Automatic Question Answering and Knowledge Discovery from Electronic Health Records

Ping Wang

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Computer Science and Applications

Chandan K. Reddy, Chair
Naren Ramakrishnan
Chang-Tien Lu
Jiepu Jiang
Sutanay Choudhury

July 13, 2021
Blacksburg, Virginia

Keywords: Electronic Health Records, Question Answering, Knowledge Discovery, Knowledge Graph, Survival Analysis.

Automatic Question Answering and Knowledge Discovery from Electronic Health Records

Ping Wang

(ABSTRACT)

Electronic Health Records (EHR) data contain comprehensive longitudinal patient information, which is usually stored in databases in the form of either multi-relational structured tables or unstructured texts, e.g., clinical notes. EHR provide a useful resource to assist doctors' decision making, however, they also present many unique challenges that limit the efficient use of the valuable information, such as large data volume, heterogeneous and dynamic information, medical term abbreviations, and noisy nature caused by misspelled words.

This dissertation focuses on the development and evaluation of advanced machine learning algorithms to solve the following research questions: (1) How to seek answers from EHR for clinical activity related questions posed in human language without the assistance of database and natural language processing (NLP) domain experts, (2) How to discover underlying relationships of different events and entities in structured tabular EHRs, and (3) How to predict when a medical event will occur and estimate its probability based on previous medical information of patients.

First, to automatically retrieve answers for natural language questions from the structured tables in EHR, we study the question-to-SQL generation task by generating the corresponding SQL query of the input question. We propose a translation-edit model driven by a language generation module and an editing module for the SQL query generation task. This model helps automatically translate clinical activity related questions to SQL queries, so that the doctors only need to provide their questions in natural language to get the answers they need. We also create a large-scale dataset for question answering on tabular EHR to simulate a more realistic setting. Our performance evaluation shows that the proposed model is effective in handling the unique challenges about clinical terminologies, such as abbreviations and misspelled words.

Second, to automatically identify answers for natural language questions from unstructured clinical notes in EHR, we propose to achieve this goal by querying a knowledge base constructed based on fine-grained document-level expert annotations of clinical records for various NLP tasks. We first create a dataset for clinical knowledge base question answering with two sets: clinical knowledge base and question-answer pairs. An attention-based aspect-level reasoning model is developed and evaluated on the new dataset. Our experimental analysis shows that it is effective in identifying answers and also allows us to analyze the impact of different answer aspects in predicting correct answers.

Third, we focus on discovering underlying relationships of different entities (e.g., patient, disease, medication, and treatment) in tabular EHR, which can be formulated as a link prediction problem in graph domain. We develop a self-supervised learning framework

for better representation learning of entities across a large corpus and also consider local contextual information for the down-stream link prediction task. We demonstrate the effectiveness, interpretability, and scalability of the proposed model on the healthcare network built from tabular EHR. It is also successfully applied to solve link prediction problems in a variety of domains, such as e-commerce, social networks, and academic networks.

Finally, to dynamically predict the occurrence of multiple correlated medical events, we formulate the problem as a temporal (multiple time-points) and multi-task learning problem using tensor representation. We propose an algorithm to jointly and dynamically predict several survival problems at each time point and optimize it with the Alternating Direction Methods of Multipliers (ADMM) algorithm. The model allows us to consider both the dependencies between different tasks and the correlations of each task at different time points. We evaluate the proposed model on two real-world applications and demonstrate its effectiveness and interpretability.

Automatic Question Answering and Knowledge Discovery from Electronic Health Records

Ping Wang

(GENERAL AUDIENCE ABSTRACT)

Healthcare is an important part of our lives. Due to the recent advances of data collection and storing techniques, a large amount of medical information is generated and stored in Electronic Health Records (EHR). By comprehensively documenting the longitudinal medical history information about a large patient cohort, this EHR data forms a fundamental resource in assisting doctors' decision making including optimization of treatments for patients and selection of patients for clinical trials. However, EHR data also presents a number of unique challenges, such as (i) large-scale and dynamic data, (ii) heterogeneity of medical information, and (iii) medical term abbreviation. It is difficult for doctors to effectively utilize such complex data collected in a typical clinical practice. Therefore, it is imperative to develop advanced methods that are helpful for efficient use of EHR and further benefit doctors in their clinical decision making.

This dissertation focuses on automatically retrieving useful medical information, analyzing complex relationships of medical entities, and detecting future medical outcomes from EHR data. In order to retrieve information from EHR efficiently, we develop deep learning based algorithms that can automatically answer various clinical questions on structured and unstructured EHR data. These algorithms can help us understand more about the challenges in retrieving information from different data types in EHR. We also build a clinical knowledge graph based on EHR and link the distributed medical information and further perform the link prediction task, which allows us to analyze the complex underlying relationships of various medical entities. In addition, we propose a temporal multi-task survival analysis method to dynamically predict multiple medical events at the same time and identify the most important factors leading to the future medical events. By handling these unique challenges in EHR and developing suitable approaches, we hope to improve the efficiency of information retrieval and predictive modeling in healthcare.

Dedications

To my beloved family.

Acknowledgements

It has been an unforgettable and rewarding journey for me to pursue my PhD. I would like to express my sincere gratitude to all the help I received over the past five years.

First of all, my deepest thanks go to my PhD advisor, Dr. Chandan K. Reddy, for his encouragement, support and guidance in my PhD study and research. He is a very kind and supportive advisor. He has made a lot of effort in guiding me towards exploring new research directions, analyzing research challenges, and improving my skills in doing independent research. It is his professional guidance and continuous encouragement that helped me build confidence in the process of graduate study and completion of this dissertation. Also, his wisdom, insights, and passion in doing research and being an advisor benefited me a lot and was influential in my career choice. I can clearly feel the importance and influence of his guidance, advice and help on my career.

I would also like to thank all my committee members: Dr. Naren Ramakrishnan, Dr. Jiepu Jiang, Dr. Chang-Tien Lu and Dr. Sutanay Choudhury, for their important guidance and assistance in the completion of this dissertation. Thanks to Dr. Naren Ramakrishnan and Dr. Jiepu Jiang for their insightful and valuable suggestions from the beginning of the research proposal to the completion of this dissertation. Their deep knowledge and expertise in the field helped me broaden the research directions. I am also very thankful to Dr. Chang-Tien Lu and Dr. Sutanay Choudhury for their guidance in my research. They are also very supportive with many valuable suggestions and advice, and strong recommendation for my future career.

Thanks to all the collaborators at Pacific Northwest National Laboratory: Dr. Sutanay Choudhury, Khushbu Agarwal, and Colby Ham. I still remember the time that we explored new techniques and were investigating new research projects. Special thanks to Dr. Sutanay Choudhury for offering a great opportunity to collaborate on the knowledge graph related projects. He spent a lot of time with me in formulating research problems, discussing technical issues and improving the writing and presentation. I have benefited a lot from his mentoring and support over the past two years.

I want to thank all others who collaborated and helped me in the past few years, including Dr. Tian Shi, Dr. Yan Li, Dr. Liuqing Li, Dr. Xuchao Zhang, Dr. Bhanukiran Vinzamuri, Dr. Mahtab J. Fard, and Karthik Padthe.

In addition, I am grateful to the lab members and others at the Sanghani Center for Artificial Intelligence and Data Analytics, including Khoa Doan, Tian Shi, Aman Ahuja, Ming Zhu, Sindhu Tipirneni, Nurendra Choudhary, Akshita Jha, Dr. Yue Ning, Dr. Rongrong Tao, Lijing Wang, Fanglan Chen, Dr. Tianyi Li, Dr. Xiangyu Zhang, Dr. Mengmeng Cai, to name a few. We had valuable discussions about research problems as colleagues and happy memories as friends. Special thanks to Dr. Yue Ning for her valuable suggestions on my presentation and for hosting my job interview at Stevens.

Moreover, I would like to thank Juanita Victoria, Wanawsha Hawrami, Joyce Newberry, Barbara L. Micale, Roxanne Paul, Sharon Kinder-Potter, Jessica Mullins, Corinne Julien and Torri K. Brown for their administrative support over the years.

My gratitude to my parents is beyond words. It is their endless support, understanding, and encouragement that makes me what I am today. I am also thankful to my sister and brother for taking care of everything at home and allowing me to focus on my research. In addition, I also appreciate the great support, help and encouragement from my parents-in-law.

Finally, I want to thank my husband and my daughter for their continuous support and love. It has been more than ten years since my husband and I met on the first day of college. We experienced ups and downs together. He is not only my partner, but also my good friend and collaborator. It is his passion and wisdom on exploring new things, his hard work and his encouragement over the years that motivated me to pursue my dreams. A huge Thanks to him for always being there. I also thank my little one for making me a mom and for her unconditional trust and love.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Healthcare systems are rapidly evolving in the era of big data. Advances of artificial intelligence in healthcare make it possible for healthcare providers to sift through tremendous amounts of information efficiently, which eventually help them take care of their patients better. There are various types of health information ranging from medical literature to pathology reports. The goal of this dissertation is to develop machine learning methods that can efficiently utilize Electronic Health Records (EHR) to help facilitate physicians' decision making in their clinical practice. EHR data contain comprehensive longitudinal patient information, which is usually stored in databases in the form of either multi-relational structured tables or unstructured texts, e.g., clinical notes. In practice, efficient use of EHR data can significantly benefit doctors/physicians in many different aspects, such as identifying the most suitable medications or procedures for patients with a rare disease, predicting a disease before it strikes, identifying patient cohort for clinical trials, and helping bio-medical research and innovations. This dissertation focuses on two important tasks, including question answering and knowledge discovery, to support better use of EHR data.

The task of question answering aims to automatically and efficiently retrieve clinical information from different types of data in EHR. There are mainly two types of challenges. On the one hand, a large portion of the EHR data in a hospital is typically stored in a relational database with multiple tables. Traditionally, doctors use rule-based systems to interact with EHR data. These systems first turn any predefined-rule, i.e., question, to a SQL query, and then retrieve an answer by executing it on the database. This type of systems are complicated and require lots of training to use them. Yet, they prevent doctors from asking any impromptu questions since they need to seek professional assistant to retrieve answers from databases. On the other hand, besides the structured data, the unstructured clinical note is another important type in EHR, e.g., discharge summaries, and it contains a wealth of information. The plain-text feature makes it extremely difficult for doctors to survey clinical notes for solutions of their problems, even if reviewing a single patient's complete record takes a lot of time and efforts.

To tackle these challenges, recently, question answering (QA) technique has attracted at lot of attention in the healthcare domain due to its ability to automatically answer a natural language question on structured tables [151, 30], plain-text documents [84, 26] and knowledge graphs [150]. More specifically, in the healthcare domain, question answering methods take natural language questions about EHR data as input and return the corresponding information included in EHR as answers. For example, the question "*How many female patients underwent the procedure of abdomen artery incision?*" can be asked against both structured tables and unstructured clinical notes, and a count number will be returned by the question answering methods as the answer to it. However, the methods to answer this question on structured tables and unstructured clinical notes tend to be different. Due to the popularity of query languages on database, the question answering on structured tables is commonly performed by predicting the SQL query corresponding to the given question. While for the unstructured clinical notes, there is no standard language to query the random textual notes. In addition, different from most of the existing question answering works on textual documents, the answer to this question is not explicitly included in the clinical notesand it requires certain reasoning to obtain its answer. All these limitations demand a new design of next-generation systems, which allow doctors to automatically retrieve information from EHR data via actively asking different questions.

