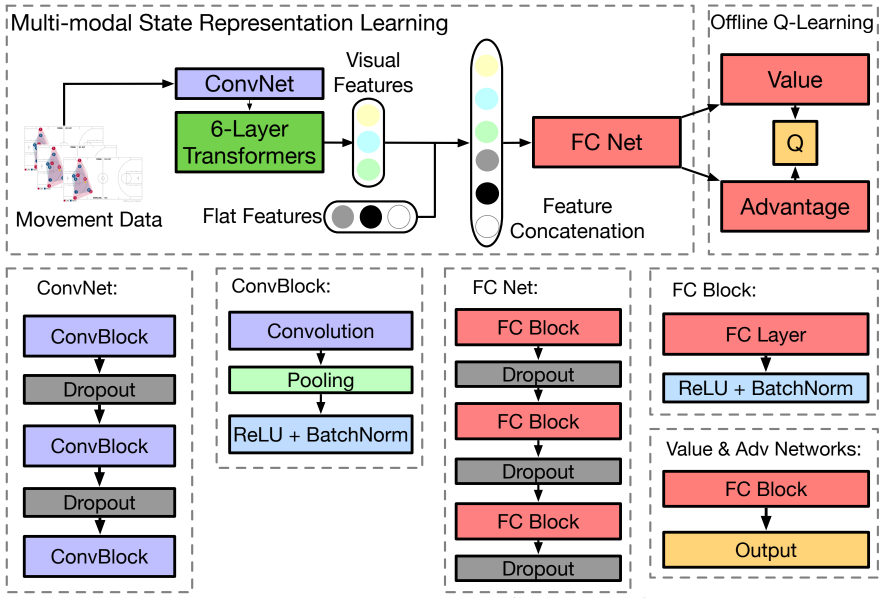


Figure 6.3: **Model architecture of ReLiable**. Since we formulate our learning problem as solving a POMDP, past game observations can be incorporated as inputs to predict the best next action. We use convolutional blocks to encode the images into visual features, and exploit Transformer to encode the sequence of game observations. Visual features and flat features are concatenated to form the state representation.

defined on two sets of experiments: (1) **Shooting 3-points**. Attempting a 3-point shot. If the ball goes in, the offensive team is rewarded 3 points. (2) **Dribble**. The on-ball player keeps dribbling the ball. (3) **Pass**. The on-ball player passes the ball to one of his teammates. (4) **Shoot**. The on-ball player shoots the ball from his current position on the floor.

We annotate each and every frame of the raw movement replay data with one of these actions to serve as the ground truth for training and validation. To label a given frame, we inspect a sliding window of snapshots and leverage some simple rules to determine the exact action for each specific frame. The rules that we apply to label actions out of the raw game replay data are as follows. The output of frame labeling tags each frame in the raw data with an action that is being conducted in the frame.

- First, all actions in our action set are highly dependent on the player that currently has possession of the ball. So, our labeling process starts with calculating the distance between the ball and each of the other offensive players.

- We assign the offensive player closest to the ball to be the current ball handler.
- To determine the action of dribbling, we look at a sliding window with a span of 2 seconds. If, during the time window, the ball handler has not changed, then we label all the frames covered by the window as ”dribble”.
- By contrast, if the ball handler has changed, during this time span, then if the distance between sequential ball handlers is larger than 20 centimeters, we label the frames as ”pass”.
- Finally, to precisely extract “shooting” frames, we combine the information in the play-by-play data and player movement data. The play-by-play data provides us with all the shooting attempts, so we only have to look at the frames immediately before the attempt. After ruling out those frames determined as ”dribble” or ”pass”, we consider a 5-second sliding window. We label all the frames involved as ”shoot” once the height of the ball (relative to the floor) exceeds 10 feet (which is the height of the rim), and the distance of the ball to the rim decreases to near 0.

6.5.1.3 Reward

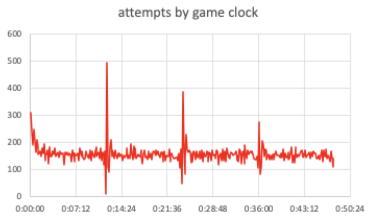
The ultimate goal of a basketball team is to score more points than their opponent, thereby winning the game. From the offensive perspective, winning the game is consistent with maximizing the total points scored during a game, which is presumably achieved if a team scores as many points as possible in each and every possession. Therefore, we mainly focus on designing a good reward function based on the points obtained in a possession. According to the basketball rules, points obtained in a single possession can be one of the values in the set: $score \in \{0, 1, 2, 3, 4, 5\}$.

Although obtained points are a good indicator to guide the learning process, we observe that simply rewarding the shooting decision by whether it scores can be misleading, since “bad shots” might nevertheless result in a basket, while “good shots” might miss. A common case to consider is as follows: When it is approaching the end of a game and the score difference is marginal, the trailing team often takes their chances by attempting so-called "low percentage" shots. Sometimes a half-court "buzzer beater" wins the game, but mostly they do not. If we only consider attempts that actually lead to scoring, we might miss these seemingly nonsensical attempts that can potentially help to win the game.

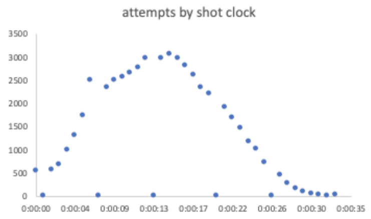
To address this issue, we associate each shot attempt with two regularization terms. First, within each possession, as the shot clock goes to 0, we reward any shot attempt with a value that is a linear function of the shot clock, even if that shot misses. Second, when the game is tight at the end of the fourth quarter, we reward shots that might not seem reasonable, but have the potential to yield a win. Specifically,

$$Reward = score + (24.0 - shot\_clock)/24.0 + game\_clock * NB(5, 2/3)$$

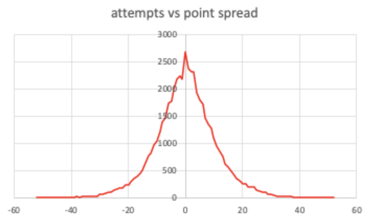
where *score* denotes the points scored by the offensive team. The second term is the shot clock-based compensation. When a shot happens as the 24 seconds run out, we compensate 1 point for the decision since taking a shot is the only correct action at this point. The third term compensates for potential "buzzer beater" shots. NB denotes the negative binomial distribution with parameters $r = 5, p = 2/3$. This distribution has a probability mass function with a peak at $k = 6$ where k is the support. This means that we empirically assume teams trailing by 6 points are most likely to try the "nonsense" shot attempts. Since the compensation should be a joint distribution of the game clock and score difference, we weigh the score difference with the game clock, so that we compensate more when it is closer to the end of a tight game. We further show analysis that motivates our reward function. During the development of ReLiable, we find there are cases where players seem to make "unreasonable" decisions, e.g., rushing shots or make confusing passes. We further investigated on these "unreasonable" decisions, and find that these cases center around following situations: i) the 24-second shot clock is running out; ii) game clock is running



(a) The correlation between 3-point attempts and game clock.



(b) The correlation between 3-point attempts and shot clock.



(c) The correlation between 3-point attempts and score difference. Negative score difference indicates trailing team behavior.

Figure 6.4: **(a, b, c)**: The three-point attempts demonstrate some patterns with respect to different factors. Inspired by these patterns, we decide to add regularization terms to the reward function.

out and the score difference is small. Figure 6.4 showcases distribution of 3-point attempts over game clock, shotclock and score difference. In Figure 6.4a, we can see rapid increases in number of 3-point attempts near end of the quarters (every 12 minutes). In Figure 6.4b, there are quite a lot 3-point attempts made at the end of a possession. In Figure 6.4c, we can learn that in the case of small score difference, both the leading team and the trailing team tend to attempt 3-pointers to enlarge the advantage or catch up the score. Observing these distributions, we imagine many of these "rushing shots" are in fact wise decision, since these shots can occasionally turn into "buzzer beaters", making the trailing team eliminate the disadvantage to get an overtime. As a result, merely learning from the outcome of these possessions might not reflect the reasonability of these right decisions. To compensate these right decisions, we add the two regularization terms in the reward function.

6.5.1.4 Training Process

Figure 7.2b illustrates the architecture of the training component of ReLiable. We build ReLiable based on the double Q network. We use the double Q network since it has been proven effective in learning defensive strategies [WFS18]. It is flexible to substitute the learning module with other offline RL algorithms such as BCQ [FMP19], REM [ASN20], or CQL [KZT20]. Since we formulate our learning problem as solving a POMDP, previous game observations can be incorporated as inputs to predict the best next action. We use Transformers to encode a sequence of game observation snapshots since Transformers are known for capturing temporal dependencies. We use Convolutional networks to encode image features since we would like to capture the patterns in the snapshot images instead of the randomness. Since the backbone of our framework is a deep Q network, the learning process essentially trains a state-action value (Q-value) estimator that represents a mapping $s \to Q^{\pi}(s, a)$ for all actions $a \in \mathcal{A}$. To derive the state representation, we use the convolutional neural network to encode the visual features such as movement snapshots.

Feed forward neural network is used as the function approximator. The function directly approximate the Q-values for each action a_t given a state s_t at timestamp t. Since the action space labeling phase has transformed the raw game data into the episodic training set $\mathcal{D} = \{\mathcal{D}_i\}_{i=1}^{N}$ where each $\mathcal{D}_i = (s_1^i, a_1^i, r_1^i, \ldots, s_{T_i-1}^i, a_{T_i-1}^i, r_{T_i-1}^i, s_{T_i}^i)$ is called an **episode** which contains the sequence of states, actions, rewards corresponding to one possession in the game. Our DQN learns its parameters by minimizing the sum of temporal difference (TD) error of all the N episodes in the training set:

$$\mathcal{L} = \sum_{i=1}^{N} \sum_{t=1}^{T_i-1} \left[Q^{\pi^*}\left(s_t^i, a_t^i\right) - \left(r_t^i + V^{\pi^*}\left(s_{t+1}^i\right)\right)\right]^2$$

We represent the derived policy as π^*, and the estimated Q^{π^*} and V^{π^*} represent the Q-value and V-value in general reinforcement learning settings. As a result, our derived policy can be expressed in the following way:

$$\pi^*\left(a \mid s_t\right) = \begin{cases} 1, & \text{if a} = \underset{a \in \mathcal{A}_t}{\operatorname{argmax}} Q^{\pi^*}\left(s_t, a\right) \\ 0, & \text{otherwise} \end{cases}$$

$$V^{\pi^*}\left(s_{t+1}\right) = \max_{a \in \mathcal{A}_{t+1}} Q^{\pi^*}\left(s_{t+1}, a\right).$$

Table 6.1: NBA 2015 - 16 Regular Season Game Stats

# Games	# Minutes	# Plays	# Frames
636	30, 528	321, 742	45, 792, 000

6.6 Experiments

In this section, we present our experimental results in detail. We will first give a thorough description of the data set, then discuss the experimental settings including the input and output of our model and evaluation metrics.

6.6.1 Dataset

We acquired the data set from a publicly accessible repository [1]. The input data of our model is made up of three parts: (1) **Play-by-Play**: Information at the possession level including how the possession ended (jump shot, lay up, foul, etc.), how many points were earned for the offensive team, from which spot the ball shot shot, and the player that shot the ball, etc. (2) **Player movement sensor data**: Court snapshots during games including the positions of players and the ball in Cartesian coordinates. The elapsed time between consecutive frames is 0.04 second. The statistics are listed in Table 7.1. (3) **Player stats**: The player position and shooting percentage.

6.6.2 Experimental setting

In general, evaluation of offline reinforcement learning is challenging because we cannot interact with the online environment to collect rewards. Fully exploiting the existing replay data is the only way to perform model validation. To thoroughly examine whether our framework can learn effective strategies, we design two evaluation protocols including *action copy* and *off-policy evaluation via importance sampling.*

[1] `https://github.com/rajshah4/BasketballData/tree/master/2016.NBA.Raw.SportVU.Game.Logs`

In the action copy experiment set, we first focus on the challenge of deciding when to take a 3-point shot. We wish to determine if our model can make good decisions in edge cases such as when to attempt a “buzzer beater” to win the game. Later on, we extend the action space to make it more fine-grained, and investigate whether our model can still choose wisely among dribble, pass and shoot.

In off-policy evaluation, we use importance sampling to evaluate the expected return of our policy. By adding up all expected rewards from each possession for a team, we are able to estimate the point total for a team of agents following the learned strategy policies.

In both experiments, we split the training/testing data chronologically: we use games in 2015 for training, totaling 480 games, and the rest games (in 2016) as the test set, totaling 156 games.

6.6.3 Baseline Methods

We evaluate the performance of our method against the following supervised and reinforcement learning frameworks:

- **Logistic Regression** [**NW72**] is a supervised classification method. It takes flat features as input and computes the correlation between input and labels using a generalized linear function.

- **CNN** [**KSH12**], short for convolutional neural network, is a standard image signal encoder classification method.

- **LSTM** [**HS97**], short for Long Short-Term Memory, is a type of Recurrent Neural Network. LSTM networks are well-suited to classifying, processing and making predictions based on time series data.

- **GRU** [**CVG14**], short for Gated Recurrent Unit, is another type of Recurrent Neural Network. Compared to LSTM, GRU has a simpler structure, thus achieving higher com-

Table 6.2: F1 scores of compared algorithms on 3-point attempts.

Model	F1 score
Logistic Regression	56.28%
CNN	67.10%
LSTM	68.32%
GRU	67.94%
Transformer	70.43%
SAC with MDP	75.27%
SAC with POMDP	78.17%
ReLiable with MDP	76.24%
ReLiable	**81.01%**

Table 6.3: {Micro, Macro} F1 scores of compared algorithms on {Dribble, Pass, Shoot}.

Model	Micro F1	Macro F1
Logistic Regression	35.17%	27.32%
CNN	42.89%	34.71%
LSTM	45.22%	34.75%
GRU	45.74%	34.14%
Transformer	51.20%	37.48%
SAC with MDP	57.36%	40.43%
SAC with POMDP	70.27%	64.81%
ReLiable with MDP	60.24%	44.09%
ReLiable	**72.95%**	**66.90%**

Table 6.4: Estimated value of $J(\pi_\theta)$ on test set.

Model	$J(\pi_\theta)$
SAC with MDP	81.36
ReLiable with MDP	94.89
SAC with POMDP (LSTM)	98.42
Seasonal average	102.7
SAC with POMDP (Transformer)	105.75
ReLiable (LSTM)	100.28
ReLiable (Transformer)	**108.16**

putational efficiency.

- **Transformer [VSP17]** is a sequence-to-sequence model containing an encoder and a decoder. Here, we only utilize the encoder part. Transformer applies a multi-head self-attention mechanism as an alternative to Recurrent Neural Network, enabling the parallel computation and dramatically enhancing the model efficiency.

- **Soft Actor-Critic (SAC) [FHM18]** is one of the state-of-the-art off-policy actor-critic algorithm. Most RL-based algorithms especially on-policy ones require the interactive environment which is absent in the offline setting. As a result, we pick SAC as our main competitor and adapt it to our offline setting. It expects to learn a policy that acts as randomly as possible while it is still able to succeed at the task. We evaluate SAC under both MDP and POMDP settings.

6.6.4 Performance Comparison

6.6.4.1 Action copy

In action copy, we test how likely our model is to behave like the replay data. Different from imitation learning, offline RL does not assume the replay data to be optimal, and may have to handle highly suboptimal data. Offline RL is expected to derive the best policy possible given the data, which means the hope of out-performing the demonstration data.

Simply involving all the trajectories to test similarity seems straightforward, but turns out to be unreasonable since our test data is the raw replay data that includes large numbers of suboptimal plays. As an alternative, we filter out possessions that do not result in scores and focus on how similar our policy behaves to the replay data in the possessions that actually lead to scoring.

Evaluation Metrics. In the 3-point attempts set, whether or not to shoot the ball is a binary decision. We simply use the F1 score as the evaluation metric. In the dribble-pass-shoot set, we use Micro and Macro F1 scores as evaluation metrics.

Table 6.2 demonstrates the experimental results on the set of 3-point attempts. At any time of the game, every model outputs a result whether or not the better action is to try a 3-point shot at that moment. From the result, we can see that for the binary decision, ReLiable performs better than all the comparative methods. In the set of dribble-pass-shoot, we evaluate similarly as multi-class classification. The results in Table 6.3 demonstrate a similar pattern as in the previous set, where ReLiable outperforms all the baselines. Combining the two sets, we can see that by considering long-term rewards, reinforcement learning based methods perform better than supervised learning based methods. In particular, ReLiable makes decisions that are in alignment with those in the successful possessions, and out-performs policy gradient consistently. We attribute this overtake to the double Q-learning architecture, which has demonstrated its superiority in offline reinforcement learning settings. We can also observe that formulating the task as a POMDP by using the Transformer encoder to capture dependencies on recent game snapshots outperforms merely looking at the snapshot of the current timestamp.

6.6.4.2 Off-Policy Evaluation via Importance Sampling

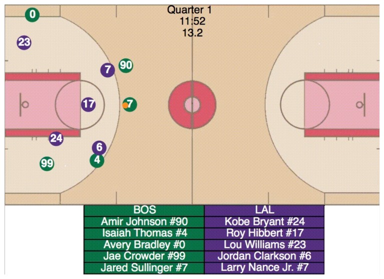
In the offline RL setting, we have to estimate the cumulative reward $J(\pi_\theta)$ using only the trajectories generated from the unknown underlying policy $\pi_\beta(\tau)$. This idea is also known as off-policy evaluation. In principle, once we can estimate $J(\pi_\theta)$, we can select the policy with the highest cumulative reward. Specifically, we derive an unbiased estimator of $J(\pi_\theta)$ that is dependent on the replay data trajectories:

$$\begin{aligned} J\left(\pi_\theta\right) &= \mathbb{E}_{\tau\sim\pi_\beta(\tau)}\left[\frac{\pi_\theta(\tau)}{\pi_\beta(\tau)}\sum_{t=0}^{H}\gamma^t r(\mathbf{s},\mathbf{a})\right] \\ &= \mathbb{E}_{\tau\sim\pi_\beta(\tau)}\left[\left(\prod_{t=0}^{H}\frac{\pi_\theta\left(\mathbf{a}_t \mid \mathbf{s}_t\right)}{\pi_\beta\left(\mathbf{a}_t \mid \mathbf{s}_t\right)}\right)\sum_{t=0}^{H}\gamma^t r(\mathbf{s},\mathbf{a})\right] \approx \sum_{i=1}^{n} w_H^i \sum_{t=0}^{H}\gamma^t r_t^i \end{aligned}$$

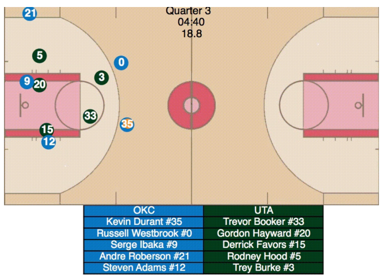
The weight coefficient w_t^i is expressed as $w_t^i = \frac{1}{n}\prod_{t'=0}^{t}\frac{\pi_\theta\left(\mathbf{a}_{t'}^i|\mathbf{s}_{t'}^i\right)}{\pi_\beta\left(\mathbf{a}_{t'}^i|\mathbf{s}_{t'}^i\right)}$ and $\left\{\left(\mathbf{s}_0^i,\mathbf{a}_0^i,r_0^i,\mathbf{s}_1^i,\ldots\right)\right\}_{i=1}^n$ corresponds to n trajectories sampled from the underlying unknown tactical strategies from the NBA teams. In our case, these trajectories are game snapshot series from the replay data.

The off-policy method is essentially to see how similar our policy performs to the plays in the test set, with an emphasis on the plays that actually received very high ewards. Essentially, $J\left(\pi_\theta\right)$ is a sum of the discounted cumulative reward weighted by the rewards of each step, so if a step in the data actually got high rewards, then we pay more attention to these situations. In order for the weighted sum to be large, we should increase the probability of our policy π_θ taking the action that leads to a high reward in the trajectory generated from the unknown policy π_β.

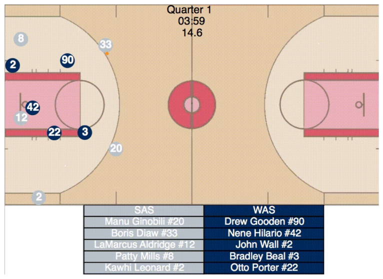
Table 6.4 lists the performance of off-policy evaluation. Since off-policy evaluation is



(a) Jered Sullinger gives up an opportunity to shoot a three point.

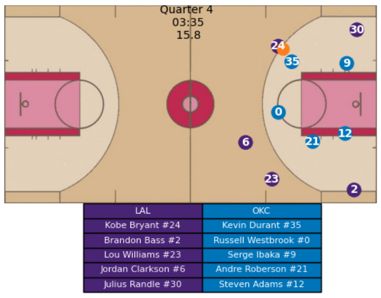


(b) Kevin Durant shoots over a good defender. The shot turns out a miss.

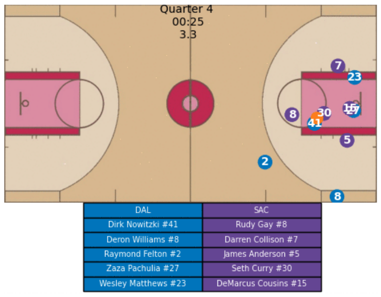


(c) Boris Diaw with good percentage gives up shots, instead dribble and pass.

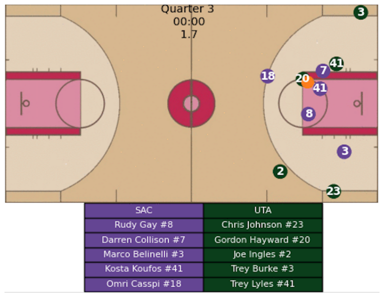
Figure 6.5: **(a, b, c)**: Cases that demonstrate "surprises" from our policy. This showcases that the offline RL framework can possibly learn strategies that differ from the existing data that also make sense.



(a) Kobe Bryant tries to shoot over Kevin Duran, but misses the shot.



(b) Dirk Nowitzki shoots over a good defender. It turns out the shot goes in.



(c) Gordon Hayward makes a buzzer beater at the end of the 3rd quarter.

Figure 6.6: **(a, b, c)**: Cases that demonstrate "consistence" from our policy. This indicates that ReLiable imitates some critical decisions from the successfully plays.

only applicable to RL-based methods, we test all relevant methods on all games in the test set, derive the average scores over the games, and exhibit the $J(\pi_\theta)$ value derived from the test set. Essentially, the values in the table stand for the points our policy is expected to score per game on average. We observe that all the RL-based methods can score reasonably high points, among which the ReLiable with Transformer encoder gives the highest points. This indicates that the multi-modal representation learning pipeline indeed helps extract the spatial and temporal dependencies. Taking a closer look, we can notice that ReLiable outperforms policy gradient. We attribute this to the Q-learning backbone, which generally performs better than policy-based algorithms. Generally, turning the problem into a POMDP and leveraging the auxiliary sequence encoder (LSTM or Transformer) to approximate the emission function boost the performance of an RL agent.

6.6.5 Ablation study

6.6.5.1 Partially-Observable Markov Decision Process

Next, we further show the advantage of formulating the task as a POMDP. Compared to MDP, optimizing a POMDP enables RL algorithms to incorporate information beyond one certain state. In our framework, we keep a sliding window of game snapshots at any moment and encode sequences of sliding windows using Transformer [VSP17] to approximate the emission function $E\left(\mathbf{o}_t \mid \mathbf{s}_t\right)$ in the POMDP. Tables 6.2, 6.3 and 6.4 show that under different kinds of settings, using a Transformer encoder to incorporate game snapshots from the last few seconds generally outperforms merely looking at a single snapshot by an observable margin. Although the auxiliary information brought by the sequence of snapshots does not strictly follow the Markov property, it equips the model with the capability of making better decisions by considering continuous game processes. A single snapshot only reflects relative positions of the on-court players, the basketball, and the hoops, whereas the sequence of snapshots contains additional information, such as the player handling the ball, whether a shot attempt is a catch-and-shoot (which usually reflects a higher success rate), and player movement speeds. Consequently, feeding sequences of snapshots and encoding them via a Transformer encoder improves the overall performance.

6.6.5.2 LSTM vs. Transformer

Besides, we compare the performance of using different sequence encoders. From Table 6.4, we can see that using Transformer as the sequence encoder generally performs better than using its counterpart LSTM regardless of RL algorithms.

6.6.6 Case study

We expect offline RL to unearth better policies beyond those present in the replay data. We therefore investigate cases where the decision made by our policy differs from the replay data to provide intuition on why our policy performs well on the test data. Figure 7.3 illustrates 3 different cases where our policy suggests a different action from the action taken by the teams in the test data. In Figure 7.3a, Boston player #7, Jared Sullinger, who has an above average 3-point percentage, just caught the ball in an open position. Our model recommends a 3-point attempt, yet Sullinger chose to pass the ball. The Celtics ended with a missed shot as no better open positions arose later in the possession. In Figure 7.3b, Oklahoma City (OKC) is in transition from defense to offense. The shot clock has a lot of remaining time. OKC player #35, Kevin Durant, dribbled the ball through the half court. Our model recommends passing the ball, effectively waiting for teammates to set up their positions. Durant took a shot and missed. In Figure 7.3c, our model suggests Boris Diaw who is a good shooter should take the open shot, while in fact, he gave up the opportunity. The possession ended up without scoring.

Figure 6.6 demonstrates some cases that discriminates ReLiable from its competitors. In 6.6a, Kobe Bryant shot over Kevin Durant who is bigger than him. Our model suggests that Kobe should have not taken the shot, while baseline methods suggest the opposite. It turns out that Kobe missed the shot. In 6.6b, Dirk Nowitzki noticed the shot clock is running out and his team is only leading by 2, so he decided to shoot the ball despite the double team on him. The output of our model is consistent with Dirk's actual behavior, while baseline methods suggest the opposite. In 6.6c, Gordon Hayward hit a buzzer beater at the end of the 3rd quarter. In the first case, ReLiable successfully avoids a bad shot. In the last two cases, ReLiable is able to figure the player should take the shot in order to hit a buzzer-beater, while comparative methods are not able to take the risk. By looking at these cases, we see that ReLiable is able to discover some decisions even better than the replay data, achieving the goal of offline reinforcement learning.

6.7 Conclusion & Future work

In this paper, we propose to study basketball tactical strategy learning with the absence of interaction with the environment. By formulating the basketball games under the context of the partially observable Markov decision process, we are able to apply a data-driven approach based on double Q-learning to derive nearly optimal strategy out of the raw game replay data. We then present ReLiable, an offline reinforcement learning framework for learning effective tactical strategies in basketball games. Experiments and case studies demonstrate the effectiveness of ReLiable over comparative methods, as well as the utility of using Transformer to parameterize the distribution over states under our POMDP setting.

In the future, it is of interest to extend our framework to offline multi-agent reinforcement learning (MARL)[BBD10] setting. Under the MARL setting, decisions are made in a finer granularity since each player should make sequential decisions instead of each team. Compared to single-agent scenarios, multi-agent mode is more challenging since it involves both collaboration and competition. Another direction worth exploring is extending the action space to a hierarchical setting.

CHAPTER 7

PlayBest: Professional Basketball Player Behavior Synthesis via Planning with Diffusion

7.1 Abstract

Dynamically planning in complex systems has been explored to improve decision-making in various domains. Professional basketball serves as a compelling example of a dynamic spatio-temporal game, encompassing context-dependent decision-making. However, processing the diverse on-court signals and navigating the vast space of potential actions and outcomes make it difficult for existing approaches to swiftly identify optimal strategies in response to evolving circumstances. In this study, we formulate the sequential decision-making process as a conditional trajectory generation process. Based on the formulation, we introduce PlayBest (PLAYer BEhavior SynThesis), a method to improve player decision-making. We extend the diffusion probabilistic model to learn challenging environmental dynamics from historical National Basketball Association (NBA) player motion tracking data. To incorporate data-driven strategies, an auxiliary value function is trained with corresponding rewards. To accomplish reward-guided trajectory generation, we condition the diffusion model on the value function via classifier-guided sampling. We validate the effectiveness of PlayBest through simulation studies, contrasting the generated trajectories with those employed by professional basketball teams. Our results reveal that the model excels at generating reasonable basketball trajectories that produce efficient plays. Moreover, the synthesized play strategies exhibit an alignment with professional tactics, highlighting the

model's capacity to capture the intricate dynamics of basketball games.[1]

7.2 Introduction

The exploration of dynamic systems and their planning has broad applicability across various domains. Whether it involves developing strategies for team sports [WFS18], managing traffic flow [ZYZ20], coordinating autonomous vehicles [KST21], or understanding the dynamics of financial markets [LXR22], these scenarios can be effectively framed as dynamic systems characterized by intricate interactions and decision-making processes. The ability to comprehend and plan within these systems becomes crucial to achieving optimal outcomes. Basketball, with its high level of dynamism and complexity as a team sport, serves as a captivating illustration of a real-time dynamic system with intricate tactical elements. A basketball game requires continuous adaptation and strategic decision-making. Coaches and players rely on pertinent environmental and behavioral cues including teammates' and opponents' current positions and trajectories to select play strategies that respond effectively to opponents' actions and adapt to real-time situational changes. Existing methods in sports analytics and trajectory optimization [WFS18, TF20, WSC22] have made progress in modeling and predicting player movements and game outcomes. However, these approaches struggle to capture the intricate dynamics of basketball games and produce flexible, adaptive play strategies that can handle the uncertainties and complexities inherent in the sport. The challenges arise from the following two features of basketball games:

Modeling the complex environmental dynamics: Capturing environmental dynamics in basketball games is a very challenging task due to the inherent complexity of the game, for example, rapid changes in game situations and numerous possible actions at any given moment. The spatio-temporal nature of basketball data, including multiple player positions and ball trajectories, further complicates the modeling process. The need for a computation-

[1]The code is at `https://github.com/xiusic/diffuser_bball`.

ally efficient and scalable approach to handle the massive amounts of data generated during basketball games presents a major challenge in modeling environmental dynamics.