Another important task with EHR data is to understand how to accurately discover new knowledge and make predictions based on the comprehensive medical history of patients. This task presents two types of challenges. First, the medical information of different patients is usually organized independently in EHR without much overlap. It is difficult for doctors to discover underlying relationships between patients with similar diseases or received similar treatment and identify the comorbidity relations between different diseases or the co-occurrence of procedures or medications. However, in fact, there exist some complex relations between patients with the same diagnosis, underwent the same procedures, or taking the same medications. To provide personalized precision medical treatment, it is important to also involve such complex underlying relationships for prediction. Second, accurately predicting when a medical event will occur and estimate its probability based on previous medical information of patients requires a large set of patients with the specific medical event. However, it is difficult to collect the training data with sufficient event occurrences in longitudinal studies since the occurrence of the event may not always be observed for all patients during the observation due to non-occurrence of the event by the end of the observation or losing follow-up during the observation. In addition, dynamically following up and predicting the occurrence of multiple medical events simultaneously is also important in the healthcare domain.

To build relationships between different patients, and also the relations between diseases, procedures, and medications, knowledge graph has attracted considerable attentions recently in healthcare domain due to its ability to not only incorporate data information from different resources, but also capture their underlying higher order relationships. In this case, the link prediction task can help us to predict the future diagnosis or recommend suitable treatment

for patients. Many of the existing methods focus on learning a static vector representation for each entity in the knowledge graph and generally apply it to different downstream tasks, including link prediction and question answering. However, in practice, the accurate prediction of the medical events on a patient usually depends on the specific contextual information, such as the current or previous medical information, medical information from patients with similar diseases or treatment. Therefore, this motivates us to contextualize a node's representation based on the specific prediction tasks. In addition, survival analysis methods are commonly used to model data with both the time and the probability of an event of interest occurs as the outcome. They are typically designed to handle the survival problems with a single specific event of interest at a given time point. However, most of them are not specially developed to handle the insufficiency of event occurrences. To support for the dynamic tracking the occurrence of multiple events of interest simultaneously, a simple method is to apply the standard survival analysis method independently to each event at each specific time point. However, it often leads to a sub-optimal solution since the underlying dependencies between these events and the correlations of each single event over time are ignored. All these limitations motivate us to develop a method that can incorporate both types of correlations to jointly analyze multiple events by leveraging the limited number of data information available for each single event.

Therefore, the goal of this dissertation is to tackle the challenges discussed above about question answering and knowledge discovery in healthcare domain using the public available datasets, including Medical Information Mart for Intensive Care III (MIMIC III) dataset and National NLP Clinical Challenges (n2c2) dataset. The main research issues focused in this work are summarized in Section 1.1.

1.1 Research Challenges

In this work, four research challenges related to EHR data are studied. The first two are about question-answering tasks on multi-relational structured tables and unstructured clinical notes, respectively. The other two research issues aim at modeling the EHR data to efficiently discover new knowledge. We summarized these research issues as follows.

1.1.1 Question Answering on Multi-relational Structured Tables

Effectively retrieving patient-specific information or corhort based statistic information from EHR database is significantly important to assist doctors with their decision making. One intuitive way is to directly return the corresponding answer to a given question about the EHR database. This will enable doctors to make complete use of the patient information flexibly without seeking professional assistant about database. This type of question answering on database can be effectively transformed to the Question-to-SQL generation problem, which

aims at translating a given question to its corresponding SQL query that can be easily used to obtain the answer based on the database. Recently, the Question-to-SQL generation problem is mainly solved with two types of methods, (1) slot-filling methods, which make use of the table schema and the semantic and syntactic information in a question to generate a predefined logic form; (2) language generation methods, which generate SQL queries directly by applying language generation models on questions. However, these existing methods are not able to handle the challenges in Question-to-SQL on EHR data, such as the abbreviation and typos in healthcare terminology and the table unaware assumption. Therefore, one of the goals for this research is to propose a model that can simultaneously tackle the above challenges. In addition, to the best of our knowledge, there is no existing dataset in healthcare domain for performing question answering on multi-relational structured tables. Therefore, we will first create a benchmark dataset for the Question-to-SQL generation task in healthcare.

1.1.2 Question Answering on Unstructured Clinical Notes

In EHR data, unstructured clinical notes mainly contain plain-text patient information and tend to include more information that is not provided in structured tables, such as the narrative description of chief complaint, allergies and history of present illness. These textual descriptions also provide important information to guide the decision making in clinical practice. For example, to test the safety and effectiveness of new medications or treatments in clinical trials, this textual information can provide a comprehensive source to guide the selection of appropriate participating patients. However, it is a challenging issue to identify the patients who satisfy certain conditions based on the original clinical notes. In most of the existing question answering works [76, 141], a question is only related to a single document and the answer to the question is explicitly included in the original text. However, to identify all qualified patients given certain conditions, it requires going through the clinical notes of all patients and checking the matching status of each patient with a certain extent of reasoning. Therefore, another goal of this research is to perform question answering on unstructured clinical notes to identify answers for given questions with various conditions.

Knowledge graph makes it practical to integrate the large-scale patient information together by building relationships between different patients and clinical notes. Therefore, to handle the complex questions, we will create a knowledge base from clinical notes to link different patients and clinical notes and perform knowledge base question answering (KBQA). In this case, the unstructured clinical notes will be transformed into the structured knowledge graphs of various patient events, and the question answering task will be performed by querying knowledge graphs. The relationships between entities in knowledge graph are commonly represented using a distributed triple format, which limits the application of knowledge graph [44, 70]. Therefore, we will learn continuous low-dimensional embedding of knowledge graph by considering various contextual information and apply the learned embedding in the downstream tasks, including question answering and link prediction.

1.1.3 Discovering Complex Relationships of Entities in EHR

Typically, the medical information of different patients in EHR data is organized independently, which makes it challenging to identify the complex relationships between different patients, diseases, and treatments. However, this complex relationship information is significantly helpful for doctors to provide personalized precision medical treatment. For example, when predicting the treatment for a patient, besides his/her own medical information, the treatment provided to other similar patients can also provide important guidelines. Therefore, we propose to leverage the power of knowledge graph to link the distributed patient information in EHR data as a heterogeneous network and further analyze their complex higher-order relationships through link prediction task. Besides, representation learning methods for heterogeneous networks produce a low-dimensional vector embedding (that is typically fixed for all tasks) for each node. Many of the existing methods focus on obtaining a static vector representation for a node in a way that is agnostic to the downstream application where it is being used. In practice, however, downstream tasks such as link prediction require specific contextual information that can be extracted from the subgraphs related to the nodes provided as input to the task. To tackle this challenge, we will develop a deep learning-based framework for bridging static representation learning methods using global information from the entire graph with localized attention driven mechanisms to learn contextual node representations.

1.1.4 Temporal Multi-task Survival Analysis

Survival analysis aims at predicting the time to event of interest along with its probability on longitudinal data. It is commonly used to make predictions for a single specific event of interest at a given time point. However, predicting the occurrence of multiple events of interest simultaneously and dynamically is needed in many real-world applications. An intuitive way to solve this problem is to simply apply the standard survival analysis method independently to each prediction task at each time point. However, it often leads to a sub-optimal solution since the underlying dependencies between these tasks are ignored. This motivates us to analyze these prediction tasks jointly to select the common features shared across all the tasks. Therefore, the goal of this work is to formulate a temporal (multiple time points) multi-task learning framework that allows us to involve both the underlying dependencies between different events and the correlations of each single event over time. It will also help handle the issue of the insufficiency of event occurrences in the single task survival analysis problems.

1.2 Contributions

The main research contributions of this dissertation are listed below.

Question Answering on Multi-relational Structured Tables:

- **Proposing a model for question-to-SQL generation model**. It consists of three main components: (1) Translating an input question to a SQL query using a Seq2Seq based model, (2) Editing the generated query with attentive-copying mechanism, and (3) Further editing it with task-specific look-up tables.
- **Creating a large-scale dataset for Question-to-SQL task in healthcare domain**. The new created MIMICSQL data has two subsets: (1) template questions (machine generated), and (2) natural language questions (human annotated).
- **Conducting extensive experiments for performance evaluation**. The evaluation was conducted on MIMICSQL dataset for both template questions and natural language questions to demonstrate the effectiveness of the proposed model. Both qualitative and quantitative results indicate that it outperforms several baseline methods.

Question Answering on Unstructured Clinical Notes:

- **Creating a dataset for knowledge base question answering (KBQA) in healthcare domain**. This ClinicalKBQA dataset consists of two sets: (1) ClinicalKB, which is a comprehensive clinical knowledge base created based on the expert annotations in n2c2 dataset, and (2) QA pairs, which is a large-scale question answering dataset on ClinicalKB.
- **Developing an attention-based aspect-level reasoning model for KBQA**. It is an embedding-based method that incorporates an attention mechanism between question representations and aspect-level answer candidate representations for calculating matching scores of candidate answers.
- **Analyzing the effectiveness of the proposed model and the impact of answer aspects.** We conducted experimental analysis on ClinicalKBQA dataset to demonstrate the effectiveness of AAR model and analyzed the significance of different aspects in providing accurate answers.

Discovering Complex Relationships of Entities in EHR:

- **Proposing a self-supervised learning framework for contextual embedding learning for graphs.** It learns higher-order semantic associations between nodes by simultaneously capturing the global and local factors that characterize a context subgraph.
- **Conducting an extensive experiments for performance evaluation**. We used several public benchmark network datasets in various applications, and introduced

a new healthcare knowledge graph from the publicly available real-world EHR data. We compare the proposed model with the existing static and contextual embedding learning methods using standard evaluation metrics for the task of link prediction.

- **Demonstrating the effectiveness of the proposed model**. We provided detailed analysis about the interpretability, effectiveness of contextual translation, and the scalability of SLiCE.

Temporal Multi-task Survival Analysis:

- **Formulating a temporal multi-task learning problem using tensor representation**. Given a survival dataset and a sequence of time points, which are considered as the monitored time points for the events of interest, we reformulate the survival analysis problem to jointly handle each task at each time point and optimize them simultaneously.
- **Proposing a model for temporal multi-task survival analysis.** It can analyze several survival problems jointly and dynamically predict the survival probability at each time point. The model is optimized with ADMM based algorithm.
- **Conducting extensive experiments for model evaluation.** We evaluate the performance of the proposed temporal multi-task learning method on important real-world datasets, compare with several state-of-the-art methods, and demonstrate the interpretability of the proposed model.

1.3 Organization of the Dissertation

The rest of this dissertation is organized as follows. Chapter 2 introduces the large-scale dataset for Question-to-SQL generation task in the healthcare domain, describes the proposed translate-edit model, and provides performance evaluaiton and experimental analysis. In Chapter 3, we introduce the clinical knowledge base question answering dataset, propose an an attention-based aspect-level reasoning model and analyze the model's effectiveness and impact. Chapter 4 analyzes the complex relationships of entities in heterogeneous network. We propose a self-supervised learning framework for contextual embedding learning, and demonstrate its effectiveness on both healthcare network and several other public benchmark network datasets. In Chapter 5, we propose a temporal multi-task learning model to analyze several survival analysis problems jointly and dynamically, and we discuss the model performance through extensive experiments. In Chapter 6, we summarize our work and discuss the future work.