Reward Sparsity: An additional challenge lies in addressing reward sparsity. Unlike other reinforcement learning (RL) environments where immediate feedback is readily available after each action, basketball games often see long sequences of actions leading up to a single reward event (e.g., the scoring of a basket). This results in a sparse reward landscape, as many actions contribute indirectly to the final outcome but are not themselves immediately rewarded. This scenario complicates the learning process as it becomes more challenging for the planning algorithm to accurately attribute the impact of individual actions to the final reward. Designing effective methods to address the reward sparsity challenge remains a significant hurdle in applying typical planning algorithms to basketball and similar sports games.

Recently, powerful trajectory optimizers that leverage learned models often produce plans that resemble adversarial examples rather than optimal trajectories [Tal14, KST19]. On the contrary, modern model-based RL algorithms tend to draw more from model-free approaches, such as value functions and policy gradients [WBC19], rather than utilizing the trajectory optimization toolbox. Methods that depend on online planning typically employ straightforward gradient-free trajectory optimization techniques like random shooting [NKF18] or the cross-entropy method [BKR13, CCM18] to circumvent the above problems.

In this work, we first formulate the planning problem in basketball as a multi-player behavior synthesis task, and instantiate the behavior synthesis task as a trajectory generation task. Following the recent success of generative models in applications of single-agent planning [JDT22, ADG22], we propose a novel application of the diffusion model called PlayBest (PLAYer BEhavior SynThesis), to generate optimal basketball trajectories and synthesize adaptive play strategies. Under most circumstances, the diffusion model serves as a generative model to capture the distribution of the input samples. In our study, we extend it as a powerful technique to enable flexible behavior synthesis in dynamic and uncertain environments since there is no existing online environment for basketball simulations. The diffusion process explores different potential trajectories and adapts to changes in the environment through the iterative sampling process to model basketball court dynamics. To guide the reverse diffusion process with rewards, PlayBest features a value guidance module that guides the diffusion model to generate optimal play trajectories by conditional sampling. This integration naturally forms a conditional generative process, and it allows PlayBest to swiftly adapt to evolving conditions and pinpoint optimal solutions in real-time.

We instantiate PlayBest in a variety of simulation studies and real-world scenarios, demonstrating the effectiveness of PlayBest in generating high-quality basketball trajectories that yield effective plays. Extensive results reveal that our proposed approach outperforms conventional planning methods in terms of adaptability, flexibility, and overall performance, showing a remarkable alignment with professional basketball tactics.

The core contributions of this work are summarized as follows:

- We attempt to formulate the basketball player behavior synthesis problem as a guided sampling/conditional generation of multiple players and ball trajectories from diffusion models.
- We present PlayBest, a framework featuring a diffusion probabilistic model with a value function, to instantiate the conditional generative model. We adapt the model to integrate multi-player behaviors and decisions in basketball and show that a number of desirable properties are obtained.
- We showcase the effectiveness of PlayBest via both quantitative and qualitative studies of the trajectories generated and validate the practicality of adopting PlayBest to investigate real basketball games.

7.3 Preliminary

7.3.1 Diffusion Probabilistic Models

Diffusion probabilistic models [SWM15, HJA20] stand out as a unique approach to learning complex data distributions, symbolized by $q(\boldsymbol{\tau})$, based on a collection of samples, denoted as $\mathcal{D} \coloneqq \{\boldsymbol{x}\}$.

On a high level, two processes are at the core of their operation: a predefined forward noising mechanism $q(\boldsymbol{\tau}^{i+1}|\boldsymbol{\tau}^{i}) \coloneqq \mathcal{N}(\boldsymbol{\tau}^{i+1}; \sqrt{\alpha_i}\boldsymbol{\tau}^{i},$
$(1-\alpha_i)\boldsymbol{I})$ and a trainable reverse or "denoising" process $p_\theta(\boldsymbol{\tau}^{i-1}|\boldsymbol{\tau}^{i})$
$\coloneqq \mathcal{N}(\boldsymbol{\tau}^{i-1}|\mu_\theta(\boldsymbol{\tau}^{i}, i), \Sigma_i)$. Here the Gaussian distribution is represented as $\mathcal{N}(\mu, \Sigma)$, and the variable α_i is pivital in determining the variance schedule. We begin with a sample $\boldsymbol{x}_0 \coloneqq \boldsymbol{x}$, followed by latents $\boldsymbol{\tau}^1, \boldsymbol{\tau}^2, ..., \boldsymbol{\tau}^{N-1}$, and culminate with $\boldsymbol{\tau}^N \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, factoring in specific values for α_i and an adequately extended N.

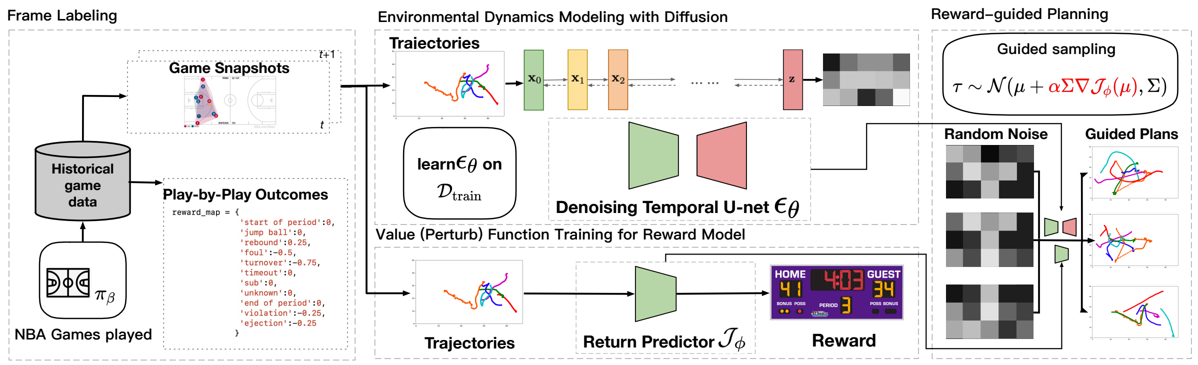
7.3.2 Trajectory Optimization Problem Setting in Basketball Strategy

In basketball, we can consider the game as a discrete-time system with dynamics $\mathbf{s}_{t+1} = \boldsymbol{f}(\mathbf{s}_t, \mathbf{a}_t)$, where $\mathbf{s}_t$ represents the state of the play, and $\mathbf{a}_t$ denotes the action or basketball maneuver. Trajectory optimization aims to find a sequence of actions $\mathbf{a}^*_{0:T}$ that maximizes an objective $\mathcal{J}$, such as maximizing the score. This can be represented as:

$$\mathbf{a}^*_{0:T} = \arg\max_{\mathbf{a}_{0:T}} \mathcal{J}(\mathbf{s}_0, \mathbf{a}_{0:T}) = \arg\max_{\mathbf{a}_{0:T}} \sum_{t=0}^{T} r(\mathbf{s}_t, \mathbf{a}_t) \tag{7.1}$$

where T defines the planning horizon. $\boldsymbol{\tau} = (\mathbf{s}_0, \mathbf{a}_0, \mathbf{s}_1, \mathbf{a}_1, \ldots, \mathbf{s}_T, \mathbf{a}_T)$ is the trajectory of states and actions, and $\mathcal{J}$ becomes the objective value of the play.

This model, when applied to basketball, facilitates the creation of dynamic strategies that adapt to real-time game scenarios. By simulating noise-corrupted play sequences and


Frame Labeling
Environmental Dynamics Modeling with Diffusion
Reward–guided Planning
Game Snapshots
$t+1$
t
Historical game data
π_β
NBA Games played
Play-by-Play Outcomes
Traiectories
$\mathbf{x}_0$
$\mathbf{x}_1$
$\mathbf{x}_2$
$\mathbf{z}$
learn ϵ_θ on $\mathcal{D}_{\text{train}}$
Denoising Temporal U-net ϵ_θ
Value (Perturb) Function Training for Reward Model
Trajectories
Return Predictor $\mathcal{J}_\phi$
Reward
HOME
GUEST
PERIOD
Guided sampling
$\tau \sim \mathcal{N}(\mu + \alpha\Sigma\nabla\mathcal{J}_\phi(\mu), \Sigma)$
Random Noise
Guided Plans

```
reward_map = {
                'start of period':0,
                'jump ball':0,
                'rebound':0.25,
                'foul':-0.5,
                'turnover':-0.75,
                'timeout':0,
                'sub':0,
                'unknown':0,
                'end of period':0,
                'violation':-0.25,
                'ejection':-0.25
              }
```

Figure 7.1: **Overview framework of PlayBest.** The overall pipeline can be split into four major components: Frame Labeling, Environmental Dynamics Learning, Value (Perturb) Function Training, and Trajectory Generation Guided by a Reward Function. The diffusion probabilistic model ϵ_θ is trained to model the environmental dynamics. The reward predictor $\mathcal{J}_\phi$ is trained on the same trajectories as the diffusion model. During guided trajectory generation, our model takes both environmental dynamics and rewards as input, performs guided planning via conditional sampling, and generates the trajectories as the guided plan.

iteratively denoising them, one can derive actionable insights into players' behaviors, leading to more effective in-game decision-making and planning.

7.3.3 Problem Description

The input for PlayBest consists of a set of basketball game records, denoted as $\mathcal{D}_{raw}$. These game records are composed of distinct elements, described as follows:

Motion Track Data. The motion track data, represented as $\mathcal{D}^{move}$, comprises static snapshots of in-game events, detailing the positions of all players and the ball at a rate of 25 frames per second. A game's progression can be reconstructed and visualized using these snapshots.

Play-by-Play Data. Denoted as $\mathcal{D}^{pbp}$, the play-by-play data offers a game transcript in the form of possessions. This data includes 1) the possession timestamp, 2) the player initiating the possession, 3) the result of the possession (e.g., points scored), and 4) additional unique identifiers employed for possession categorization.

To facilitate learning, we divide $\mathcal{D}_{raw}$ into $\mathcal{D}_{train}$ and $\mathcal{D}_{test}$, representing the training and testing sets, based on gameplay timestamps. We formally define our task as follows:

Given a set of game records $\mathcal{D}_{train} = \mathcal{D}^{move}_{train} \cup \mathcal{D}^{pbp}_{train}$ and a reward function $\mathcal{J}_\phi$, with $\mathcal{J}_\phi$ depending on the reward definition given by the discriminative rules applied to $\mathcal{D}^{pbp}_{train}$, the objective is to generate trajectories $\{\boldsymbol{\tau}\}$ leaning towards the higher-reward regions of the state-action space. In essence, our goal is to develop a policy $\pi_{\theta,\phi}(\mathbf{a} \mid \mathbf{s})$, parameterized by θ and ϕ, that determines the optimal action based on the states associated with each frame in $\mathcal{D}^{move}_{test}$.

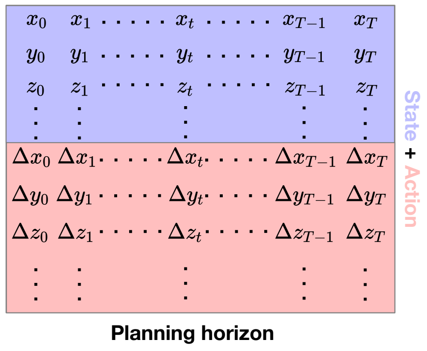
7.4 The PlayBest Framework

In this section, we describe in detail how our framework is designed. We first give an overview and then present details of the model architecture including the diffusion and value function modules.

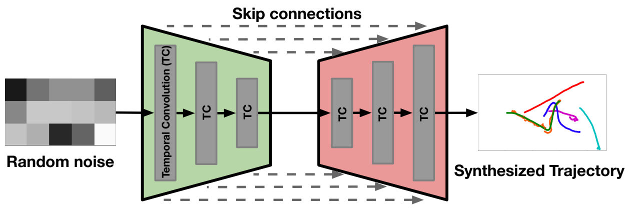
7.4.1 Framework Overview

Figure 7.1 depicts the PlayBest pipeline. The historical game replay data originates from actual games played during the 2015-2016 NBA regular season. Each team competes per their unknown policies π_β. The raw game data encompasses multiple modalities, and a game is characterized by a series of high-frequency snapshots (e.g., 25 frames per second). At any given time t, a snapshot includes an image displaying all player and ball positions, as well as additional metadata like the results of each possession (shot made/miss, free-throw made/miss, rebound, foul, turnover, etc), shot clock, and game clock at time t.

Out of the historical game replay data, we construct the player trajectories and ball trajectories to create the trajectory dataset $\mathcal{D}^{move}$. We then use the trajectory dataset $\mathcal{D}^{move}_{train}$ to train a diffusion model ϵ_θ that aims at modeling the distribution of the 3-dimensional player and ball movements. The training process of the diffusion model mimics the training procedure of what is usually referred to as offline RL, where there is no online environment to



(a) The shape of the training data. Trajectories are represented by the (x, y, z) coordinates of the ten on-court players across two teams and the ball (11 channels). The action is made up of the momentum of each object at the same timestep.



(b) The general structure of the diffusion model ϵ_θ is implemented by a U-net with temporal convolutional blocks, which have been widely utilized in image-centric diffusion models.

Figure 7.2: **(a, b) The input and diffusion architecture.**

interact with. However, the diffusion model by itself can only generate "like-real" trajectories that do not necessarily lead to a goal-specific outcome. To further generate trajectories that can represent "good plans", we train a value function that maps any possible trajectory to its expected return. During the sampling stage, the mean of the diffusion model is perturbed by the gradient of the value function. In this way, the guided sampling is capable of generating the trajectories biased towards the high-reward region. Incorporating a diffusion model in planning problems not only enhances efficient exploration and resilience in volatile environments, but also addresses the challenge of long-horizon planning, enabling the generation of strategic, noise-reduced trajectories over extended periods.

In essence, our framework utilizes a dataset $\mathcal{D}$ collected by an unknown behavior policy π_β, which can be approximated as the "average" policy for all NBA teams. This dataset is gathered once and remains unaltered during training. The training process relies entirely on the training set $\mathcal{D}_{\text{train}}$ and does not interact with the environment. Upon completion of training, we anticipate that π_θ will exhibit strong generalization on $\mathcal{D}_{\text{test}}$.

7.4.2 Environmental Dynamics Modeling with Diffusion

Since there is no public basketball environment that is able to provide online simulation, previous studies focus on offline simulations [CJJ22]. However, these approaches fall short in providing trajectories with planning strategies and efficiency due to the autoregressive designs, which are also challenging to be extended to incorporate dynamic planning. Therefore, we adopt diffusion models not only to simulate trajectories simultaneously from modeling environmental dynamics but also to be guided by the specific outcomes with conditional sampling.

Model Input and Output. To represent our input that can be consumed by the diffusion model, we represent all the trajectories in the format of a 2-dimensional image as described in Figure 7.2a. To be specific, we concatenate the state features and action features at each timestep in the game together to form one column of the model input. The features from different timesteps are then stacked together following the temporal order to form the rows. In other words, the rows in the model input correspond to the *planning horizon* T in Section 7.3.2.

Architecture. As illustrated in Figure 7.2b, the backbone of the environmental dynamics modeling module is a diffusion probabilistic model ϵ_θ. Diffusion models have been found effective in fitting the distribution of images [HJA20]. Our assumption is that the diffusion models can also learn the underlying distribution of basketball player trajectories by framing as the trajectory optimization problem, thereby modeling the player and ball dynamics. Following image-based diffusion models, we adopt the U-net architecture [RFB15] as the overall architecture. Moreover, to account for the temporal dependencies between different timesteps of the trajectories, we replace two-dimensional spatial convolutions with one-dimensional temporal convolutions.

Diffusion Training. We follow the usual way by parameterizing the Gaussian noise term to make it predict ϵ_t from the input x_t at diffusion step t to learn the parameters θ,:

$$\mathcal{L}(\theta) = \mathbb{E}_{t,\epsilon_t,\boldsymbol{\tau}^0}\left[\|\epsilon_t - \epsilon_\theta(\boldsymbol{\tau}^t, t)\|^2\right], \tag{7.2}$$

where $\epsilon_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ denotes the noise target, t represents the diffusion step, and $\boldsymbol{\tau}^t$ is the trajectory $\boldsymbol{\tau}^0$ corrupted by noise ϵ at diffusion step t.

7.4.3 Value Function Training for Reward Model

At the heart of the value function is an encoder that takes the trajectory data as input and returns the estimated cumulative reward. The structure of the return predictor $\mathcal{J}_\phi$ takes exactly the first half of the U-Net employed in the diffusion model, and it is followed by a linear layer that generates a single scalar output indicating the reward value.

7.4.4 Guided Planning as Conditional Sampling

Existing studies [JDT22, ADG22] have revealed the connections between classifier-guided / classifier-free sampling [DN21] and reinforcement learning. The sampling routine of PlayBest resembles the classifier-guided sampling. In detail, we condition a diffusion model $p_\theta(\boldsymbol{\tau})$ on the states and actions encompassed within the entirety of the trajectory data. Following this, we develop an isolated model, $\mathcal{J}_\phi$, with the aim of forecasting the aggregated rewards of trajectory instances $\boldsymbol{\tau}^i$. The trajectory sampling operation is directed by the gradients of $\mathcal{J}_\phi$, which adjust the means μ of the reverse process as per the following equations:

$$\begin{aligned}
&\mu \leftarrow \mu_\theta\left(\boldsymbol{\tau}^i\right), \\
&\boldsymbol{\tau}^{i-1} \sim \mathcal{N}\left(\mu + \alpha\Sigma\nabla\mathcal{J}_\phi(\mu), \Sigma^i\right), \\
&\boldsymbol{\tau}^{i-1}_{\boldsymbol{s0}} \leftarrow \boldsymbol{s},
\end{aligned} \tag{7.3}$$

# Training Games	# Minutes	# Plays	# Frames
480	23, 040	210, 952	34, 560, 000
# Testing Games	**# Minutes**	**# Plays**	**# Frames**
151	7, 248	68, 701	10, 872, 000
# Games	**# Minutes**	**# Plays**	**# Frames**
631	30, 288	279, 653	45, 432, 000

Table 7.1: **NBA 2015 - 16 Regular Season Game Stats**. Games are split chronically so that all the games in the test set are after any game in the training set.

Event type	Reward
"start of period"	0
"jump ball"	0
"rebound"	0.25
"foul"	-0.25
"turnover"	-1
"timeout"	0
"substitution"	0
"end of period"	0
"violation"	-0.25
"3 pointer made"	3
"2 pointer made"	2
"free-throw made"	1

Table 7.2: **Definition of Reward per possession.**

where α is the scaling factor to measure the impact of the guidance on the sampling, and

$$\nabla \mathcal{J}(\mu) = \sum_{t=0}^{T} \nabla_{\mathbf{s}_t, \mathbf{a}_t} r\left(\mathbf{s}_t, \mathbf{a}_t\right)\Big|_{(\mathbf{s}_t, \mathbf{a}_t) = \mu_t} . \tag{7.4}$$

where r is the reward function given by the environment. In our case, it comes from the outcome of the possessions derived from $\mathcal{D}^{pbp}$. The detailed algorithm of reward-guided planning is illustrated in Algorithm 2.

7.5 Experiments

7.5.1 Experimental Setup

To quantitatively evaluate the effectiveness of player behavior planning, we focus on measuring the cumulative return given by the learned policy, which serves as an objective evaluation metric to compare the performance of PlayBest with other comparative methods. Evaluating offline RL is inherently difficult as it lacks real-time environment interaction for reward accumulation. Thus the model verification is primarily reliant on utilizing existing

Algorithm 2 Reward Guided Planning

```
Require diffusion model $\mu_\theta$, guide $\mathcal{J}_\phi$, scale $\alpha$, covariances $\Sigma^i$
while not done do
    Acquire state $\mathbf{s}$; initialize trajectory $\boldsymbol{\tau}^N \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$

    //$N$ diffusion steps in total

    for $i = N, \ldots, 1$ do
        $\mu \leftarrow \mu_\theta(\boldsymbol{\tau}^i)$

        $\boldsymbol{\tau}^{i-1} \sim \mathcal{N}(\mu + \alpha\Sigma\nabla\mathcal{J}(\mu), \Sigma^i)$

        //conditioned on the initial player positions
        $\boldsymbol{\tau}^{i-1}_{\boldsymbol{s}_0} \leftarrow \boldsymbol{s}$

    end for
    Execute first action of trajectory $\boldsymbol{\tau}^0_{\mathbf{a}_0}$
end while
```

replay data. To validate the capacity of our framework in learning efficient tactics, we assess PlayBest's ability to generate efficient plans using diverse data of varying standards.

Dataset. We obtained our data from an open-source repository [spo16, pbp16]. The model's input data is a combination of two components: (1) **Player Movement Sensor Data**: This component captures real-time court events, detailing the positions of the players and the ball in Cartesian coordinates. The sampling frequency of this data is 25 frames per second. The statistics are detailed in Table 7.1. (2) **Play-by-Play**: This segment of information contains the specifics of each possession, such as the termination of the possession (whether through a jump shot, layup, foul, and so forth), the points gained by the offensive team, the location from which the ball was shot, and the player who made the shot, among other details. The data for training and testing is split chronologically: the training set includes games from 2015, amounting to 480 games, while the remaining games from 2016 form the testing set, amounting to 151 games. The statistics are described in Table 7.1.

Reward Definition. As there is no fine-grained reward design in basketball in previous work, e.g., [YSK22, CJJ22], we define the reward of each possession based on its outcomes,

Methods	Random Walk	Ground Truth	BCQ	CQL	IQL	PlayBest
AVG	-9.1172±0.035	0.0448±0.000	0.0964±0.000	0.0986±0.001	0.0992±0.000	**0.4473±1.235**
MAX	-9.0753	0.0448	0.0967	0.0995	0.0992	**2.2707**

Table 7.3: **Overall performance in return values per possession.**

as listed in Table 7.2. For a certain team that plays the possession, we encourage the possession trajectory if it leads to positive outcomes (e.g., score, rebound) and we punish otherwise (turnover, foul, violation). Note that the same event by the opponent team takes the negative value of the rewards. For example, a 2-point basket made by the team on offense leads to a -2 reward to the training sample of the value function for the team on defense. During our offline evaluation, we employ our value function $\mathcal{J}_\phi$ to gauge the expected return of our policy. By summing all expected rewards from each possession for a team, we can approximate the total points for the team following the learned strategic policies. For each game in the test set, all comparative methods plan trajectories from each possession's actual initial state.

Baselines. As this task has yet to be explored, there are no widely adopted baselines for direct comparison. Therefore, we examine our model with several state-of-the-art offline RL algorithms and a naive baseline to verify its effectiveness:

- Batch-Constrained deep Q-learning (**BCQ**) [FMP19] is an off-policy algorithm for offline RL. It mitigates overestimation bias by constraining the policy to actions similar to the behavior policy, ensuring a more conservative policy.
- Conservative Q-Learning (**CQL**) [KZT20] is an offline RL approach that minimizes an upper bound of the expected policy value to conservatively estimate the action-value function, leading to a more reliable policy.
- Independent Q-Learning (**IQL**) [KNL21] is a multi-agent reinforcement learning approach where each agent learns its own Q-function independently. It offers an efficient solution for multi-agent environments.
- **Random Walk** is the "naive" baseline that can be used to validate the correctness of the

value function and to offer an auxiliary comparative method that corresponds to the case where all the players navigate randomly within the range of the court.

7.5.2 Implementation Details

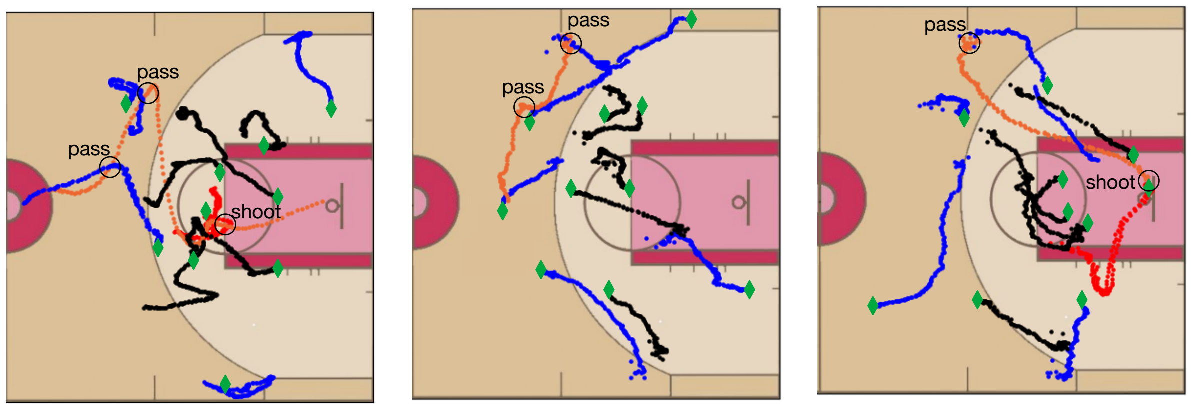
We set the planning horizon length to $1,024$ so that all trajectories in the training data can be fitted in our diffusion model. The diffusion step is set to 20 in all experiments. The learning rate is 2×10^{-5} without learning rate scheduling. The hidden dimension is set following [JDT22]. The training batch size is set to 512. We train all models for $245K$ training steps. The value function is optimized with the mean square error loss. All experiments are run on the NVIDIA Tesla V100 Tensor Core GPUs with 16GB memory.

7.5.3 Overall Performance

Table 7.3 shows the cumulative scores of the generated trajectories of the compared methods. For all the models, we run each 5 times and report the average performance with the corresponding variance. We observe that: (1) PlayBest consistently and significantly outperforms the baselines and the historical gameplay in generating trajectories with higher rewards. (2) The dedicated offline RL baselines CQL and IQL are also able to learn from historical replays with mixed rewards. However, they perform noticeably worse than PlayBest, indicating that the diffusion model in PlayBest better captures the intrinsic dynamics of basketball gameplay. (3) As expected, the random walk baseline performs poorly, further highlighting the effectiveness of the value function in distinguishing between superior and inferior planning trajectories. These observations suggest that the diffusion model is a powerful tool of modeling complex environmental dynamics and, when combined with guided sampling, becomes a strong planning tool.

α	**0**	**0.01**	**0.1**	**1**	**10**
$\boldsymbol{AVG}$	0.0859±0.0052	0.0894±1.2263	0.4473±1.2349	3.0870±1.4955	10.8090±2.4050
$\boldsymbol{MAX}$	0.0932	1.8844	2.2707	5.3534	14.2389

Table 7.4: **The effects of the scaling factor α.** We repeat our sampling process 5 times and report the mean and variance for the average returns per possession.



(a) Reward: 2.194 (b) Reward: 0.864 (c) Reward: 1.541

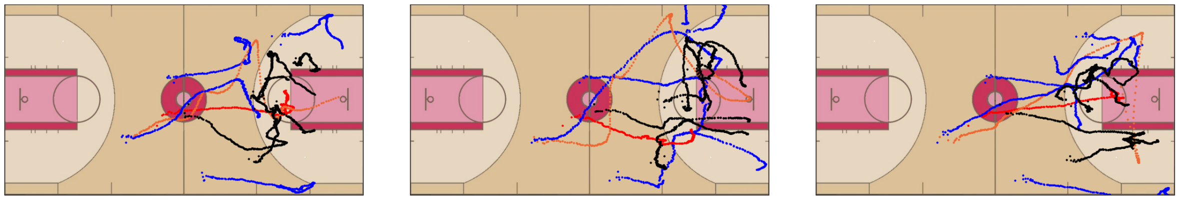
Figure 7.3: **(a, b, c): Sampled cases of possessions generated by PlayBest.** PlayBest learns strategies deviating from existing data yet still aligning with subjective expectations for effective basketball play. The **blue** team is on offense and moves towards the right basket, while the **black** team is on defense. The ball is marked in orange. The player who scores for the **blue** team is highlighted in Red (no shot attempt in (b)). Diamonds(♦) are final positions of the players. More details are in Section 7.5.5.