Chapter 2

Text-to-SQL Generation for Question Answering on Electronic Health Records

This chapter presents a novel method for Question-to-SQL generation along with the way we created the MIMICSQL dataset based on the Electronic Health Records. First, the introduction to this work is provided in Section 2.1. Section 2.2 describes some prior work related to Question-to-SQL generation, and differentiate our work from other existing works. Section 2.3 provides a comprehensive description of the MIMICSQL data generation process. Section 2.4 provides the details of the proposed translate-edit model. Section 2.5 shows the comparison of our proposed model with the state-of-the-art methods by analyzing both quantitative and qualitative results. Finally, we conclude the chapter in Section 2.6.

2.1 Introduction

Due to recent advances of data collection and storing techniques, a large amount of healthcare related data, typically in the form of electronic medical records (EMR), are accumulated every day in clinics and hospitals. EMR data contains a comprehensive set of longitudinal information about patients and are usually stored in structured databases with multiple relational tables, such as demographics, diagnosis, procedures, prescriptions, and laboratory tests. One important mechanism of assisting doctors' clinical decision making is to directly retrieve patient information from EMR data, including patient-specific information (e.g., individual demographic and diagnosis information) and cohort-based statistics (e.g., mortality rate and prevalence rate). Typically, doctors interact with EMR data using searching and filtering functions available in rule-based systems that first turn any predefined-rule (front-end) to a SQL query (back-end), and then, return an answer. These systems are complicated and

require special training before being used. They are also difficult to manage and extend. For example, the front-end needs to be adapted for newer functionalities. Therefore, doctors who depend on these systems cannot fully and freely explore EMR data. Another challenge for these systems is that the users have to first transform their questions to a combination of rules in the front-end, which is not convenient and efficient. For instance, if a doctor wants to know the number of patients who are under the age of 40 and suffering from diabetes, then, he/she may have to create two filters, one for disease and the other for age. An alternate way to solve this problem is to build a model that can translate this question directly to its SQL query, so that the doctor only needs to type his/her question as: "Give me the number of patients whose age is below 40 and have diabetes", in the search box to get the answer. Motivated by this intuition, we propose a new deep learning based model that can translate textual questions on EMR data to SQL queries (*Text-to-SQL generation*) without any assistance from a database expert. As a result, these systems can assist doctors with their clinical decisions more efficiently. Since the textual input in our task is a clinical question, we will refer to Text-to-SQL generation as *Question-to-SQL generation* from now onwards.

Recently, Question-to-SQL generation task has gained significant attention and found applications in a variety of domains, including WikiSQL [30, 151, 129] for Wikipedia, ATIS [52] about flight-booking, GeoQuery [36] about US geography, and Spider [140] for general purpose cross-domain applications. There are also few works in this line of research in healthcare domain [9, 87]. Broadly speaking, various approaches for Question-to-SQL generation task belong to one of the following two categories: (1) *Semantic parsing or slot-filling methods* [129, 138, 30, 134, 41, 139]: These models make use of semantic and syntactic information in a question and a table schema to generate a SQL logic form (parsing tree), which can be easily converted to the corresponding executable query. However, they strongly depend on pre-defined templates, which limits their applications in generating complex SQL queries. (2) *Language generation methods* [151, 120, 146]: These models leverage the power of language generation models and can directly generate SQL queries without building pre-defined SQL templates [74]. Therefore, they can be easily applied to produce complex SQL queries, regardless of the number of tables and columns involved. However, the predicted SQL queries may not be executable due to limited and inaccurate information from questions (e.g., a random question with typos and missing keywords). Moreover, it is difficult to interpret language generation models and their outputs.

Many recent Question-to-SQL models have been primarily benchmarked on WikiSQL [30, 151, 129] and Spider [41, 139, 10] datasets. In WikiSQL, questions are asked against databases with a single table, and every SQL query is composed of a "SELECT" (column/aggregation) clause along with a "WHERE" (condition) clause that consists of one or multiple conditions. Different from WikiSQL, the Spider dataset contains lots of complex and nested queries (e.g., "GROUP BY" and "HAVING") which may involve multiple tables [140]. Some recent studies have shown that several models which perform well on WikiSQL achieve poor results on Spider [146, 41]. It indicates that models for Question-to-SQL generation on single-table databases cannot be simply adapted to database with multiple relational tables. For Spider

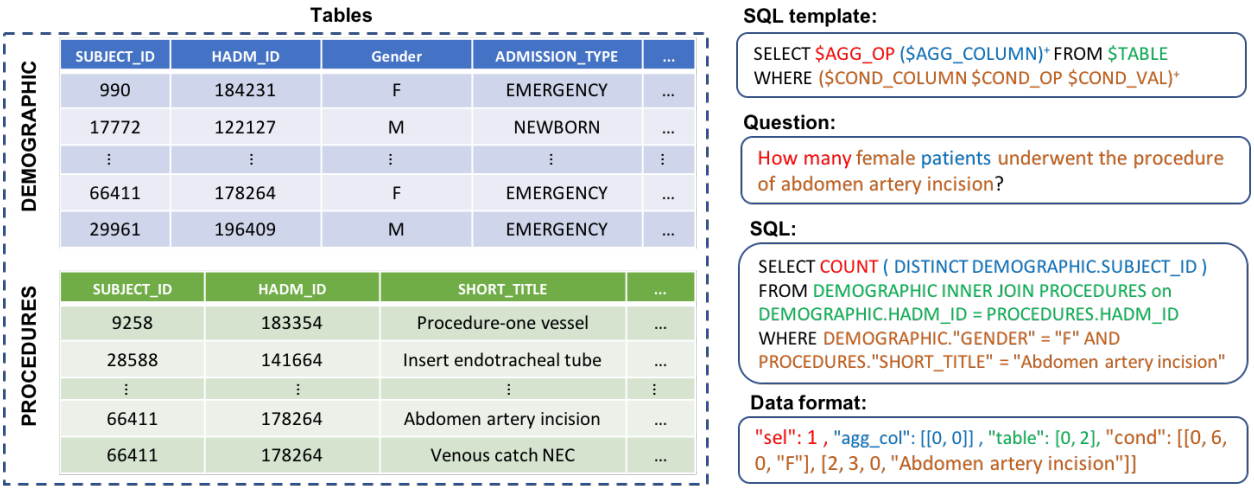


Figure 2.1: An example from MIMICSQL. The two tables, namely, Demographics and Diagnoses, are used to answer the question. Different colors are used to show the correspondence between various components in source question, targeted SQL query, and SQL template.

dataset, the current Question-to-SQL generation task focuses on generating SQL queries without actual values for "WHERE" conditions, which means models are only required to predict SQL structures and parse corresponding table and column names. However, even if a model can produce high quality SQL structures and columns, condition value generation may still be the bottleneck in producing correct and executable SQL queries [134]. Another issue with WikiSQL and Spider dataset is that most words (78% for WikiSQL and 65% for Spider) in database schema in development/testing sets have appeared in the training set [41]. Therefore, it is not feasible to apply the models trained on the Spider dataset to some other domains like chemistry, biology, and healthcare. Specific to healthcare domain, Question-to-SQL generation for EMR data is still under-explored. There are three primary challenges: (1) **Medical terminology abbreviations.** Due to the wide use of abbreviation of medical terminology (sometimes typos), it is difficult to match keywords in questions to those in database schema and table content. (2) **Condition value parsing and recovery.** It is still a challenging task to extract condition values from questions and recover them based on table content, especially in the appearance of medical abbreviations. (3) **Lack of large-scale healthcare Question-to-SQL dataset.** Currently, there is no dataset available for the Question-to-SQL task in the healthcare domain.

To tackle these challenges, we first generated a large-scale healthcare Question-to-SQL dataset, namely MIMICSQL, that consists of 10,000 Question-SQL pairs, by using the publicly available real-world Medical Information Mart for Intensive Care III (MIMIC III) dataset [55, 39] and leveraging the power of *crowdsourcing*. An illustrative example in MIMICSQL is provided in Figure 2.1 to illustrate various components of the dataset. Based on MIMICSQL data, we further propose a language generation based Translate-Edit model, which can first translate a question to the corresponding SQL query, and then, retrieve

condition values based on the question and table content. The editing meta-algorithms make our model more robust to randomly asked questions with insufficient information and typos, and make it practical to retrieve and recover condition values effectively. The major contributions of this work are as follows:

- Propose a two-stage **TR**anslate-**E**dit Model for **Q**uestion-to-**S**QL (TREQS) generation model, which consists of three main components: (1) Translating an input question to a SQL query using a Seq2Seq based model, (2) Editing the generated query with attentive-copying mechanism, and (3) Further editing it with task-specific look-up tables.
- Create a large-scale dataset for Question-to-SQL task in healthcare domain. MIMICSQL has two subsets, in which the first set is composed of template questions (machine generated), while the second consists of natural language questions (human annotated). To the best of our knowledge, it is the first dataset for healthcare question answering on EMR data with multi-relational tables.
- Conduct an extensive set of experiments on MIMICSQL dataset for both template questions and natural language questions to demonstrate the effectiveness of the proposed model. Both qualitative and quantitative results indicate that it outperforms several baseline methods.

2.2 Related Work

Question-to-SQL generation is a sub-task of semantic parsing, which aims at translating a natural language text to a corresponding formal semantic representation, including SQL queries, logic forms and code generation [130, 29]. It has attracted significant attention in various applications, including WikiSQL [151, 129] for Wikipedia, ATIS [52] about flight-booking, GeoQuery [36] about US geography and Spider [140] about cross-domain. In the literature of Question-to-SQL generation, a common way is to utilize a SQL structure-based sketch with multiple slots and formulate the problem as a slot filling task [30, 151, 129, 138, 92] by incorporating some form of pointing/copying mechanism [117]. Seq2SQL method [151] is an augmented pointer network-based framework and mainly prunes the output space of the target query by leveraging the unique structures of SQL commands. SQLNet method [129] is proposed to avoid the "order-matter" problem in the condition part by using a sketch-based approach instead of the sequence-to-sequence (Seq2Seq) based method. By further improving SQLNet, TYPESQL method [138] captures the rare entities and numbers in natural language questions by utilizing the type information. The two-stage semantic parsing method named Coarse2Fine [30] first generates a sketch of a given question and then fills in missing details based on both input question and the sketch. Recently, several semantic parsing methods [41, 139, 10] are also proposed on Spider to tackle the problem across different domains. One limitation of these methods is that they are highly dependent on

the SQL structure and the lexicons, and thus cannot efficiently retrieve the condition values. Therefore, compared to other components, the performance of most semantic parsing methods in predicting condition values tend to be relatively low and these methods primarily focus on predicting correct SQL structures and columns, especially for the cross-domain problem present in the recently released Spider data [140].

To overcome the disadvantage of slot filling methods, Seq2Seq based methods [102, 29, 119, 74] are proposed to tackle this challenge by directly generating the targeted SQL queries. More specifically, Seq2Seq based methods first encode input questions into vector representations and then decode the corresponding SQL conditioned on the encoded vectors. A type system of SQL expressions is applied in the deep Seq2Seq model in [119] to guide the decoder to either directly generate a token from the vocabulary or copy it from the input question. The table schema and the input question are encoded and concatenated as the model input. In contrast, the column names are encoded independently from the encoding of questions in [69], which extended the pointer-generator in SQL generation when the order of conditions in SQL query does not matter. In [74], a unified question-answering framework was proposed to handle ten different natural language processing tasks, including WikiSQL semantic parsing task. To perform question answering on databases with multiple relational tables, there are some other works that aim at guiding the SQL generation indirectly using the answers obtained by query execution [136, 78, 135] or accomplish the goal by directly identifying the correct table cells corresponding to the question answers [99, 42].

Both semantic parsing and language generation approaches show great efficiency in the existing application domains. However, the Question-to-SQL generation task in healthcare domain is still under-explored. There are some efforts in directly seeking answers from unstructured clinical notes to assist doctors with their clinical decision making [137, 63]. However, these problems are significantly different from our task of answering natural language questions on structured EMR data since in our task, the answers to the questions may not be directly included in the structured data. For example, instead of directly retrieving answers, a certain extent of reasoning is required to answer the counting questions starting with "*how many*". There are a few research efforts in solving the Question-to-SQL generation tasks in healthcare domain using semantic parsing and named entity extraction [9, 87]. Due to the domain-specific challenges and the lack of large-scale datasets for model training, there are still several challenges for the Question-to-SQL generation in healthcare. For example, due to the commonly occurring abbreviations of healthcare terminology in EMR data and potential typos in questions, it is possible that the keywords provided in questions are not exactly the same ones used in the EMR data. Therefore, besides predicting the SQL structure and columns, one important task in healthcare is correctly predicting condition values in order to ensure the accuracy of query results for input questions. These challenges motivate us to develop a model that can tackle these issues specifically in healthcare. To train and test our model, we also create the MIMICSQL dataset, which consists of Question-SQL pairs based on MIMIC III dataset. This is the first work that focuses on the Question-to-SQL generation on the healthcare databases with multiple relational tables.

2.3 MIMICSQL Dataset Creation

To the best of our knowledge, there is no existing dataset for Question-to-SQL generation task in the healthcare domain. In this section, we provide a detailed illustration of Question-SQL pair generation for Question-to-SQL tasks on EMR data.

2.3.1 MIMIC III Dataset

To ensure both the public availability of the dataset and the reproducibility of the results for Question-to-SQL generation methods, the widely used Medical Information Mart for Intensive Care III (MIMIC III) dataset [55, 39] is used in this work to create the Question-SQL pairs. Typically, the healthcare related patient information is grouped into five categories in healthcare literature, including demographics (Demo), laboratory tests (Lab), diagnosis (Diag), procedures (Pro), and prescriptions (Pres). We extracted patient information and prepared a specific table for each category separately. These tables compose a relational patient database where tables are linked through patient ID and admission ID as shown on the top of Figure 2.1.

2.3.2 MIMICSQL Generation

Based on the aforementioned five tables, we create the MIMICSQL dataset, including the Question-SQL pairs along with the logical format for slot filling methods, specifically for such Question-to-SQL generation task. Figure 2.1 provides an overview of basic components used for MIMICSQL generation. Due to the large amount of information included in EMR database, it is challenging and time-consuming for domain experts to manually generate the Question-SQL pairs. It should be noted that, for machine generated questions, there exists some drawbacks, including not being natural compared to questions provided by humans and usually are not grammatically accurate. In this work, we take advantage of both human and machine generation to collect the Question-SQL pairs for the MIMICSQL dataset in the following two steps. Figure 2.2 shows the flowchart of the MIMICSQL generation.

Machine Generation of Questions

Following the question types used in [56], there are two types of questions in MIMICSQL, including retrieval questions and reasoning questions. Following the generation of question templates in [76], we first identify the questions that are possibly asked on the EMR data and then normalize them by identifying and replacing the entities regarding table headers, operations, and condition values with generic placeholders. The question templates for retrieval and reasoning questions are finally integrated into two generic templates. These

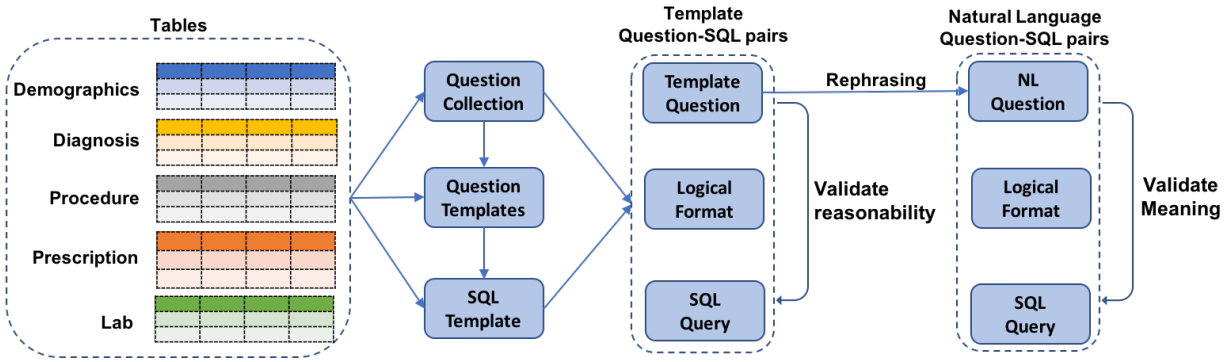


Figure 2.2: The generation framework of our MIMICSQL dataset for Question-to-SQL task using MIMIC III dataset.

question templates provide a guidance regarding the question topics or perspectives for the machine generated questions.

1. **Retrieval questions** are designed to directly retrieve specific patient information from tables. The two generic templates mainly used for retrieval questions include:
   - What is the H_1 and H_2 of Patient Pat (or Disease D, or Procedure Pro, or Prescription Pre, or Lab test L)?
   - List all the Patients (or Disease, or Procedures, or medications, or lab tests) whose H_1 O_1 V_1 and H_2 O_2 V_2.
2. **Reasoning questions** are designed to indirectly collect patient information by combining different components of five tables. The templates mainly used for reasoning questions include:
   - How many patients whose H_1 O_1 V_1 and H_2 O_2 V_2?
   - What is the maximum (or minimum, or average) H_1 of patient whose H_2 O_2 V_2 and H_3 O_3 V_3?

Here, H_i, O_i, V_i represent placeholders for the ith table column used in the question, its corresponding operation and condition value, respectively. To avoid complicated query structure, the number of conditions in each question cannot exceed a pre-defined threshold, which is set to be 2 in this work.

During question generation, the corresponding SQL query for each question is also generated simultaneously. To respond to all questions without changing the query structure and facilitate the prediction of SQL for Question-to-SQL models, we adopt a general **SQL template** SELECT \$AGG_OP (\$AGG_COLUMN)$^+$ FROM \$TABLE WHERE (\$COND_COLUMN \$COND_OP \$COND_VAL)$^+$. Here, the superscript "+" indicates that it allows one or

more items. *AGG_OP* is the operation used for the selected AGG_COLUMN and takes one of the five values, including "NULL" (representing no aggregation operation), "COUNT", "MAX", "MIN" and "AVG". *AGG_COLUMN* is the question topic that we are interested in each question and is stored as the column header in tables. Since it is possible for a given question to be related to more than one table, *TABLE* used here can be either a single table or a new table obtained by joining different tables. The part after WHERE represents the various conditions present in the question and each condition takes the form of ($COND_COLUMN $COND_OP $COND_VAL). During query generation, we mainly consider five different condition operations, including "=", ">", "<", ">=" and "<=".

Natural Language Question Collection

These machine generation criteria make it practical to effectively obtain a set of Question-SQL pairs, however, there are two main drawbacks for the machine generated template questions. On the one hand, the questions may not be realistic in the clinical practice. For example, the unreasonable question "*How many patients whose primary disease is newborn and marital status is married?*" will also be generated. On the other hand, the template questions tend to be not as natural as questions asked by doctors since they follow a fixed structure provided in the question templates. To overcome these drawbacks, we recruited eight Freelancers with medical domain knowledge on a *crowd-sourcing platform named Freelancer*[1] to filter and paraphrase the template questions in three steps: (1) To ensure that the generated questions are realistic in the healthcare domain, each machine generated question is validated to ignore the unreasonable template questions. (2) Each selected template question is rephrased as its corresponding natural language (NL) question. (3) The rephrased questions are further validated to ensure that they share the same meaning as the original template questions.

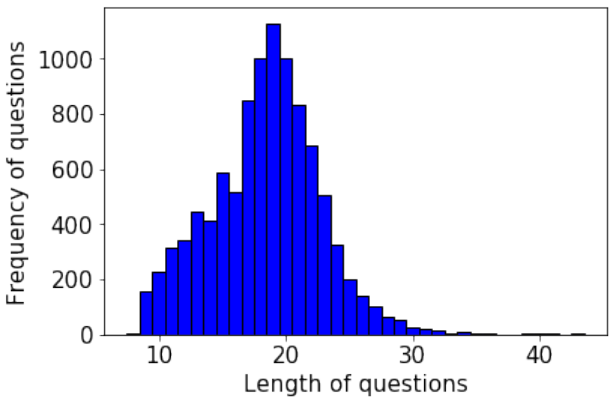
2.3.3 MIMICSQL Statistics

MIMICSQL dataset is publicly available at[2]. We include 10,000 Question-SQL pairs in MIMICSQL whose basic statistics are provided in Figure 2.3 and Table 2.1. Figure 2.3(a) and Figure 2.3(b) shows the distributions of the question length for template questions and natural language questions, respectively. The distribution of the SQL length is given in Figure 2.3(c). Figure 2.3(d) shows the distribution of number of questions over five tables. Note that the total number of questions in Figure 2.3(d) is more than 10,000 since some questions are related to more than one table.

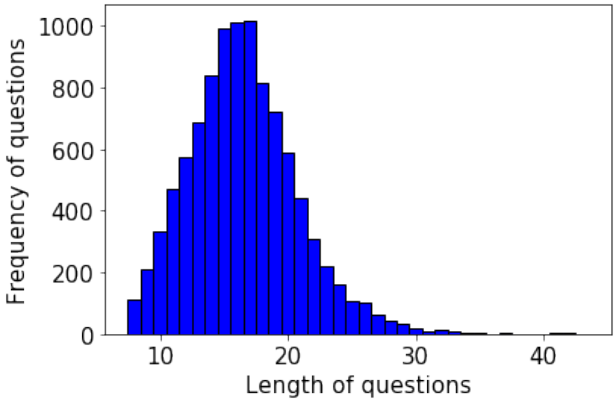
[1] www.freelancer.com
[2] https://github.com/wangpinggl/TREQS

Table 2.1: Statistics of MIMICSQL dataset. The tables are in the order of Demographics, Diagnosis, Procedure, Prescriptions, and Laboratory tests.

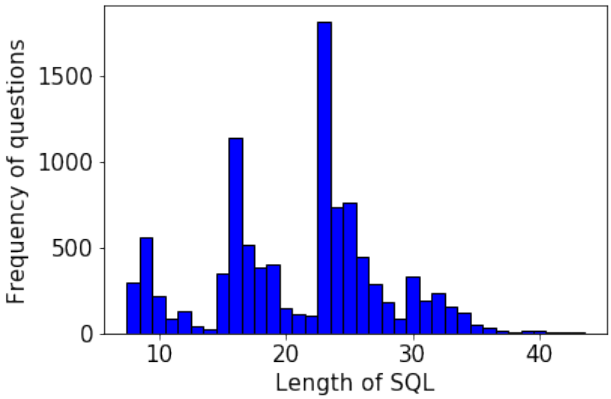
Data	**Value**
# of patients	46,520
# of tables	5
# of columns in tables	23/5/5/7/9
# of Question-SQL pairs	10,000
Average template question length (in words)	18.39
Average NL question length (in words)	16.45
Average SQL query length	21.14
Average aggregation columns	1.10
Average conditions	1.76



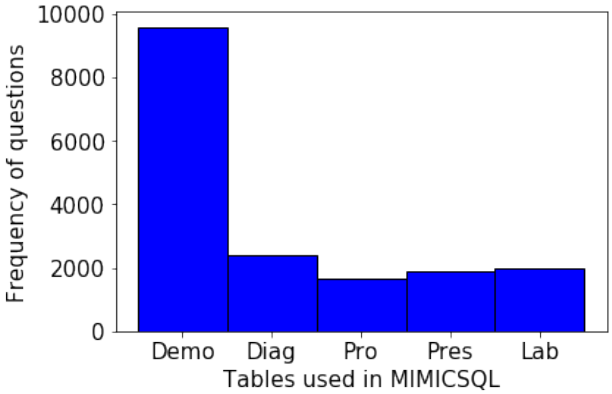
(a) Dist. of length of template questions.