7.5.4 Analysis

Table 7.4 demonstrates the overall return evaluated on all the trajectories generated by PlayBest with α being $\{0, 0.01, 0.1, 1.0, 10.0\}$. It is noted that $\alpha = 0$ indicates PlayBest performing unconditional sampling without the perturbation of the gradient of the value function.

7.5.4.1 Hyperparameter Study

When the diffusion model performs conditional sampling for trajectories, the scaling factor α serves as a balance between quantitative scores and interpretability. With the increase



(a) $\alpha = 0.1$ (b) $\alpha = 1.0$ (c) $\alpha = 10.0$

Figure 7.4: **(a, b, c): Possessions generated by PlayBest with different α.**

of α, the value guidance generally has a larger impact and improves the overall cumulative rewards on the test games. Then the question becomes, *why not keep increasing the value of α?* To provide a deeper insight into this, we conduct a comparative study demonstrated in Figure 7.4. We consider trajectories initiated from the same state but with different scaling factors, specifically α values of 0.1, 1.0, and 10.0. By visualizing these trajectories, we aim to demonstrate how variations in the scaling factor can significantly influence the progression and outcomes of the game, further emphasizing the crucial role of this parameter in our model. When $\alpha = 1.0$, there seems to be a mysterious force that pulls the ball to the basket. In the $\alpha = 10.0$ case, the synthesized trajectory becomes even less interpretable since the ball never goes through the basket. In both $\alpha = 1.0$ and $\alpha = 10.0$ cases, the ball exhibits behaviors that defy the laws of physics, seemingly being propelled towards the basket as if being controlled by an invisible player.

7.5.4.2 Ablation Study

The full PlayBest model with sufficient value guidance outperforms the ablation version (i.e., $\alpha = 0$), indicating the necessity of the value guidance. By mere unconditional sampling, the ablation version is already able to generate on average better plans than the ground truth plays in the test set. These observations confirm our two claims: The value-based guided sampling directs the diffusion model to generate trajectories leaning towards the higher-reward regions of the state-action space; and the diffusion model on its own can generate coherent and realistic trajectories representing a competent game plan.

length m	**25**	**50**	**75**	**100**
man-to-man	1.410 ± 0.368	1.750 ± 0.059	2.526 ± 0.039	2.814 ± 0.008
2-3 zone	1.424 ± 0.284	1.558 ± 0.309	2.229 ± 0.011	2.327 ± 0.029

Table 7.5: Return values competing against defense.

7.5.4.3 The adversary of the game

Notably, basketball games and many other team sports are adversarial. We implemented additional defensive strategies including man-to-man and 2-3 zone defense, and ran the learned policy against these strategies *iteratively* to add adversaries. In each iteration, PlayBest samples a trajectory of length m, and we replace the trajectories corresponding to *defensive* players (5 channels) with those generated with man-to-man or 2-3 zone defense. The trajectories on the defensive side act as adversarial agents competing against the diffusion policy. The results are reported in Table 7.5. We observe that: (1) The offensive strategy encoded in PlayBest outplays the man-to-man defense and 2-3 zone defense. (2) When increasing the length of the segment of the trajectory, PlayBest is more likely to generate more coherent trajectories, leading to better returns when faced with the same defense.

7.5.5 Case Study

We now perform a case study to qualitatively demonstrate the practicability of value-guided conditional generation. Figure 7.3 shows three cases, all of which are sampled from the trajectories generated by PlayBest. In Figure 7.3a, we visualize a possession generated with a high reward. The players in the blue team share the ball well and managed to find the red player near the free-throw line. At the time the red player shoots the ball, no defender is between him and the basket. The outcome of this simulated play is a 2-point basket. In Figures 7.3b and 7.3c, two different plans with the same horizon are generated by PlayBest given the same initial player and ball positions. In Figure 7.3b, we observe a more conservative strategy where the ball is repeatedly passed between the blue players near the perimeter, which is also valued with a lower reward. In spite of the same initial conditions, PlayBest generates a more aggressive strategy in Figure 7.3c in that the ball is passed directly to the low post that leads to a 2-point basket, suggesting an aggressive tactic execution. These cases illustrate that PlayBest is able to not only synthesize realistic trajectories but also output high-reward and diverse trajectories for planning tactics as well as for enhancing decision-making.

7.6 Related Work

Reinforcement Learning for Planning. Reinforcement learning is a learning-based control approach. A wide range of application domains have seen remarkable achievements through the use of reinforcement learning algorithms, such as robotics [KIP18a], autonomous vehicles [BMG19], industrial regulation [GGG18], financial sectors [MK19], healthcare [YLN19], gaming [SSS17], and marketing [JSL18]. Despite its wide use, many RL applications depend on an online environment that facilitates interactions. In numerous circumstances, acquiring data online is either expensive, unethical, or dangerous, making it a luxury. Consequently, it is preferable to learn effective behavior strategies using only pre-existing data. Offline RL has been suggested to fully utilize previously gathered data without the need for environmental interaction [FMP19, ASN20, KZT20, LKT20, FKN20], which has found applications in areas such as dialogue systems [JGS19], robotic manipulation techniques [KIP18b], and navigation [KAL21].

Sports & Machine Learning. Machine learning and AI have recently been employed in sports analytics to comprehend and advise human decision-making [AAM17, RPW17, DVD18, SDS20, TOM21, RVD21, WCP22]. [LSP21] suggested a player ranking technique that combines inverse RL and Q-learning. [WCP22] proposed a deep-learning model composed of a novel short-term extractor and a long-term encoder for capturing a shot-by-shot sequence. [WSC22] developed a position-aware fusion framework for objectively forecasting stroke returns based on rally progress and player style. [CWP22] predicted returning strokes and player movements based on previous strokes using a dynamic graph and hierarchical fusion approach. While these methods are effective for producing simulations, they may not fully address the goal of maximizing specific objectives (e.g., winning games). Previous basketball analytics mainly focused on employing recurrent neural networks to analyze player-tracking data for offensive tactics identification and player movement prediction [MBG16, WZ16, TDC20, TF20]. However, these methods lack labeled interactions between the learning agent and the environment, limiting their ability to uncover optimal decision sequences. Wang et al. [WFS18] explored the use of RL to improve defensive team decisions, especially the execution of a "double team" strategy. Liu et al. [LH18] designed a method using motion capture data to learn robust basketball dribbling maneuvers by training on both locomotion and arm control, achieving robust performance in various scenarios.

7.7 Conclusion

In this paper, we introduce PlayBest, the diffusion model with conditional sampling in planning high-rewarded basketball trajectories and synthesizing adaptive play strategies. With the extension of environmental dynamics into the diffusion model and fine-grained rewards for the value function, PlayBest has shown impressive capabilities in capturing the intricate dynamics of basketball games and generating play strategies that are consistent with or even surpass professional tactics. Its adaptive nature has allowed for swift adjustments to evolving conditions and facilitated real-time identification of optimal solutions. Extensive simulation studies and analysis of real-world NBA data have confirmed the advantages of PlayBest over traditional planning methods. The generated trajectories and play strategies not only outperform conventional techniques but also exhibit a high level of alignment with professional basketball tactics. Future work will explore the integration of additional sources of information, such as player fatigue and skill levels, into our framework to further enhance its performance. In addition, we plan to develop an open environment and a set of benchmarks to not only facilitate research on machine learning for sports but also extend to other real-time dynamic systems.

7.8 Limitation

Currently we only consider the player movement and only conduct offline evaluation since no online environment for our application is available. Future work will explore the integration of additional sources of information, such as player fatigue and skill levels, into our framework to further enhance its performance. Moreover, we plan to extend the application of PlayBest to other team sports/e-sports, investigating its efficacy in generating adaptive play strategies and trajectories in various dynamic and uncertain environments. Finally, we plan to develop an open environment and a set of benchmarks to not only facilitate research on machine learning for sports but also extend to other real-time dynamic systems.

CHAPTER 8

Conclusion

Large foundation models such as LLMs and diffusion models have demonstrated their potential to accelerate human production and innovation by driving autonomous AI agents. However, existing foundation models still come with drawbacks. Among various limitations, this thesis identifies trustworthiness, efficiency, and planning ability as the major challenges to work on. Through developing and applying data-centric knowledge-enhanced reasoning, data-driven constitutional alignment, and reward-guided generative planning algorithms, this thesis presents a series of efforts to build a more reliable and unbiased AI agent that can assist humans in understanding and planning complex tasks in various domains through the design and implementation of the moral reasoning module, alignment module, and planning module.

8.1 Future Research Agenda

The future research plan is to stick to the ultimate goal of building AI agents that are reliable, unbiased, and capable of planning safely and effectively in various domains, such as science and engineering.

8.1.1 Building More Capable AI Agents

The development of more capable LLMs has led to the creation of LLM-based agent frameworks such as AutoGPT [1] and AutoGen [WBZ]. These frameworks demonstrate the ability of LLMs to tackle complex problems through research and reasoning in multiple steps, demonstrating remarkable diligence and intelligence in navigating challenges. However, a significant limitation shared between these agents is their lack of learning capabilities. Despite their abilities, when faced with similar problems repeatedly, these agents do not take advantage of past experiences. Consequently, they must undergo the same laborious process to tackle the problems as if they were encountering them for the first time. This observation introduces an interesting concept: the idea that agents can record their successful strategies and recall them when encountering similar situations in the future. This capability would enable these agents to make decisions more swiftly and efficiently, eliminating the necessity to navigate through the same extensive and complex reasoning processes again.

8.1.2 Aligning AI Decisions with Human Experts

As aforementioned, current foundation models are capable of conducting effective decision-making in the form of planning. A more challenging setting is when complex and rapid decision-making is required in dynamic situations where no single correct answer exists. For instance, two experienced military leaders or medical experts may choose different tactical actions or triage decisions when confronted with the same battlefield scenario and challenging options. As AI systems evolve to collaborate more closely with humans, fostering appropriate human trust in the AI's decision-making abilities is crucial. Capturing the essential characteristics of expert human decision-making in dynamic environments and representing that data computationally in algorithmic decision-makers could be a critical component in ensuring that algorithms make reliable choices in challenging circumstances.

[1]https://news.agpt.co/

8.1.3 Enabling Lifespan Learning of AI Agents

Although significant efforts have been made to enhance the human-like qualities of large language models, there has been a notable absence of thorough discussion on current methods and efforts aimed at creating ways that can self-update in response to a rapidly evolving environment. These systems should be capable of assimilating information from their surroundings and producing responses that integrate both recent inputs and past experiences, similar to human cognitive processes. Four implementations have been widely adopted, including Retrieval-Augmented Generation with Knowledge Bases, Long Context Methods, External Memory Utilization, and Continual Learning. The four instances each stand out in different use cases. However, building a unified agent requires a fully principled integration of these implementations, and how to integrate remains challenging yet exciting.

REFERENCES

[AAM17] Raquel YS Aoki, Renato M Assuncao, and Pedro OS Vaz de Melo. “Luck is hard to beat: The difficulty of sports prediction.” In *KDD*, pp. 1367–1376, 2017.

[AAP19] Chris Alberti, Daniel Andor, Emily Pitler, Jacob Devlin, and Michael Collins. “Synthetic QA Corpora Generation with Roundtrip Consistency.” In *ACL’19*, pp. 6168–6173, 2019.

[ABC21] Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Ben Mann, Nova DasSarma, et al. “A general language assistant as a laboratory for alignment.” *arXiv preprint arXiv:2112.00861*, 2021.

[AC75] Alfred V Aho and Margaret J Corasick. “Efficient string matching: an aid to bibliographic search.” *Communications of the ACM*, **18**(6):333–340, 1975.

[ACG20] Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. “Do not have enough data? Deep learning to the rescue!” In *AAAI’20*, pp. 7383–7390, 2020.

[ADG22] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal. “Is Conditional Generative Modeling all you need for Decision-Making?” *arXiv preprint arXiv:2211.15657*, 2022.

[AL78] Robert B Allan and Renu Laskar. “On domination and independent domination numbers of a graph.” *Discrete mathematics*, **23**(2):73–76, 1978.

[APK03] Baher Abdulhai, Rob Pringle, and Grigoris J Karakoulas. “Reinforcement learning for true adaptive traffic signal control.” *Journal of Transportation Engineering*, **129**(3):278–285, 2003.

[ASD21] Haytham Assem, Rajdeep Sarkar, and Sourav Dutta. “QASAR: self-supervised learning framework for extractive question answering.” In *2021 IEEE International Conference on Big Data (Big Data)*, pp. 1797–1808. IEEE, 2021.

[ASN20] Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. “An optimistic perspective on offline reinforcement learning.” In *International Conference on Machine Learning*, pp. 104–114. PMLR, 2020.

[Ast65] Karl Johan Åström. “Optimal control of Markov processes with incomplete state information.” *JMAA*, **10**(1):174–205, 1965.

[BBD10] Lucian Buşoniu, Robert Babuška, and Bart De Schutter. “Multi-agent reinforcement learning: An overview.” *Innovations in multi-agent systems and applications-1*, pp. 183–221, 2010.

[BCL23] Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, Quyet V. Do, Yan Xu, and Pascale Fung. "A Multitask, Multilingual, Multimodal Evaluation of ChatGPT on Reasoning, Hallucination, and Interactivity." In *ACL*, 2023.

[Bel66] Richard Bellman. "Dynamic programming." *Science*, **153**(3731):34–37, 1966.

[BKK22] Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. "Constitutional ai: Harmlessness from ai feedback." *arXiv preprint arXiv:2212.08073*, 2022.

[BKR13] Zdravko I Botev, Dirk P Kroese, Reuven Y Rubinstein, and Pierre L'Ecuyer. "The cross-entropy method for optimization." In *Handbook of statistics*, volume 31, pp. 35–59. Elsevier, 2013.

[BLC19] Iz Beltagy, Kyle Lo, and Arman Cohan. "SciBERT: A Pretrained Language Model for Scientific Text." In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3615–3620, 2019.

[BMG19] Bharathan Balaji, Sunil Mallya, Sahika Genc, Saurabh Gupta, Leo Dirac, Vineet Khare, Gourav Roy, Tao Sun, Yunzhe Tao, Brian Townsend, et al. "Deepracer: Educational autonomous racing platform for experimentation with sim2real reinforcement learning." *arXiv preprint arXiv:1911.01562*, 2019.

[BMR20] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. "Language models are few-shot learners." In *NeurIPS*, 2020.

[BP23] Rishabh Bhardwaj and Soujanya Poria. "Red-teaming large language models using chain of utterances for safety-alignment." *arXiv preprint arXiv:2308.09662*, 2023.

[BPH24] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. "Video generation models as world simulators." 2024.

[BSA18] Mikhail Burtsev, Alexander Seliverstov, Rafael Airapetyan, Mikhail Arkhipov, Dilyara Baymurzina, Nickolay Bushkov, Olga Gureenkova, Taras Khakhulin, Yurii Kuratov, Denis Kuznetsov, et al. "Deeppavlov: Open-source library for dialogue systems." In *ACL'18, System Demonstrations*, pp. 122–127, 2018.

[BZL20] Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. "PIQA: Reasoning about Physical Commonsense in Natural Language." In *AAAI*, 2020.

[CCE18] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. "Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge." *arXiv preprint arXiv:1803.05457*, 2018.

[CCM18] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. "Deep reinforcement learning in a handful of trials using probabilistic dynamics models." *Advances in neural information processing systems*, **31**, 2018.

[CH19] Jang Hyun Cho and Bharath Hariharan. "On the efficacy of knowledge distillation." In *ICCV*, 2019.

[CHD22] Ganqu Cui, Shengding Hu, Ning Ding, Longtao Huang, and Zhiyuan Liu. "Prototypical Verbalizer for Prompt-based Few-shot Tuning." In *ACL'22*, pp. 7014–7024, 2022.

[CHL22] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. "Scaling Instruction-Finetuned Language Models." *arXiv preprint arXiv:2210.11416*, 2022.

[CJC24a] Xiusi Chen, Jyun-Yu Jiang, Wei-Cheng Chang, Cho-Jui Hsieh, Hsiang-Fu Yu, and Wei Wang. "MinPrompt: Graph-based minimal prompt data augmentation for few-shot question answering." In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, 2024.

[CJC24b] Xiusi Chen, Jyun-Yu Jiang, Wei-Cheng Chang, Cho-Jui Hsieh, Hsiang-Fu Yu, and Wei Wang. "MinPrompt: Graph-based Minimal Prompt Data Augmentation for Few-shot Question Answering." In *ACL*, 2024.

[CJJ22] Xiusi Chen, Jyun-Yu Jiang, Kun Jin, Yichao Zhou, Mingyan Liu, P Jeffrey Brantingham, and Wei Wang. "ReLiable: Offline Reinforcement Learning for Tactical Strategies in Professional Basketball Games." In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pp. 3023–3032, 2022.

[CJW22] Xiusi Chen, Jyun-Yu Jiang, and Wei Wang. "Scalable Graph Representation Learning via Locality-Sensitive Hashing." In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pp. 3878–3882, 2022.

[CLL23] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. "Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality.", March 2023.

[CN21] Rakesh Chada and Pradeep Natarajan. “FewshotQA: A simple framework for few-shot learning of question answering tasks using pre-trained text-to-text models.” In *EMNLP’21*, pp. 6081–6090, 2021.

[CND23] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. “PALM: Scaling Language Modeling with Pathways.” *Journal of Machine Learning Research*, 2023.

[CRL18] Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. “e-SNLI: Natural Language Inference with Natural Language Explanations.” In *NeurIPS*, 2018.

[CVG14] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. “Learning phrase representations using RNN encoder-decoder for statistical machine translation.” *arXiv preprint arXiv:1406.1078*, 2014.

[CWH24] Xiusi Chen, Wei-Yao Wang, Ziniu Hu, Curtis Chou, Lam Hoang, Kun Jin, Mingyan Liu, P Jeffrey Brantingham, and Wei Wang. “PlayBest: Professional Basketball Player Behavior Synthesis via Planning with Diffusion.” In *Proceedings of the 33rd ACM International Conference on Information & Knowledge Management*, 2024.

[CWN24] Xiusi Chen, Hongzhi Wen, Sreyashi Nag, Chen Luo, Qingyu Yin, Ruirui Li, Zheng Li, and Wei Wang. “ITERALIGN: Iterative constitutional alignment of large language models.” *arXiv preprint arXiv:2403.18341*, 2024.

[CWP22] Kai-Shiang Chang, Wei-Yao Wang, and Wen-Chih Peng. “Where Will Players Move Next? Dynamic Graphs and Hierarchical Fusion for Movement Forecasting in Badminton.” *arXiv preprint arXiv:2211.12217*, 2022.

[CWQ23] Hongzhan Chen, Siyue Wu, Xiaojun Quan, Rui Wang, Ming Yan, and Ji Zhang. “MCC-KD: Multi-CoT Consistent Knowledge Distillation.” In *EMNLP Findings*, 2023.

[CWZ19] Yu Chen, Lingfei Wu, and Mohammed J Zaki. “Reinforcement learning based graph-to-sequence model for natural question generation.” *arXiv preprint arXiv:1908.04942*, 2019.

[CZD23] Xiusi Chen, Yu Zhang, Jinliang Deng, Jyun-Yu Jiang, and Wei Wang. “Gotta: generative few-shot question answering by prompt-based cloze data augmentation.” In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*, pp. 909–917. SIAM, 2023.

[CZW21] Jianyu Cai, Zhanqiu Zhang, Feng Wu, and Jie Wang. "Deep Cognitive Reasoning Network for Multi-hop Question Answering over Knowledge Graphs." In *Findings of ACL'21*, pp. 219–229, 2021.

[DBV19] Tom Decroos, Lotte Bransen, Jan Van Haaren, and Jesse Davis. "Actions speak louder than goals: Valuing player actions in soccer." In *KDD*, pp. 1851–1861, 2019.

[DCL18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding." *arXiv preprint arXiv:1810.04805*, 2018.

[DCL19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." In *NAACL'19*, pp. 4171–4186, 2019.

[DN21] Prafulla Dhariwal and Alexander Nichol. "Diffusion models beat gans on image synthesis." *Advances in neural information processing systems*, **34**:8780–8794, 2021.

[DSH17] Matthew Dunn, Levent Sagun, Mike Higgins, V Ugur Guney, Volkan Cirik, and Kyunghyun Cho. "Searchqa: A new q&a dataset augmented with context from a search engine." *arXiv preprint arXiv:1704.05179*, 2017.

[DVD18] Tom Decroos, Jan Van Haaren, and Jesse Davis. "Automatic discovery of tactics in spatio-temporal soccer match data." In *KDD*, pp. 223–232, 2018.

[DYY23] Dorottya Demszky, Diyi Yang, David S. Yeager, Christopher J. Bryan, Margarett Clapper, Susannah Chandhok, Johannes C. Eichstaedt, Cameron Hecht, Jeremy Jamieson, Meghann Johnson, Michaela Jones, Danielle Krettek-Cobb, Leslie Lai, Nirel JonesMitchell, Desmond C. Ong, Carol S. Dweck, James J. Gross, and James W. Pennebaker. "Using Large Language Models in Psychology." *Nature Reviews Psychology*, **2**:688–701, 2023.

[DZJ23] Xuan Long Do, Bowei Zou, Shafiq Joty, Anh Tai Tran, Liangming Pan, Nancy F Chen, and Ai Ti Aw. "Modeling What-to-ask and How-to-ask for Answer-unaware Conversational Question Generation." *arXiv preprint arXiv:2305.03088*, 2023.

[EAB22] Jacob Eisenstein, Daniel Andor, Bernd Bohnet, Michael Collins, and David Mimno. "Honest Students from Untrusted Teachers: Learning an Interpretable Question-Answering Pipeline from a Pretrained Language Model." *arXiv preprint arXiv:2210.02498*, 2022.

[EKP21] Andre Esteva, Anuprit Kale, Romain Paulus, Kazuma Hashimoto, Wenpeng Yin, Dragomir Radev, and Richard Socher. "COVID-19 information retrieval with deep-learning based semantic search, question answering, and abstractive summarization." *NPJ digital medicine*, **4**(1):1–9, 2021.

[FHM18] Scott Fujimoto, Herke Hoof, and David Meger. "Addressing function approximation error in actor-critic methods." In *ICML*, pp. 1587–1596, 2018.

[FKN20] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. "D4rl: Datasets for deep data-driven reinforcement learning." *arXiv preprint arXiv:2004.07219*, 2020.

[FMP19] Scott Fujimoto, David Meger, and Doina Precup. "Off-policy deep reinforcement learning without exploration." In *International conference on machine learning*, pp. 2052–2062. PMLR, 2019.

[FTJ19] Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. "MRQA 2019 Shared Task: Evaluating Generalization in Reading Comprehension." In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pp. 1–13, 2019.

[GFC21] Tianyu Gao, Adam Fisch, and Danqi Chen. "Making Pre-trained Language Models Better Few-shot Learners." In *ACL'21*, pp. 3816–3830, 2021.

[GGG18] A Gasparik, C Gamble, and J Gao. "Safety-first ai for autonomous data centre cooling and industrial control." *DeepMind Blog*, 2018.

[GJK19] Omer Gottesman, Fredrik Johansson, Matthieu Komorowski, Aldo Faisal, David Sontag, Finale Doshi-Velez, and Leo Anthony Celi. "Guidelines for reinforcement learning in healthcare." *Nature medicine*, **25**(1):16–18, 2019.

[GLK22] Deep Ganguli, Liane Lovitt, Jackson Kernion, Amanda Askell, Yuntao Bai, Saurav Kadavath, Ben Mann, Ethan Perez, Nicholas Schiefer, Kamal Ndousse, et al. "Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned." *arXiv preprint arXiv:2209.07858*, 2022.

[GYM21] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. "Knowledge distillation: A survey." *International Journal of Computer Vision*, 2021.

[HB22] Peter Hase and Mohit Bansal. "When Can Models Learn From Explanations? A Formal Framework for Understanding the Roles of Explanation Data." In *ACL Workshop*, 2022.

[HBM19] Braden Hancock, Antoine Bordes, Pierre-Emmanuel Mazare, and Jason Weston. "Learning from Dialogue after Deployment: Feed Yourself, Chatbot!" In *ACL*, 2019.

[HDW22] Shengding Hu, Ning Ding, Huadong Wang, Zhiyuan Liu, Jingang Wang, Juanzi Li, Wei Wu, and Maosong Sun. "Knowledgeable Prompt-tuning: Incorporating Knowledge into Prompt Verbalizer for Text Classification." In *ACL'22*, pp. 2225–2240, 2022.

[HGH22] Jiaxin Huang, Shixiang Shane Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. "Large language models can self-improve." *arXiv preprint arXiv:2210.11610*, 2022.

[HJA20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models." *Advances in Neural Information Processing Systems*, **33**:6840–6851, 2020.

[HKM21] Karen Hambardzumyan, Hrant Khachatrian, and Jonathan May. "WARP: Word-level Adversarial ReProgramming." In *ACL'21*, pp. 4921–4933, 2021.

[HL91] Stephen T Hedetniemi and Renu C Laskar. "Bibliography on domination in graphs and some basic definitions of domination parameters." In *Annals of discrete mathematics*, volume 48, pp. 257–277. Elsevier, 1991.

[HLY23] Cheng-Yu Hsieh, Chun-Liang Li, Chih-kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. "Distilling Step-by-Step! Outperforming Larger Language Models with Less Training Data and Smaller Model Sizes." In *ACL Findings*, 2023.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory." *Neural computation*, **9**(8):1735–1780, 1997.

[HSW22] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. "LoRA: Low-Rank Adaptation of Large Language Models." In *ICLR*, 2022.

[HSY22] Namgyu Ho, Laura Schmid, and Se-Young Yun. "Large Language Models Are Reasoning Teachers." *arXiv preprint arXiv:2212.10071*, 2022.

[HVD15] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. "Distilling the knowledge in a neural network." *arXiv preprint arXiv:1503.02531*, 2015.

[HY20] Ammar Haydari and Yasin Yilmaz. "Deep reinforcement learning for intelligent transportation systems: A survey." *TITS*, 2020.

[ISF08] Sergio J Ibáñez, Jaime Sampaio, Sebastian Feu, Alberto Lorenzo, Miguel A Gómez, and Enrique Ortega. "Basketball game-related statistics that discriminate between teams' season-long success." *EJSS*, **8**(6):369–372, 2008.

[JCL20] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. "Spanbert: Improving pre-training by representing and predicting spans." *TACL*, **8**:64–77, 2020.

[JCW17] Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. "TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension." In *ACL'17*, pp. 1601–1611, 2017.

[JDL19] Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. "PubMedQA: A Dataset for Biomedical Research Question Answering." In *EMNLP*, 2019.

[JDT22] Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. "Planning with diffusion for flexible behavior synthesis." *arXiv preprint arXiv:2205.09991*, 2022.

[JGS19] Natasha Jaques, Asma Ghandeharioun, Judy Hanwen Shen, Craig Ferguson, Agata Lapedriza, Noah Jones, Shixiang Gu, and Rosalind Picard. "Way off-policy batch deep reinforcement learning of implicit human preferences in dialog." *arXiv preprint arXiv:1907.00456*, 2019.

[JHC20] Hangtian Jia, Yujing Hu, Yingfeng Chen, Chunxu Ren, Tangjie Lv, Changjie Fan, and Chongjie Zhang. "Fever Basketball: A Complex, Flexible, and Asynchronized Sports Game Environment for Multi-agent Reinforcement Learning." *arXiv preprint arXiv:2012.03204*, 2020.

[JLF23] Z Ji, N Lee, R Frieske, T Yu, D Su, Y Xu, E Ishii, Y J Bang, A Madotto, and P Fung. "Survey of Hallucination in Natural Language Generation." *ACM Computing Surveys*, 2023.

[JRH20] Hangtian Jia, Chunxu Ren, Yujing Hu, Yingfeng Chen, Tangjie Lv, Changjie Fan, Hongyao Tang, and Jianye Hao. "Mastering basketball with deep reinforcement learning: An integrated curriculum training approach." In *AAMAS*, pp. 1872–1874, 2020.

[JSL18] Junqi Jin, Chengru Song, Han Li, Kun Gai, Jun Wang, and Weinan Zhang. "Real-time bidding with multi-agent reinforcement learning in display advertising." In *CIKM*, pp. 2193–2201, 2018.

[KAL21] Gregory Kahn, Pieter Abbeel, and Sergey Levine. "Badgr: An autonomous self-supervised learning-based navigation system." *IEEE Robotics and Automation Letters*, **6**(2):1312–1319, 2021.

[KB14] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980*, 2014.

[KBP13] Jens Kober, J Andrew Bagnell, and Jan Peters. “Reinforcement learning in robotics: A survey.” *IJRR*, **32**(11):1238–1274, 2013.

[KGR22] Takeshi Kojima, Shixiang (Shane) Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. “Large Language Models are Zero-Shot Reasoners.” In *NeurIPS*, 2022.