(b) Dist. of length of NL questions.



(c) Dist. of length of SQL queries.



(d) Dist. of No. of Questions.

Figure 2.3: Distribution of questions and queries in MIMICSQL dataset. "Dist." is used as an acronym for "Distribution".

2.4 A Translate-Edit Model for Question-to-SQL Query Generation

In this section, we will first formulate the Question-to-SQL query generation problem. Then, we present our TREQS model in detail.

2.4.1 Problem Formulation

In this work, we aim to translate healthcare related questions asked by doctors to database queries and then retrieve the answer from health records. We adapt the language generation approach in our model, since questions may be related to a single table or multiple tables, and keywords in the questions may not be accurate due to the healthcare terminology involved. To tackle the challenges for general applications, we propose a translate-edit model that first generates a query draft using a language generation model and then edits based on the table schema.

Let us denote a given question by $x = (x_1, x_2, ..., x_J)$, the table schema context information as z and the corresponding query as $y = (y_1, y_2, ..., y_T)$, where J and T represents the length of the input and output, respectively. x_j and y_t denote the one-hot representations of the tokens in the question and query, respectively. Then, the goal of our model is to infer y from x based on z with probability $P(y|x, z)$. In our approach, we assume that the table schema information z is implicitly included in the input questions as semantic information. Therefore, during the translation, we only need to deal with inferring y from x. However, since the exact table schema has not appeared at this stage, the generated query can only roughly capture this information. At the second stage, we edit the query draft based on the table schema and look-up tables of content keywords to recover the exact information. This two-stage strategy allows us to easily adapt our model to other general purpose tasks. In the following sections, we will introduce our model layer-by-layer in more detail.

2.4.2 The Proposed TREQS Model

Now we introduce the details of the three components in the proposed **TR**anslate-**E**dit Model for **Q**uestion-to-**S**QL (**TREQS**) generation. Figure 2.4 shows the framework of the proposed model.

Sequence-to-Sequence Framework

We adopt a RNN sequence-to-sequence (Seq2Seq) framework for the Question-to-SQL generation task. Our Seq2Seq framework is composed of a question encoder (a single-layer

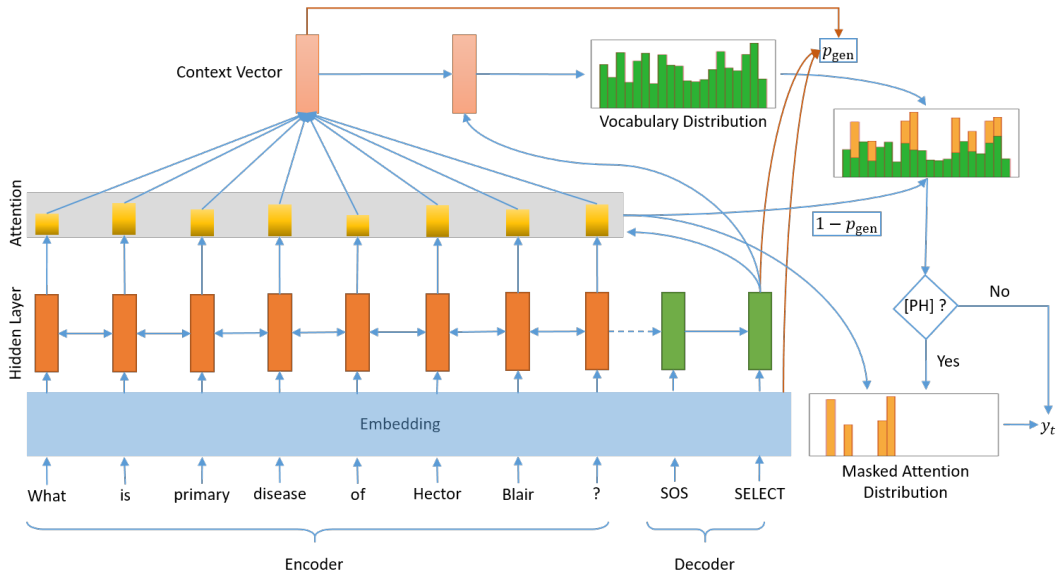


Figure 2.4: The overall framework of the proposed TREQS model. In this figure, we do not incorporate the temporal attention and attention on decoder mechanisms. [PH] represents the out of vocabulary words in condition values.

bidirectional LSTM [49]) and a SQL decoder (a single-layer unidirectional LSTM). The encoder reads a sequence of word embeddings of input tokens and turns them into a sequence of encoder hidden states (features) $h^e = (h_1^e, h_2^e, \ldots, h_J^e)$, where the superscript e indicates that the hidden states are obtained from the encoder, and $h_j^e = \overrightarrow{h}_j^e \oplus \overleftarrow{h}_{J-j+1}^e$ is the concatenation of the hidden states of forward and backward LSTM. At each decoding step t, the decoder takes the encoder hidden states and word embedding of the previous token as an input and produce a decoder hidden state h_t^d. Both word embeddings in the encoder and decoder are taken from the same matrix W_{emb}. The decoder LSTM hidden and cell states are initialized with

$$\begin{aligned} h_0^d &= \tanh\left(W_{\text{e2dh}}\left(\overrightarrow{h}_J^e \oplus \overleftarrow{h}_1^e\right) + b_{\text{e2dh}}\right) \\ c_0^d &= \tanh\left(W_{\text{e2dc}}\left(\overrightarrow{c}_J^e \oplus \overleftarrow{c}_1^e\right) + b_{\text{e2dc}}\right) \end{aligned} \tag{2.1}$$

where the weight matrices W_{e2dh}, W_{e2dc}, and vectors b_{e2dh}, b_{e2dc} are learnable parameters.

Temporal Attention on Question

At each decoding step t, the decoder not only takes its internal hidden state and previously generated token as input, but also selectively focuses on parts of the question that are relevant to the current generation. However, the standard attention models proposed in the literature [6, 71] cannot prevent the decoder from repetitively attending on the same part of the question, therefore, we adopt a temporal attention strategy [75] that was demonstrated to be effective in tackling such problem.

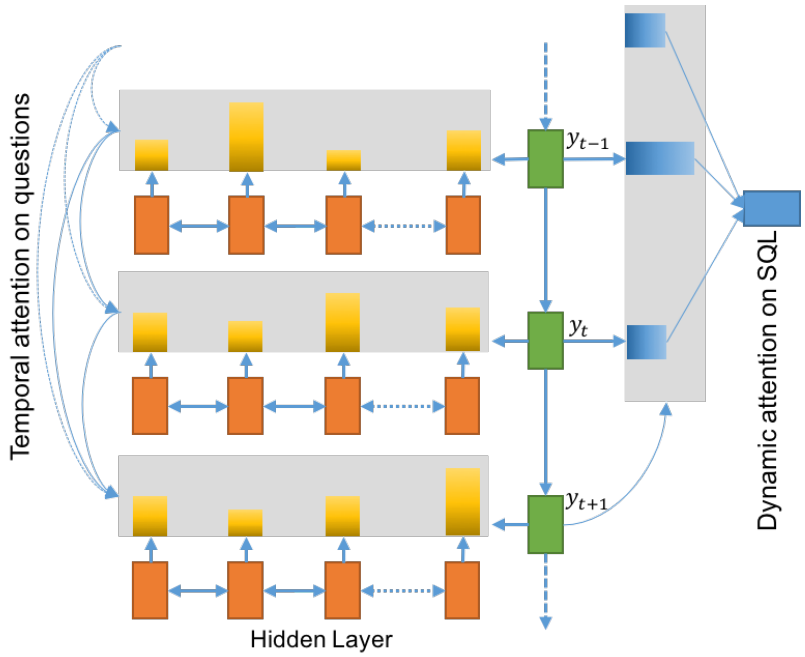


Figure 2.5: Illustration of attention techniques used in TREQS.

To achieve this goal, we first define an alignment score function between the current decoder hidden state and each of the encoder hidden states as follows:

$$s_{tj}^{e} = (h_j^e)^\top W_{\text{align}} h_t^d \tag{2.2}$$

where W_{align} are parameters. As shown in the left-hand side of Figure 2.5, to avoid repetitive attention, we penalize the tokens that have obtained high attention scores in the previous decoding steps with the following normalization rule:

$$s_{tj}^{\text{temp}} = \begin{cases} \exp\left(s_{tj}^{e}\right) & \text{if } t = 1 \\ \frac{\exp\left(s_{tj}^{e}\right)}{\sum_{k=1}^{t-1} \exp\left(s_{kj}^{e}\right)} & \text{if } t > 1 \end{cases}, \quad \alpha_{tj} = \frac{s_{tj}^{\text{temp}}}{\sum_{k=1}^{J} s_{tk}^{\text{temp}}} \tag{2.3}$$

where s_{tj}^{temp} is the new alignment score with temporal dependency, and α_{tj} is an attention weight at current decoding step. With the temporal attention mechanism, we finally obtain a context vector for the input question as follows:

$$z_t^e = \sum_{j=1}^{J} \alpha_{tj} h_j^e. \tag{2.4}$$

Dynamic Attention on SQL

In our Question-to-SQL generation task, different parts of a query may not strictly have sequential dependency. For example, switching two conditions in a query will yield the same

query. However, when generating the condition values, the decoder may need to not only take the previously generated token, its own hidden states and encoder context vector into consideration, but also places more attention on the previously generated table names and headers as shown in the right-hand side of Figure 2.5. Therefore, we introduce a dynamic attention mechanism to the decoder [93, 91], which allows it to dynamically attend on the previous generated tokens.

More formally, for $t > 1$, the alignment scores (denoted by $s^d_{t\tau}$, $\tau \in \{1, ..., t-1\}$) on previously generated tokens can be calculated in the same manner as the alignment scores for the encoder. Then, the attention weight for each token is calculated as follows:

$$\alpha^d_{t\tau} = \frac{\exp(\mathrm{s}^d_{t\tau})}{\sum_{k=1}^{t-1} \exp(\mathrm{s}^d_{tk})} \tag{2.5}$$

With the attention distribution and the decoder hidden states, we can calculate the decoder-side context vector as follows:

$$z^d_t = \sum_{\tau=1}^{t-1} \alpha^d_{t\tau} h^d_\tau \tag{2.6}$$

Controlled Generation and Copying

A Question-to-SQL generation task is very different from the general purpose language generation tasks. First, there are strict templates for SQL queries. For example, SELECT \$AGG_OP (\$AGG_COLUMN)$^+$ FROM \$TABLE WHERE (\$COND_COLUMN \$COND_OP \$COND_VAL)$^+$ is the template we used. Second, the aggregation and condition columns in queries are table headers, which usually do not exactly appear in the questions. For instance, for a given question: "How many patients who have bowel obstruction and stay in hospital for more than 10 days?", its corresponding query looks like "SELECT COUNT (PATIENT_ID) FROM DEMOGRAPHIC WHERE PRIMARY_DISEASE = bowel obstruction AND DAYS_OF_STAY > 10". Obviously, we cannot find words, like PATIENT_ID, PRIMARY_DISEASE, and DAYS_OF_STAY, in the question. Third, the values of conditions should be best possibly retrieved from questions, such as "bowel obstruction" and "10" in the above example, since the questions may contain terms that are out-of-vocabulary (OOV).