[KIP18a] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. “Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation.” *arXiv preprint arXiv:1806.10293*, 2018.

[KIP18b] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. “Scalable deep reinforcement learning for vision-based robotic manipulation.” In *ICRL*, pp. 651–673, 2018.

[KMH20] J Kaplan, S McCandlish, T Henighan, et al. “Scaling laws for neural language models.” *arXiv preprint arXiv:2001.08361*, 2020.

[KNL21] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. “Offline reinforcement learning with implicit q-learning.” *arXiv preprint arXiv:2110.06169*, 2021.

[KPR19] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. “Natural Questions: A Benchmark for Question Answering Research.” *TACL*, **7**:453–466, 2019.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks.” *NeurIPS*, **25**:1097–1105, 2012.

[KSS17] Aniruddha Kembhavi, Minjoon Seo, Dustin Schwenk, Jonghyun Choi, Ali Farhadi, and Hannaneh Hajishirzi. “Are you smarter than a sixth grader? textbook question answering for multimodal machine comprehension.” In *CVPR’17*, pp. 4999–5007, 2017.

[KST19] Nan Rosemary Ke, Amanpreet Singh, Ahmed Touati, Anirudh Goyal, Yoshua Bengio, Devi Parikh, and Dhruv Batra. “Modeling the long term future in model-based reinforcement learning.” In *International Conference on Learning Representations*, 2019.

[KST21] B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A Al Sallab, Senthil Yogamani, and Patrick Pérez. “Deep reinforcement learning for autonomous driving: A survey.” *IEEE Transactions on Intelligent Transportation Systems*, **23**(6):4909–4926, 2021.

[KZT20] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. "Conservative q-learning for offline reinforcement learning." *Advances in Neural Information Processing Systems*, **33**:1179–1191, 2020.

[LAC21a] B Lester, R Al-Rfou, and N Constant. "The Power of Scale for Parameter-Efficient Prompt Tuning." In *EMNLP*, 2021.

[LAC21b] Brian Lester, Rami Al-Rfou, and Noah Constant. "The Power of Scale for Parameter-Efficient Prompt Tuning." In *EMNLP'21*, pp. 3045–3059, 2021.

[LBW22] Robert Logan IV, Ivana Balažević, Eric Wallace, Fabio Petroni, Sameer Singh, and Sebastian Riedel. "Cutting Down on Prompts and Parameters: Simple Few-Shot Learning with Language Models." In *Findings of ACL'22*, pp. 2824–2835, 2022.

[LCG20] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations." In *ICLR'20*, 2020.

[LCY17] Hoang M Le, Peter Carr, Yisong Yue, and Patrick Lucey. "Data-driven ghosting using deep imitation learning." 2017.

[LDR19] Patrick Lewis, Ludovic Denoyer, and Sebastian Riedel. "Unsupervised Question Answering by Cloze Translation." In *ACL'19*, pp. 4896–4910, 2019.

[LDT24] Zheyuan Liu, Guangyao Dou, Zhaoxuan Tan, Yijun Tian, and Meng Jiang. "Towards Safer Large Language Models through Machine Unlearning." *arXiv preprint arXiv:2402.10058*, 2024.

[LDX22] Lihui Liu, Boxin Du, Jiejun Xu, Yinglong Xia, and Hanghang Tong. "Joint Knowledge Graph Completion and Question Answering." In *KDD'22*, pp. 1098–1108, 2022.

[LH18] Libin Liu and Jessica Hodgins. "Learning basketball dribbling skills using trajectory optimization and deep reinforcement learning." *TOG*, **37**(4):1–14, 2018.

[LHE21] Stephanie Lin, Jacob Hilton, and Owain Evans. "Truthfulqa: Measuring how models mimic human falsehoods." *arXiv preprint arXiv:2109.07958*, 2021.

[LHT24] Zheyuan Liu, Xiaoxin He, Yijun Tian, and Nitesh V Chawla. "Can we soft prompt LLMs for graph learning tasks?" In *WWW*, 2024.

[Li23] Liunian Harold Li et al. "Symbolic Chain-of-Thought Distillation: Small Models Can Also "Think" Step-by-Step." *arXiv preprint arXiv:2306.14050*, 2023.

[LJA18] Hoang Le, Nan Jiang, Alekh Agarwal, Miroslav Dudík, Yisong Yue, and Hal Daumé. "Hierarchical imitation and reinforcement learning." In *ICML*, 2018.

[LKT20] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. "Offline reinforcement learning: Tutorial, review, and perspectives on open problems." *arXiv preprint arXiv:2005.01643*, 2020.

[LL21] X L Li and P Liang. "Prefix-Tuning: Optimizing Continuous Prompts for Generation." In *ACL*, 2021.

[LLG20] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension." In *ACL'20*, pp. 7871–7880, 2020.

[LMX22] P Lu, S Mishra, T Xia, L Qiu, K-W Chang, S-C Zhu, O Tafjord, P Clark, and A Kalyan. "Learn to Explain: Multimodal Reasoning via Thought Chains for Science Question Answering." In *NeurIPS*, 2022.

[LOG19] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. "Roberta: A robustly optimized bert pretraining approach." *arXiv preprint arXiv:1907.11692*, 2019.

[LPM23] Harrison Lee, Samrat Phatale, Hassan Mansoor, Kellie Lu, Thomas Mesnard, Colton Bishop, Victor Carbune, and Abhinav Rastogi. "Rlaif: Scaling reinforcement learning from human feedback with ai feedback." *arXiv preprint arXiv:2309.00267*, 2023.

[LSP21] Yudong Luo, Oliver Schulte, and Pascal Poupart. "Inverse reinforcement learning for team sports: valuing actions and players." In *IJCAI*, pp. 3356–3363, 2021.

[LWY21] Bill Yuchen Lin, Ziyi Wu, Yichi Yang, Dong-Ho Lee, and Xiang Ren. "RiddleSense: Reasoning about Riddle Questions Featuring Linguistic Creativity and Commonsense Knowledge." In *ACL Findings*, 2021.

[LXR22] Xiao-Yang Liu, Ziyi Xia, Jingyang Rui, Jiechao Gao, Hongyang Yang, Ming Zhu, Christina Wang, Zhaoran Wang, and Jian Guo. "FinRL-Meta: Market environments and benchmarks for data-driven financial reinforcement learning." *Advances in Neural Information Processing Systems*, **35**:1835–1849, 2022.

[LYC17] Hoang M Le, Yisong Yue, Peter Carr, and Patrick Lucey. "Coordinated multi-agent imitation learning." In *ICML*, pp. 1995–2003, 2017.

[LYT23] Yang Liu, Yuanshun Yao, Jean-Francois Ton, Xiaoying Zhang, Ruocheng Guo Hao Cheng, Yegor Klochkov, Muhammad Faaiz Taufiq, and Hang Li. "Trustworthy LLMs: a Survey and Guideline for Evaluating Large Language Models' Alignment." *arXiv preprint arXiv:2308.05374*, 2023.

[LYZ23] Xian Li, Ping Yu, Chunting Zhou, Timo Schick, Luke Zettlemoyer, Omer Levy, Jason Weston, and Mike Lewis. "Self-alignment with instruction backtranslation." *arXiv preprint arXiv:2308.06259*, 2023.

[LZD21] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. "GPT understands, too." *arXiv preprint arXiv:2103.10385*, 2021.

[MBG16] Avery McIntyre, Joel Brooks, John Guttag, and Jenna Wiens. "Recognizing and analyzing ball screen defense in the nba." In *Proc. of the MIT Sloan Sports Analytics Conference*, pp. 11–12, 2016.

[MBM21] Charbel Merhej, Ryan J Beal, Tim Matthews, and Sarvapali Ramchurn. "What Happened Next? Using Deep Learning to Value Defensive Actions in Football Event-Data." In *KDD*, pp. 3394–3403, 2021.

[MCK18] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. "Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering." In *EMNLP*, 2018.

[MK19] Terry Lingze Meng and Matloob Khushi. "Reinforcement learning in financial markets." *Data*, **4**(3):110, 2019.

[MLH22] Sewon Min, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. "Noisy Channel Language Model Prompting for Few-Shot Text Classification." In *ACL'22*, pp. 5316–5330, 2022.

[MMA22] Lucie Charlotte Magister, Jonathan Mallinson, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. "Teaching Small Language Models to Reason." *arXiv preprint arXiv:2212.08410*, 2022.

[MSY22] Alireza Mohammadshahi, Thomas Scialom, Majid Yazdani, Pouya Yanki, Angela Fan, James Henderson, and Marzieh Saeidi. "RQUGE: Reference-Free Metric for Evaluating Question Generation by Answering the Question." *arXiv preprint arXiv:2211.01482*, 2022.

[MTG23] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. "Self-refine: Iterative refinement with self-feedback." *arXiv preprint arXiv:2303.17651*, 2023.

[MZZ20] Xiyao Ma, Qile Zhu, Yanlin Zhou, and Xiaolin Li. "Improving question generation with sentence-level semantic matching and answer position inferring." In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 8464–8471, 2020.

[NKF18] Anusha Nagabandi, Gregory Kahn, Ronald S Fearing, and Sergey Levine. "Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning." In *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 7559–7566. IEEE, 2018.

[NL11] Tal Neiman and Yonatan Loewenstein. "Reinforcement learning in professional basketball players." *Nature Comm*, **2**(1):1–8, 2011.

[NRL20] Sharan Narang, Colin Raffel, Katherine Lee, Adam Roberts, Noah Fiedel, and Karishma Malkan. "WT5?! Training Text-to-Text Models to Explain Their Predictions." *arXiv preprint arXiv:2004.14546*, 2020.

[NW72] John Ashworth Nelder and Robert WM Wedderburn. "Generalized linear models." *J. R. Stat. Soc.*, **135**(3):370–384, 1972.

[Ope23a] OpenAI. "GPT-4: Improving Language Model Performance through Scale and Data." *arXiv*, 2023.

[Ope23b] R OpenAI. "GPT-4 technical report." *arXiv*, pp. 2303–08774, 2023.

[PBD22] Danish Pruthi, Rachit Bansal, Bhuwan Dhingra, Livio Baldini Soares, Michael Collins, Zachary C Lipton, Graham Neubig, and William W Cohen. "Evaluating Explanations: How Much Do Explanations from the Teacher Aid Students?" In *ACL*, 2022.

[pbp16] "Play-by-Play Data." Available at `https://www.bigdataball.com/datasets/nba/`, 2016.

[PBT19] Caleb Pagé, Pierre-Michel Bernier, and Maxime Trempe. "Using video simulations and virtual reality to improve decision-making skills in basketball." *Journal of sports sciences*, **37**(21):2403–2410, 2019.

[PCX21] Liangming Pan, Wenhu Chen, Wenhan Xiong, Min-Yen Kan, and William Yang Wang. "Unsupervised Multi-hop Question Answering by Question Generation." In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 5866–5880, 2021.

[PN17] Athanasios S Polydoros and Lazaros Nalpantidis. "Survey of model-based reinforcement learning: Applications on robotics." *JINT*, **86**(2):153–173, 2017.

[PSS20] Raul Puri, Ryan Spring, Mohammad Shoeybi, Mostofa Patwary, and Bryan Catanzaro. "Training Question Answering Models From Synthetic Data." In *EMNLP'20*, pp. 5811–5826, 2020.

[RFB15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pp. 234–241. Springer, 2015.

[RHD17] Andrew Slavin Ross, Michael C Hughes, and Finale Doshi-Velez. "Right for the Right Reasons: Training Differentiable Models by Constraining Their Explanations." *arXiv preprint arXiv:1703.03717*, 2017.

[RJP23] Kavel Rao, Liwei Jiang, Valentina Pyatkin, Yuling Gu, Niket Tandon, Nouha Dziri, Faeze Brahman, and Yejin Choi. "What Makes it Ok to Set a Fire? Iterative Self-distillation of Contexts and Rationales for Disambiguating Defeasible Social and Moral Situations." In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 12140–12159, 2023.

[RKB21] Ori Ram, Yuval Kirstain, Jonathan Berant, Amir Globerson, and Omer Levy. "Few-Shot Question Answering by Pretraining Span Selection." In *ACL'21*, pp. 3066–3079, 2021.

[RMX19] Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. "Explain Yourself! Leveraging Language Models for Commonsense Reasoning." In *ACL*, 2019.

[RPW17] Hector Ruiz, Paul Power, Xinyu Wei, and Patrick Lucey. "" The Leicester City Fairytale?" Utilizing New Soccer Analytics Tools to Compare Performance in the 15/16 & 16/17 EPL Seasons." In *KDD*, pp. 1991–2000, 2017.

[RSR20] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. "Exploring the limits of transfer learning with a unified text-to-text transformer." *JMLR*, **21**(140):1–67, 2020.

[RVD21] Pieter Robberechts, Jan Van Haaren, and Jesse Davis. "A Bayesian Approach to In-Game Win Probability in Soccer." In *KDD*, pp. 3512–3521, 2021.

[RZL16] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. "SQuAD: 100,000+ Questions for Machine Comprehension of Text." In *EMNLP'16*, pp. 2383–2392, 2016.

[SDC19] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter." *arXiv preprint arXiv:1910.01108*, 2019.

[SDS20] Xiangyu Sun, Jack Davis, Oliver Schulte, and Guiliang Liu. "Cracking the black box: Distilling deep sports analytics." In *KDD*, pp. 3154–3162, 2020.

[SJH23] Tianhao Shen, Renren Jin, Yufei Huang, Chuang Liu, Weilong Dong, Zishan Guo, Xinwei Wu, Yan Liu, and Deyi Xiong. "Large language model alignment: A survey." *arXiv preprint arXiv:2309.15025*, 2023.

[SPN22] Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, Elton Zhang, Rewon Child, Reza Yazdani Aminabadi, Julie Bernauer, Xia Song, Mohammad Shoeybi, Yuxiong He, Michael Houston, Saurabh Tiwary, and Bryan Catanzaro. "Using DeepSpeed and Megatron to Train Megatron-Turing NLG 530B, a Large-Scale Generative Language Model." *arXiv preprint arXiv:2201.11990*, 2022.

[spo16] "SportVU Data." Available at `https://github.com/rajshah4/BasketballData/tree/master/2016.NBA.Raw.SportVU.Game.Logs`, 2016.

[SRR22] Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. "Beyond the imitation game: Quantifying and extrapolating the capabilities of language models." *arXiv preprint arXiv:2206.04615*, 2022.

[SS21a] Timo Schick and Hinrich Schütze. "Exploiting Cloze-Questions for Few-Shot Text Classification and Natural Language Inference." In *EACL'21*, pp. 255–269, 2021.