Because of these characteristics, our decoder combines a generation network and a pointer network [117] for the token generation. The pointer network has been widely used in language modeling and generation tasks, such as abstractive text summarization [90] and question-answering [74], due to its ability of copying OOV tokens in the source and context sequences to the target sequences. However, in our model, it is primarily used for generating the words in-vocabulary and putting placeholders, denoted as [PH], for OOV words. Intrinsically, it is only used in generating condition values in SQL queries. Formally, to generate a token at step t, we first calculate the probability distribution on a vocabulary $\mathcal{V}$ as follows:

$$\begin{aligned} \tilde{h}_t^d &= W_z\big(z_t^e \oplus z_t^d \oplus h_t^d\big) + b_z \\ P_{\mathcal{V},t} &= \text{softmax}\Big(W_{\text{emb}}(W_{\text{d2v}}\tilde{h}_t^d + b_{\text{d2v}})\Big) \end{aligned} \tag{2.7}$$

where W_z, W_{d2v}, b_z, and b_{d2v} are parameters. We reuse the syntactic and semantic information contained in the word embedding matrix in token generation. Then, combining with the pointer mechanism, the probability of generating a token y_t is calculated by

$$P(y_t) = p_{\text{gen},t}P_{\text{gen}}(y_t) + (1 - p_{\text{gen},t})P_{\text{ptr}}(y_t) \tag{2.8}$$

where the probability $P_{\text{gen}}(y_t)$ given by the generation network is calculated as follows:

$$P_{\text{gen}}(y_t) = \begin{cases} P_{\mathcal{V},t}(y_t) & y_t \in \mathcal{V} \\ 0 & \text{otherwise} \end{cases} \tag{2.9}$$

The probability $P_{\text{ptr}}(y_t)$ by the pointer network is obtained with the following attention distribution

$$P_{\text{ptr}}(y_t) = \begin{cases} \sum_{j:x_j=y_t} \alpha_{tj}^e & y_t \in \mathcal{X} \cap \mathcal{V} \\ 0 & \text{otherwise} \end{cases} \tag{2.10}$$

where $\mathcal{X}$ is a set with all tokens in a question. $p_{\text{gen},t}$ is a 'soft-switch' (probability) of using a generation network for token generation

$$p_{\text{gen},t} = \sigma(W_{\text{gen}}z_t^e \oplus h_t^d \oplus E_{y_{t-1}} + b_{\text{gen}}) \tag{2.11}$$

where $E_{y_{t-1}}$ is the word embedding of the previous token y_{t-1}. W_{gen} and b_{gen} are model parameters. Note that all OOV words in the question have been replaced with the placeholder [PH] for the condition values. In our model, the vocabulary is a union of two sets, i.e., vocabulary of regular tokens and a vocabulary of template keywords as well as table names and headers, denoted as $\mathcal{V}_{\text{schema}}$. Since $\mathcal{X} \cap \mathcal{V}_{\text{schema}} = \emptyset$, the template, table names and headers in a SQL rely only on the generation network. On the other hand, keywords of the condition values and placeholder are obtained from both generation and pointer networks. Note that we always switch the option of [PH] in Figure 2.4 to"No" during training.

With the final probability of generating a token y_t, we are ready to define our loss function. In this work, we adopt the cross-entropy loss which tries to maximize the log-likelihood of observed sequences (ground-truth), i.e.,

$$\mathcal{L} = -\log P_\theta(\hat{y}|x) = \sum_{t=1}^{T} \log P_\theta(\hat{y}_t|\hat{y}_{<t}, x) \tag{2.12}$$

where θ denotes all the model parameters, including weight matrices W and biases b. $\hat{y} = (\hat{y}_1, \hat{y}_2, ..., \hat{y}_T)$ represents a ground-truth SQL sequence in the training data and $\hat{y}_{<t} = (\hat{y}_1, \hat{y}_2, ..., \hat{y}_{t-1})$.

Placeholder Replacement

After a query has been generated, we replace each [PH] with a token in the source question. For a [PH] at time step t', the replacement probability is calculated by

$$P_{\text{rps}}(y_{t'}) = \begin{cases} \sum_{j:x_j=y_{t'}} \alpha^e_{t'j} & y_{t'} \in \mathcal{X} - \mathcal{V} \\ 0 & \text{otherwise} \end{cases} \tag{2.13}$$

Here, we implement this technique by applying a mask (0 or 1) on the attention weights (named as masked attention mechanism). This replacement technique can make use of the semantic relationships (captured by attention and decoder LSTM) between previously generated words and their neighboring OOV words. Intuitively, if the model attends word x_j at the step $t-1$, it has a high chance of attending the neighboring words of x_j at step t. This meta-algorithm can be used for any attention based Seq2Seq model.

Recover Condition Values with Table Content

So far, we have used our translate-edit model to translate given questions on a table to the SQL queries without explicitly using any table content and schema. However, we cannot guarantee that all these queries are executable since the condition values in the questions may not be accurate. In the aforementioned example, the doctor may ask "How many patients who have **bowel obstruct** and stay in hospital for more than 10 days?", then, one of the conditions in the SQL is "PRIMARY_DISEASE = **bowel obstruct**". Obviously, we will get a different answer since **bowel obstruct** does not appear in the database. To alleviate this problem, *we propose a condition value recover technique to retrieve the exact condition values* based on the predicted ones. This approach makes use of string-matching metric ROUGE-L [66] (L denotes the longest common sub-sequence) to find the most similar condition value from the look-up table for each predicted one, and then replaces it. In our implementation, we calculate both word- and character-level similarities, i.e., ROUGE-L scores, between two sequences.

2.5 Experiments

In this section, we first introduce the datasets used in our experiments, and then briefly describe the baseline comparison methods, implementation details, and evaluation metrics. Finally, different sets of qualitative and quantitative results are provided to analyze the query generation performance of the proposed model.

2.5.1 Experimental Settings

Dataset Description

We use both template and natural language (NL) questions in MIMICSQL dataset (described in Section 2.3) for evaluation. We first tokenize both source questions and target SQL queries using Spacy package[3]. Then, they are randomly split into training, development and testing sets in the ratio of 0.8/0.1/0.1. To recover the condition values, we also created a look-up table that contains table schema and keywords, i.e., table name, header and keywords of each column. Finally, for template questions in the testing set, we also generated a dataset that has missing information and typos (*testing with noise*) to demonstrate the effectiveness of our condition value recover technique.

Comparison Methods

We demonstrate the superior performance of our TREQS model by comparing it with the following methods. The first two are slot filling methods and generate logic format of queries, while the others produce SQL queries directly.

- **Coarse2Fine model [30]:** It is a two-stage structure-aware neural architecture for semantic parsing. For a given question, a rough sketch of the logical form is first generated by omitting low-level information, such as the arguments and name entities, which will be filled in the second step by considering both the natural language input and the generated sketch.
- **Multi-table SQLNET (M-SQLNET) [129]:** For SQL with multiple conditions, it may have multiple equivalent variants by varying the order of conditions. SQLNET mainly focuses on tackling the unordered property by leveraging the structure-based dependencies in SQL. However, it can only handle questions on a single table under the table-aware assumption. In this work, we implemented a multi-table version of SQLNET for comparison.
- **Sequence-to-Sequence (Seq2Seq) model [71]:** In this model, there is a bidirectional LSTM encoder and a LSTM decoder. To be consistent with this work, we adopt the "general" global attention mechanism described in [71]. The placeholder replacement algorithm is also used in the query generation step to tackle the OOV words problem in this model.
- **Pointer-Generator Network (PtrGen) [90]:** The pointing mechanism is primarily used to deal with the OOV words. Therefore, an extended vocabulary of all OOV words in a batch is built at each training step to encourage the copying of low-frequency words in the source questions, which is different from our model. In our pointer network, we encourage the model to either copy tokens related to the condition values or put placeholders.

[3]https://spacy.io/

Note that the proposed condition value recover mechanism can be combined with different models that directly generate SQL queries, therefore, we also apply it to the results obtained from Seq2Seq and PtrGen to boost their performance. However, it is not applicable to Coarse2Fine and M-SQLNET since their predicted condition values have already been in the look-up table.

Implementation Details

We implemented the proposed TREQS model and M-SQLNET with Pytorch [79]. For all language generation models, the dimension of word embeddings and the size of hidden states (both encoder and decoder hidden states) are set to be 128 and 256, respectively. Instead of using pre-trained word embeddings [80], we learn them from scratch. ADAM optimizer [59] with hyper-parameter $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$ is adopted to train the model parameters. The learning rate is set to be 0.0005 with a decay for every 2 epochs and gradient clipping is used with a maximum gradient norm of 2.0. During the training, we set the mini-batch size to be 16 in all our experiments and run all models for 20 epochs. The development set is used to determine the best model parameters. During the testing, we implement a beam search algorithm for the SQL generation and the beam size is set to be 5. To build the vocabulary, we keep the words with a minimum frequency of 5 in the training set. Thus, the vocabulary size is 2353 and it is shared between the source question and target SQL. In our experiments, both the source questions and SQL queries are truncated to 30 tokens. The implementation of our proposed TREQS method is made publicly available at[4].

Evaluation Metrics

To evaluate the performance of different Question-to-SQL generation models, we mainly adopt the following two commonly used evaluation metrics [151]. (1) **Execution accuracy** is defined as $Acc_{EX} = N_{EX}/N$, where N denotes the number of Question-SQL pairs in MIMICSQL, and N_{EX} represents the number of generated SQL queries that can result in the correct answers [151]. Note that execution accuracy may include questions that are generated with incorrect SQL queries which lead to correct query results. (2) In order to overcome the disadvantage of execution accuracy, **logic form accuracy** [151], defined as $Acc_{LF} = N_{LF}/N$, is commonly used to analyze the string match between the generated SQL query and the ground truth query. Here, N_{LF} denotes the number of queries that match exactly with the ground truth query.

[4]https://github.com/wangpinggl/TREQS

Table 2.2: The SQL prediction performance results using logic form accuracy (Acc_{LF}) and execution accuracy (Acc_{EX}).

	Template Questions				**NL Questions**			
Method	**Development**		**Testing**		**Development**		**Testing**	
	Acc_{LF}	Acc_{EX}	Acc_{LF}	Acc_{EX}	Acc_{LF}	Acc_{EX}	Acc_{LF}	Acc_{EX}
Coarse2Fine	0.298	0.321	0.518	0.526	0.217	0.309	0.378	0.496
M-SQLNET	0.258	0.588	0.382	0.603	0.086	0.225	0.142	0.260
Seq2Seq	0.098	0.372	0.160	0.323	0.076	0.112	0.091	0.131
Seq2Seq + recover	0.138	0.429	0.231	0.397	0.092	0.195	0.103	0.173
PtrGen	0.312	0.536	0.372	0.506	0.126	0.174	0.160	0.222
PtrGen + recover	0.442	0.645	0.426	0.554	0.181	0.325	0.180	0.292
TREQS (ours)	0.712	0.803	0.802	0.825	0.451	0.511	0.486	0.556
TREQS + recover	**0.853**	**0.924**	**0.912**	**0.940**	**0.562**	**0.675**	**0.556**	**0.654**

2.5.2 Experimental Results

Query Generation Performance

Table 2.2 provides the quantitative results on both template questions and NL questions for different methods. The best performing methods are highlighted in bold and the second best performing methods are underlined. It can be observed from Table 2.2 that the Seq2Seq model is the worst performer among all the compared methods due to its poor generating behavior, including factual errors, repetitions and OOV words. PtrGen performs significantly better than Seq2Seq model since it is able to copy words from the input sequence to the target SQL. As seen from the results, it can capture the factual information and handle OOV words more accurately. It works well when most words in the target sequence are copied from the source sequence, which is similar to other problems such as abstractive text summarization task [90, 38]. However, in Question-to-SQL task, most tokens (template, table names and headers) are obtained from generation and only condition values are copied from questions to queries. Therefore, the task discourages copying in general, which causes PtrGen model to produce the condition values by generation instead of copying, thus increasing the chances of making mistakes. Coarse2Fine achieves outstanding performance for the questions on a single table. The limitation of Coarse2Fine is that it cannot handle complex SQL generation, such as queries including multiple tables. However, it still outperforms both Seq2Seq and PtrGen in most of the cases. Compared to Coarse2Fine, the M-SQLNET method considers the dependencies between slots using a dependency graph determined by the intrinsic structure of SQL. It performs significantly better than Seq2Seq and PtrGen on both testing and testing with noise set (in Table 2.3). It also significantly outperforms Coarse2Fine based on the execution accuracy. Compared to all the aforementioned baseline methods, our proposed TREQS model gains a significant performance improvement on both development and testing

Table 2.3: The SQL prediction performance results and their break-down on template testing questions with noise.

Method	**Overall**		**Break-down**					
	Acc_{LF}	Acc_{EX}	Agg_{op}	Agg_{col}	$Table$	Con_{col+op}	Con_{val}	Average
Coarse2Fine	0.444	0.526	0.528	0.528	0.528	0.520	0.444	0.510
M-SQLNET	0.356	0.606	**1.000**	0.953	**0.998**	0.875	0.376	0.840
Seq2Seq	0.157	0.320	0.997	0.862	0.967	0.817	0.206	0.770
Seq2Seq + recover	0.225	0.389	0.999	0.862	0.967	0.817	0.290	0.787
PtrGen	0.301	0.451	0.999	0.988	0.991	0.970	0.309	0.851
PtrGen + recover	0.353	0.498	0.999	0.988	0.991	0.970	0.360	0.862
TREQS (ours)	0.699	0.756	**1.000**	**0.996**	0.995	**0.976**	0.706	0.935
TREQS + recover	**0.872**	**0.907**	**1.000**	**0.996**	0.995	**0.976**	**0.877**	**0.969**

Table 2.4: Accuracy of break-down matching on template questions in MIMICSQL dataset.

Method	**Development**						**Testing**					
	Agg_{op}	Agg_{col}	$Table$	Con_{col+op}	Con_{val}	Average	Agg_{op}	Agg_{col}	$Table$	Con_{col+op}	Con_{val}	Average
Coarse2Fine	0.321	0.321	0.321	0.321	0.298	0.316	0.528	0.528	0.528	0.520	0.518	0.524
M-SQLNet	**1.000**	0.978	0.994	0.876	0.274	0.824	**1.000**	0.956	**0.996**	0.881	0.401	0.847
Seq2Seq	0.999	0.950	0.972	0.761	0.119	0.760	0.999	0.865	0.963	0.818	0.210	0.771
Seq2Seq + recover	0.999	0.950	0.972	0.761	0.163	0.769	0.999	0.865	0.963	0.818	0.296	0.788
PtrGen	0.999	0.991	0.992	**0.979**	0.325	0.857	**1.000**	0.988	0.992	**0.985**	0.381	0.869
PtrGen + recover	0.999	0.991	0.992	**0.979**	0.449	0.882	**1.000**	0.988	0.992	**0.985**	0.433	0.880
TREQS (ours)	**1.000**	**0.999**	**0.995**	0.924	0.719	0.927	**1.000**	0.995	**0.996**	0.980	0.810	0.956
TREQS + recover	**1.000**	**0.999**	**0.995**	0.924	**0.859**	**0.955**	**1.000**	**0.996**	**0.996**	0.984	**0.918**	**0.979**

dataset and 30 percent, on average, more accurate than others.

We have also applied the proposed condition value recover technique to three language generation models. It can be observed that such a heuristic approach can significantly boost the performance of these models. From our experiments, we found that language models fail in many cases because they cannot capture all keywords of condition values. As a result, they are not executable or may yield different answers. Hence, the recover mechanism can correct these errors in the conditions of SQL by making the best use of the look-up table. Moreover, as shown in Table 2.3, after applying some noise to the template testing questions by removing partial condition values or using abbreviations of words, the performance of different models drops. Our TREQS model is affected significantly because it strongly relies on the pointing mechanism to copy keywords of condition values from questions to queries. However, as we can see, the recover mechanism can still correct most of the errors, thus improving the accuracy by more than 20%, which is 13% for the testing set without introducing noise.

Table 2.5: Accuracy of break-down matching on NL questions in MIMICSQL dataset.

Method	**Development**						**Testing**					
	Agg_{op}	Agg_{col}	$Table$	Con_{col+op}	Con_{val}	Average	Agg_{op}	Agg_{col}	$Table$	Con_{col+op}	Con_{val}	Average
Coarse2Fine	0.319	0.313	0.321	0.260	0.214	0.285	0.524	0.490	0.528	0.448	0.413	0.481
M-SQLNet	**0.994**	**0.939**	0.933	0.722	0.080	0.734	0.989	**0.873**	**0.941**	0.749	0.140	0.738
Seq2Seq	0.978	0.872	0.926	0.466	0.137	0.676	0.970	0.696	0.892	0.563	0.239	0.672
Seq2Seq + recover	0.978	0.872	0.926	0.471	0.174	0.684	0.970	0.696	0.892	0.565	0.296	0.684
PtrGen	0.987	0.917	**0.944**	0.795	0.172	0.766	0.987	0.830	0.926	0.824	0.214	0.757
PtrGen + recover	0.987	0.917	**0.944**	0.795	0.236	0.776	0.987	0.830	0.926	0.824	0.235	0.760
TREQS (ours)	0.990	0.912	0.942	**0.834**	**0.574**	**0.850**	**0.993**	0.827	**0.941**	0.841	0.679	0.856
TREQS + recover	0.990	0.912	0.942	**0.834**	**0.694**	**0.873**	**0.993**	0.827	**0.941**	**0.844**	**0.763**	**0.874**

Break-down Generation Performance

In order to further evaluate the performance on each component of SQL query, in Tables 2.3, 2.4 and 2.5, we provide the break-down accuracy results based on SQL query structure, including aggregation operation (Agg_{op}), aggregation column (Agg_{col}), table ($Table$), condition column along with its operation (Con_{col+op}), and condition value (Con_{val}). The results of Coarse2Fine are not provided due to its table-aware assumption and its inability in handling multi-table questions. We can observe that there is no significant difference between these methods on predictions of both aggregation operation and table. Seq2Seq model performs relatively worse on aggregation column and condition column and its operation.

It is easy to observe from Tables 2.2 to 2.5 that the performance of condition value dominates the overall SQL generation performance. Seq2Seq is not able to capture the correct condition values due to its limitation in handling the OOV words. PtrGen performs slightly better since it is able to copy OOV words directly from the input questions, however, it still cannot capture the condition values as accurately as our proposed TREQS model. We believe that this is due to the fact that we consider temporal attention on questions, dynamic attention on SQL and the controlled generation and copying techniques in the proposed model. We can also observe that the proposed recover technique on the condition values can also improve the model performance significantly on both template questions and NL questions. As shown in Table 2.3, the condition values can also be recovered effectively even if only partial condition value information is provided in the input questions. More analysis about the recover technique will be provided.

Also, the better performance on the template questions indicates that the template questions are easier to be translated to the correct SQL queries compared to the NL questions. The reason is that the template questions are generated based on the pre-defined patterns, which is easier for the model to capture during learning. Therefore, this provides us the guidance to ask the questions following the predefined templates to obtain more accurate SQL queries. In

addition, we can also consider the predicted SQL queries as the model feedback and further refine the input questions accordingly.

Analysis of the Generated SQL Query

In addition to the quantitative evaluations, we have also conducted an extensive set of qualitative case studies to compare the SQL queries produced by various models. Two examples on NL questions are provided in Table 2.6. For both examples, different comparison models have given correct answers for the template, table name, and columns. Note that the Coarse2Fine model cannot handle questions on two tables. For example 1, M-SQLNET provides a wrong procedure short title "*parent infus nutrit sub*" due to the mis-classification error. Seq2Seq and Ptr generate a partially correct procedure short title "*other abdomen*" and "*spinal abdomen artery*", respectively. In addition to their inability to obtain the correct condition values, these baseline methods do not even have the ability to correctly predict the condition column "*procedures.short_title*" in example 1. Similarly, in example 2, the generated SQL query by Seq2Seq model is not executable even if it correctly predicts the value "*ferritin*" for the second condition, since it predicts an incorrect condition column "*lab.itemid*". In this case, the recover technique can only recover the condition value "*51200*" for "*lab.itemid*" instead of keeping the condition value "*ferritin*" that is correctly generated. These predicted results indicate that successfully recovering the condition values still requires the language generation models to produce correct condition columns and sufficiently relevant keywords. Note that it is unable to recover the condition values for M-SQLNET since its predicted values are already in the look-up table. Different from these baseline methods, our proposed TREQS model is able to generate totally correct SQL queries for both examples even without applying the recover technique. This shows the ability of our TREQS method to predict the correct condition values without affecting the performance of other components in the SQL query.

Accumulated Attention Visualization

Visualization of attention weights can help in interpreting the model and explaining experimental results by providing an intuitive view about the relationships between generated tokens and source context, i.e., input questions. In Table 2.7, we show seven natural language examples with reasoning questions and SQL queries that are generated using the proposed TREQS method. The goal here is to investigate if TREQS can successfully detect important keywords in a question when generating conditions in its corresponding SQL query. Therefore, we choose to visualize the accumulated attention weights instead of the weights for each of the generated tokens. For example, for the question "*get me the number of elective hospital admission patients who had coronary artery primary disease*", the model mainly focuses on "*elective*" and "*admission*" when generating condition "*demographic.admission_type = elective*", and on "*coronary artery*" when generating "*demographic.diagnosis = coronary*

Table 2.6: SQL Queries generated by different models on two NL questions in testing set. The incorrectly predicted words are highlighted in red color.