[SS21b] Timo Schick and Hinrich Schütze. "Generating Datasets with Pretrained Language Models." In *EMNLP'21*, pp. 6943–6951, 2021.

[SS21c] Timo Schick and Hinrich Schütze. "It's Not Just Size That Matters: Small Language Models Are Also Few-Shot Learners." In *NAACL'21*, pp. 2339–2352, 2021.

[SSS17] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. "Mastering the game of go without human knowledge." *Nature*, **550**(7676):354–359, 2017.

[SSZ23] Zhiqing Sun, Yikang Shen, Qinhong Zhou, Hongxin Zhang, Zhenfang Chen, David Cox, Yiming Yang, and Chuang Gan. "Principle-driven self-alignment of language models from scratch with minimal human supervision." *arXiv preprint arXiv:2305.03047*, 2023.

[SWM15] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. "Deep unsupervised learning using nonequilibrium thermodynamics." In *International Conference on Machine Learning*, pp. 2256–2265. PMLR, 2015.

[SX23] Yucheng Shi, Shaochen Xu, et al. "Mededit: Model Editing for Medical Question Answering with External Knowledge Bases." *arXiv preprint arXiv:2309.16035*, 2023.

[SYW22] William Saunders, Catherine Yeh, Jeff Wu, Steven Bills, Long Ouyang, Jonathan Ward, and Jan Leike. "Self-critiquing models for assisting human evaluators." *arXiv preprint arXiv:2206.05802*, 2022.

[SZH22] Omar Shaikh, Hongxin Zhang, William Held, Michael Bernstein, and Diyi Yang. "On Second Thought, Let's Not Think Step by Step! Bias and Toxicity in Zero-Shot Reasoning." *arXiv preprint arXiv:2212.08061*, 2022.

[Tal14] Erik Talvitie. "Model Regularization for Stable Sample Rollouts." In *UAI*, pp. 780–789, 2014.

[TBM15a] George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, et al. "An overview of the BIOASQ large-scale biomedical semantic indexing and question answering competition." *BMC Bioinformatics*, 2015.

[TBM15b] George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, et al. "An overview of the BIOASQ large-scale biomedical semantic indexing and question answering competition." *BMC bioinformatics*, **16**(1):1–28, 2015.

[TDC20] Changjia Tian, Varuna De Silva, Michael Caine, and Steve Swanson. "Use of machine learning to automate the identification of basketball strategies using whole team player tracking data." *Applied Sciences*, **10**(1):24, 2020.

[TF20] Zachary Terner and Alexander Franks. "Modeling player and team performance in basketball." *Annual Review of Statistics and Its Application*, **8**, 2020.

[TGZ23] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. "Stanford Alpaca: An Instruction-following LLaMA model." *GitHub repository*, 2023.

[THC24] Yijun Tian, Yikun Han, Xiusi Chen, Wei Wang, and Nitesh V Chawla. "TinyLLM: Learning a Small Student from Multiple Large Language Models." *arXiv preprint arXiv:2402.04616*, 2024.

[TM23] Hugo Touvron, Louis Martin, et al. "Llama 2: Open Foundation and Fine-Tuned Chat Models." *arXiv preprint arXiv:2307.09288*, 2023.

[TMB21] Derek Tam, Rakesh R Menon, Mohit Bansal, Shashank Srivastava, and Colin Raffel. "Improving and Simplifying Pattern Exploiting Training." In *EMNLP'21*, pp. 4980–4991, 2021.

[TMS23] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. "Llama 2: Open foundation and fine-tuned chat models." *arXiv preprint arXiv:2307.09288*, 2023.

[TOM21] Karl Tuyls, Shayegan Omidshafiei, Paul Muller, Zhe Wang, Jerome Connor, Daniel Hennes, Ian Graham, William Spearman, Tim Waskett, Dafydd Steel, et al. "Game Plan: What AI can do for Football, and What Football can do for AI." *JAIR*, **71**:41–88, 2021.

[TPZ23] Yijun Tian, Shichao Pei, Xiangliang Zhang, Chuxu Zhang, and Nitesh V Chawla. "Knowledge Distillation on Graphs: A Survey." *arXiv preprint arXiv:2302.00219*, 2023.

[TSW24a] Yijun Tian, Huan Song, Zichen Wang, Haozhu Wang, Ziqing Hu, Fang Wang, Nitesh V Chawla, and Panpan Xu. "Graph neural prompting with large language models." In *AAAI*, 2024.

[TSW24b] Yijun Tian, Huan Song, Zichen Wang, Haozhu Wang, Ziqing Hu, Fang Wang, Nitesh V Chawla, and Panpan Xu. "Graph neural prompting with large language models." In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19080–19088, 2024.

[TWY17] Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. "NewsQA: A Machine Comprehension Dataset." In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pp. 191–200, 2017.

[TZG23] Yijun Tian, Chuxu Zhang, Zhichun Guo, Xiangliang Zhang, and Nitesh V Chawla. "Learning MLPs on Graphs: A Unified View of Effectiveness, Robustness, and Efficiency." In *ICLR*, 2023.

[TZT24] Zhaoxuan Tan, Qingkai Zeng, Yijun Tian, Zheyuan Liu, Bing Yin, and Meng Jiang. "Democratizing Large Language Models via Personalized Parameter-Efficient Fine-tuning." *arXiv preprint arXiv:2402.04401*, 2024.

[VGS16] Hado Van Hasselt, Arthur Guez, and David Silver. "Deep reinforcement learning with double q-learning." In *AAAI*, volume 30, 2016.

[VK14] Denny Vrandečić and Markus Krötzsch. "Wikidata: a free collaborative knowledgebase." *Communications of the ACM*, **57**(10):78–85, 2014.

[VSP17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. "Attention is all you need." *arXiv preprint arXiv:1706.03762*, 2017.

[WBC19] Tingwu Wang, Xuchan Bao, Ignasi Clavera, Jerrick Hoang, Yeming Wen, Eric Langlois, Shunshi Zhang, Guodong Zhang, Pieter Abbeel, and Jimmy Ba. "Benchmarking model-based reinforcement learning." *arXiv preprint arXiv:1907.02057*, 2019.

[WBZ] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. "AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation." In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*.

[WCI22] Peifeng Wang, Aaron Chan, Filip Ilievski, Muhao Chen, and Xiang Ren. "PINTO: Faithful Language Reasoning Using Prompt-Generated Rationales." In *ICLR*, 2022.

[WCP22] Wei-Yao Wang, Teng-Fong Chan, Wen-Chih Peng, Hui-Kuo Yang, Chih-Chuan Wang, and Yao-Chung Fan. "How Is the Stroke? Inferring Shot Influence in Badminton Matches via Long Short-Term Dependencies." *ACM Transactions on Intelligent Systems and Technology*, **14**(1):1–22, 2022.

[WFS18] Jiaxuan Wang, Ian Fox, Jonathan Skaza, Nick Linck, Satinder Singh, and Jenna Wiens. "The advantage of doubling: A deep reinforcement learning approach to studying the double team in the NBA." *arXiv preprint arXiv:1803.02940*, 2018.

[WLX21] Shuohang Wang, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. "Want To Reduce Labeling Cost? GPT-3 Can Help." In *EMNLP Findings*, 2021.

[WMS21] Sarah Wiegreffe, Ana Marasovic, and Noah A Smith. "Measuring Association Between Labels and Free-Text Rationales." In *EMNLP*, 2021.

[WRT24] W Wei, X Ren, J Tang, Q Wang, L Su, S Cheng, J Wang, D Yin, and C Huang. "LLMRec: Large Language Models with Graph Augmentation for Recommendation." In *WSDM*, 2024.

[WSC22] Wei-Yao Wang, Hong-Han Shuai, Kai-Shiang Chang, and Wen-Chih Peng. "Shuttlenet: Position-aware fusion of rally progress and player styles for stroke forecasting in badminton." In *AAAI*, volume 36, pp. 4219–4227, 2022.

[WSF22] BigScience Workshop, Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, et al. "Bloom: A 176b-parameter open-access multilingual language model." *arXiv preprint arXiv:2211.05100*, 2022.

[WSM19] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. "Glue: A multi-task benchmark and analysis platform for natural language understanding." In *ICLR'19*, 2019.

[WTB22] J Wei, Y Tay, R Bommasani, et al. "Emergent Abilities of Large Language Models." *arXiv preprint arXiv:2206.07682*, 2022.

[WW22] Jason Wei, Xuezhi Wang, et al. "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models." In *NeurIPS*, 2022.

[WWL23] Zhongwei Wan, Xin Wang, Che Liu, Samiul Alam, Yu Zheng, Zhongnan Qu, Shen Yan, Yi Zhu, Quanlu Zhang, Mosharaf Chowdhury, et al. "Efficient large language models: A survey." *arXiv preprint arXiv:2312.03863*, **1**, 2023.

[WWQ22] Jianing Wang, Chengyu Wang, Minghui Qiu, Qiuhui Shi, Hongbin Wang, Jun Huang, and Ming Gao. "KECP: Knowledge Enhanced Contrastive Prompting for Few-shot Extractive Question Answering." *arXiv preprint arXiv:2205.03071*, 2022.

[WYK20] Qifan Wang, Li Yang, Bhargav Kanagal, Sumit Sanghai, D Sivakumar, Bin Shu, Zac Yu, and Jon Elsas. "Learning to extract attribute value from product via question answering: A multi-task approach." In *KDD'20*, pp. 47–55, 2020.

[WZ16] Kuan-Chieh Wang and Richard Zemel. "Classifying NBA offensive plays using neural networks." In *Proc. of MIT Sloan Sports Analytics Conference*, volume 4, 2016.

[WZ19] Jason Wei and Kai Zou. "EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks." In *EMNLP'19*, pp. 6382–6388, 2019.

[WZQ23] Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, et al. "A Survey on Large Language Models for Recommendation." *arXiv preprint arXiv:2305.19860*, 2023.

[XDH20] Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. "Unsupervised data augmentation for consistency training." In *NeurIPS'20*, pp. 6256–6268, 2020.

[XLZ23] Weiwen Xu, Xin Li, Wenxuan Zhang, Meng Zhou, Wai Lam, Luo Si, and Lidong Bing. "From Cloze to Comprehension: Retrofitting Pre-trained Masked Language Models to Pre-trained Machine Reader." In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[YLN19] Chao Yu, Jiming Liu, and Shamim Nemati. "Reinforcement learning in healthcare: A survey." *arXiv preprint arXiv:1908.08796*, 2019.

[YMF20] Yiben Yang, Chaitanya Malaviya, Jared Fernandez, Swabha Swayamdipta, Ronan Le Bras, Ji-Ping Wang, Chandra Bhagavatula, Yejin Choi, and Doug Downey. "Generative Data Augmentation for Commonsense Reasoning." In *Findings of EMNLP'20*, pp. 1008–1025, 2020.

[YQZ18] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. "HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering." In *EMNLP'18*, pp. 2369–2380, 2018.

[YSK22] Chen Yanai, Adir Solomon, Gilad Katz, Bracha Shapira, and Lior Rokach. "Q-Ball: Modeling Basketball Games Using Deep Reinforcement Learning." In *AAAI*, pp. 8806–8813. AAAI Press, 2022.

[YXL19] Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. "End-to-End Open-Domain Question Answering with BERTserini." In *ACL'19, System Demonstrations*, pp. 72–77, 2019.

[ZBL13] Tong Zhao, Naiwen Bian, Chunping Li, and Mengya Li. "Topic-level expert modeling in community question answering." In *SDM'13*, pp. 776–784, 2013.

[ZC23] Lianmin Zheng, Wei-Lin Chiang, et al. "Judging LLM-as-a-judge with MT-Bench and Chatbot Arena." *arXiv preprint arXiv:2306.05685*, 2023.

[ZCJ24] Yu Zhang, Xiusi Chen, Bowen Jin, Sheng Wang, Shuiwang Ji, Wei Wang, and Jiawei Han. "A Comprehensive Survey of Scientific Large Language Models and Their Applications in Scientific Discovery." *arXiv preprint arXiv:2406.10833*, 2024.

[ZEP07] Omar Zaidan, Jason Eisner, and Christine Piatko. "Using "Annotator Rationales" to Improve Machine Learning for Text Categorization." In *NAACL*, 2007.

[ZFC21] Zexuan Zhong, Dan Friedman, and Danqi Chen. "Factual Probing Is [MASK]: Learning vs. Learning to Recall." In *NAACL'21*, pp. 5017–5033, 2021.

[ZGS19] Jie Zhao, Ziyu Guan, and Huan Sun. "Riker: Mining rich keyword representations for interpretable product question answering." In *KDD'19*, pp. 1389–1398, 2019.

[ZLC21] Ningyu Zhang, Luoqiu Li, Xiang Chen, Shumin Deng, Zhen Bi, Chuanqi Tan, Fei Huang, and Huajun Chen. "Differentiable Prompt Makes Pre-trained Language Models Better Few-shot Learners." In *ICLR'21*, 2021.

[ZLW21] Fengbin Zhu, Wenqiang Lei, Chao Wang, Jianming Zheng, Soujanya Poria, and Tat-S Chua. "Retrieving and Reading: A Comprehensive Survey on Open-Domain Question Answering." *arXiv preprint arXiv:2101.00774*, 2021.

[ZMT23] Eric Zelikman, Wanjing Ma, Jasmine Tran, Diyi Yang, Jason Yeatman, and Nick Haber. "Generating and Evaluating Tests for K-12 Students with Language Model Simulations: A Case Study on Sentence Reading Efficiency." In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 2190–2205, 2023.

[ZMW16] Ye Zhang, Iain Marshall, and Byron C Wallace. "Rationale-Augmented Convolutional Neural Networks for Text Classification." In *EMNLP*, 2016.

[ZWM22] Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. "Star: Bootstrapping Reasoning with Reasoning." In *NeurIPS*, 2022.

[ZY23] Xuanyu Zhang and Qing Yang. "Self-QA: Unsupervised Knowledge Guided Language Model Alignment." *arXiv preprint arXiv:2305.11952*, 2023.

[ZYZ20] Xinshi Zang, Huaxiu Yao, Guanjie Zheng, Nan Xu, Kai Xu, and Zhenhui Li. "Metalight: Value-based meta-reinforcement learning for traffic signal control." In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 1153–1160, 2020.

[ZZL23] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. "A Survey of Large Language Models." *arXiv preprint arXiv:2303.18223*, 2023.

[ZZZ18] Guanjie Zheng, Fuzheng Zhang, Zihan Zheng, Yang Xiang, Nicholas Jing Yuan, Xing Xie, and Zhenhui Li. "DRN: A deep reinforcement learning framework for news recommendation." In *WWW*, pp. 167–176, 2018.