Method	Example 1	Example 2
Question	how many female patients underwent the procedure of abdomen artery incision?	how many patients admitted in emergency were tested for ferritin?
Ground truth	**select** count (distinct demographic."subject_id") **from** demographic inner join procedures on demographic.hadm_id = procedures.hadm_id **where** demographic."gender" = "f" and procedures."short_title" = "abdomen artery incision"	**select** count (distinct demographic."subject_id") **from** demographic inner join lab on demographic.hadm_id = lab.hadm_id **where** demographic."admission_type" = "emergency" and lab."label" = "ferritin"
M-SQLNET	**select** count (distinct demographic."subject_id") **from** demographic inner join procedures on demographic.hadm_id = procedures.hadm_id **where** demographic."gender" = "f" and procedures."short_title" = "parent infus nutrit sub"	**select** count (distinct demographic."subject_id") **from** demographic inner join lab on demographic.hadm_id = lab.hadm_id **where** demographic."admission_type" = "emergency" and lab."label" = "po2"
Seq2Seq	**select** count (distinct demographic."subject_id") **from** demographic inner join procedures on demographic.hadm_id = procedures.hadm_id **where** demographic."gender" = "m" and procedures."long_title" = "other abdomen"	**select** count (distinct demographic."subject_id") **from** demographic inner join lab on demographic.hadm_id = lab.hadm_id **where** demographic."admission_location" = "phys referral/normal deli" and lab."itemid" = "ferritin"
Seq2Seq+recover	**select** count (distinct demographic."subject_id") **from** demographic inner join procedures on demographic.hadm_id = procedures.hadm_id **where** demographic."gender" = "m" and procedures."long_title" = "other bronchoscopy"	**select** count (distinct demographic."subject_id") **from** demographic inner join lab on demographic.hadm_id = lab.hadm_id **where** demographic."admission_location" = "phys referral/normal deli" and lab."itemid" = "51200"
PtrGen	**select** count (distinct demographic."subject_id") **from** demographic inner join procedures on demographic.hadm_id = procedures.hadm_id **where** demographic."gender" = "f" and procedures."long_title" = "spinal abdomen artery"	**select** count (distinct demographic."subject_id") **from** demographic inner join lab on demographic.hadm_id = lab.hadm_id **where** demographic."admission_type" = "emergency" and lab."label" = "troponin i"
PtrGen+recover	**select** count (distinct demographic."subject_id") **from** demographic inner join procedures on demographic.hadm_id = procedures.hadm_id **where** demographic."gender" = "f" and procedures."long_title" = "spinal tap"	**select** count (distinct demographic."subject_id") **from** demographic inner join lab on demographic.hadm_id = lab.hadm_id **where** demographic."admission_type" = "emergency" and lab."label" = "troponin i"
TREQS	**select** count (distinct demographic."subject_id") **from** demographic inner join procedures on demographic.hadm_id = procedures.hadm_id **where** demographic."gender" = "f" and procedures."short_title" = "abdomen artery abdomen"	**select** count (distinct demographic."subject_id") **from** demographic inner join lab on demographic.hadm_id = lab.hadm_id **where** demographic."admission_type" = "emergency" and lab."label" = "ferritin"
TREQS + recover	**select** count (distinct demographic."subject_id") **from** demographic inner join procedures on demographic.hadm_id = procedures.hadm_id **where** demographic."gender" = "f" and procedures."short_title" = "abdomen artery incision"	**select** count (distinct demographic."subject_id") **from** demographic inner join lab on demographic.hadm_id = lab.hadm_id **where** demographic."admission_type" = "emergency" and lab."label" = "ferritin"

Table 2.7: Visualization of the accumulated attention on conditions that are used in the proposed TREQS approach on NL questions. Different conditions are labeled with different colors. An intense shade on a word indicates a higher attention weight.

Question	Generated SQL
provide the number of patients less than 83 years of age who were diagnosed with pneumococcal pneumonia .	select count (distinct demographic.subject_id) from demographic inner join diagnoses on demographic.hadm_id = diagnoses.hadm_id where demographic.age < 83 and diagnoses.short_title = pneumococcal pneumonia
provide the number of patients whose diagnoses is lap surg convert to open and they are female .	select count (distinct demographic.subject_id) from demographic inner join diagnoses on demographic.hadm_id = diagnoses.hadm_id where demographic.gender = f and diagnoses.long_title = lap surg convert to open they
among patients treated with amitriptyline , calculate the number of female patients .	select count (distinct demographic.subject_id) from demographic inner join prescriptions on demographic.hadm_id = prescriptions.hadm_id where demographic.gender = f and prescriptions.drug = amitriptyline
get me the number of elective hospital admission patients who had coronary artery primary disease .	select count (distinct demographic.subject_id) from demographic where demographic.admission_type = elective and demographic.diagnosis = coronary artery disease
give the number of patients whose admission type is elective and primary disease is abdominal abscess .	select count (distinct demographic.subject_id) from demographic where demographic.admission_type = elective and demographic.diagnosis = abdominal abscess
how many patients aged below 36 years have stayed in the hospital for more than 14 days ?	select count (distinct demographic.subject_id) from demographic where demographic.age < 36 and demographic.days_stay > 14
what is the number of patients whose admission location is emergency room admit and with primary disease t5 fracture ?	select count (distinct demographic.subject_id) from demographic where demographic.admission_location = emergency room admit and demographic.diagnosis = t5 fracture

artery disease". In this example, the condition values are mainly obtained by directly copying from the input question since they are explicitly included. On the other hand, the condition value "*f*" in the SQL query for question "*among patients treated with amitriptyline, calculate the number of female patients*" is mainly obtained through the controlled generation and copying technique since "*f*" is not explicitly provided in the input question. Similarly, TREQS model is able to capture relevant keywords for each condition in other examples.

2.6 Summary

Large amounts of EMR data are collected and stored in relational databases at many clinical centers. Effective usage of the EMR data, such as retrieving patient information, can assist doctors in making future clinical decisions. Recently, the Question-to-SQL generation methods have received a great deal of attention due to their ability to predict SQL query for a given question about a database. Such an automated query generation from a natural language question is a challenging problem in the healthcare domain. In this work, based on the publicly available MIMIC III dataset, a Question-SQL pair dataset (MIMICSQL) is first created specifically for the Question-to-SQL generation task in healthcare. We further proposed a TRanslate-Edit Model for Question-to-SQL (TREQS) generation task on MIMICSQL by first generating the targeted SQL directly and then editing with both attentive-copying mechanism and a recover technique. The proposed model can handle the unique challenges in healthcare and is robust to randomly asked questions. Both the qualitative and quantitative results demonstrate the effectiveness of our proposed method.

Chapter 3

Attention-based Aspect Reasoning for Knowledge Base Question Answering on Clinical Notes

This chapter presents a novel way of retrieving medical information from unstructured clinical notes. First, we discuss the motivation and background of this work in Section 3.1. Section 3.2 provides the related work about question answering. In Section 3.3, we describe the detailed steps to create the dataset for clinical knowledge base question answering. Section 3.4 provides the details of the proposed attention-based aspect reasoning model, and Section 3.5 shows the effectiveness of the model and analyzes the impact of different aspects on predicting correct answers. We conclude the chapter in Section 3.6.

3.1 Introduction

Electronic Health Records (EHR) provide comprehensive information that can assist doctors with their clinical decision making. Traditionally, doctors retrieve the information of patients via accessing structured databases with rule-based systems and reading their clinical notes. Recently, several attempts have been made to build Question-Answering (QA) systems on EHR [76, 101, 125], so doctors can get answers for their questions more efficiently. Generally speaking, QA systems can be grouped into several categories according to the format of data sources. For example, machine reading comprehension (MRC) performs QA on plain text data [84]. Text-to-SQL problem performs QA on database [140, 151]. Knowledge Base QA (KBQA) [14] aims at finding answers from the underlying Knowledge Base (KB), such as Freebase [11]. Wang *et al.* introduced a MIMICSQL dataset for Text-to-SQL generation on MIMIC III database [125]. However, their system is limited to retrieving answers from a database, which does not have information that cannot be quantified, such as family history

Table 3.1: Comparisons of the answerable questions over different types of EHR, including Clinical Notes (CN), Structured Tables (ST) and Knowledge Base (KB). The symbol "✓" indicates that the questions are answerable.

Questions	**CN**	**ST**	**KB**
Q1: What medications has patient P939003 ever been prescribed?	✓	✓	✓
Q2: For patient P164, what are the comorbidities associated with Asthma?	✓	✓	✓
Q3: What does patient P961115 take ibuprofen for?	✓		✓
Q4: Give me all diseases that are revealed by non contrast head CT scan on patient P0126.	✓		✓
Q5: Which patients have been diagnosed with both Gout and GERD?		✓	✓
Q6: Give me all patients who have been prescribed with propofol.		✓	✓
Q7: Which medications can be prescribed for reducing creatinine levels?			✓
Q8: What are the obese indicators of heart disease in all medical records of patient P258?			✓

and discharge conditions. Pampari *et al.* proposed an emrQA dataset for MRC on clinical notes [76]. However, their model can only access information from a single block of texts, which is not practical for doctors who may need information from a collection of clinical notes.

In this work, we present ClinicalKBQA, a dataset for QA on clinical KB (ClinicalKB) constructed from clinical notes, which alleviates the problems encountered with emrQA by allowing doctors to access information across different notes. ClinicalKBQA is composed of two subsets, namely, Clinical Knowledge Base (ClinicalKB) and Question-Answering (QA) pairs, both of which are constructed by leveraging existing annotations of clinical notes that are available for various NLP tasks in n2c2[1] (previously known as i2b2).

ClinicalKB integrates advantages of both structured database and unstructured clinical notes. On the one hand, the intrinsic graph structure of ClinicalKB connects the information of different patients and clinical notes via relations/edges, which allows it to answer questions associated with many patients and clinical notes (e.g., **Q5-Q8** in Table 3.1). On the other hand, ClinicalKB includes comprehensive patient information as in clinical notes, which makes it possible to answer questions not covered in database (e.g., **Q3, Q4, Q7, and Q8** in Table 3.1).

To tackle the KBQA challenges in ClinicalKBQA dataset, we proposed an attention-based aspect reasoning (AAR) approach. Specifically, for each input question, we represent each candidate answer as four aspects, including entity, type, path, and context, and analyze the matching scores between the input question and candidate answers based on their embeddings. Through the results analysis, we found that the impact of different candidate aspects on retrieving final answers tends to be different. Two aspects, entity and context, provides the node specific information, which helps to retrieve nodes that satisfy the constraints specified in the questions. While the general information included in the other two aspects, type and path, are helpful for the model to filter out more nodes that satisfy the constraints of the node type and path.

[1] `https://portal.dbmi.hms.harvard.edu/projects/n2c2-nlp/`

In summary, the major contributions of this work are as follows.

- Created a dataset for knowledge base question answering task in healthcare domain, namely ClinicalKBQA, which consists of two sets: (i) ClinicalKB: which is a comprehensive clinical knowledge base created based on the expert annotations in n2c2 dataset, and (ii) QA pairs: a large-scale question answering dataset on ClinicalKB.
- Proposed an attention-based aspect-level reasoning (AAR) method for KBQA.
- Conducted experimental analysis on ClinicalKBQA dataset to demonstrate the effectiveness of AAR model and analyzed the significance of different aspects in providing accurate answers. We aim at improving current KBQA models in healthcare QA systems via addressing challenges presented by ClinicalKBQA and provide more efficient assistance for doctors to retrieve, understand, and utilize the clinical information in clinical notes.

3.2 Related Works

Question-Answering (QA) aims at automatically answering natural language questions about data sources in a variety of formats, including free text [84], knowledge base [22], and database [151]. Knowledge base question answering (KBQA) has gained a lot of attention in recent years with the rapid growth of large-scale knowledge bases, such as YAGO2 [50] and Freebase [11]. Advances in deep neural networks also allowed KBQA models to be trained in an end-to-end manner [12, 45, 21] and achieve competitive performance compared to traditional semantic parsing based methods [2, 61].

QA in the healthcare domain is still an underexplored research topic, especially due to the lack of large-scale annotated datasets and patient privacy issues [54]. Traditional biomedical QA depends on rule-based or heuristic feature-based methods [5]. Recently, several datasets have been created for machine reading comprehension (MRC), including BioASQ for semantic indexing and QA [107], CliCR for MRC on clinical case reports [101], PubMedQA for MRC on biomedical research texts [53] and emrQA for MRC on clinical notes [76]. MIMICSQL [125] was presented for QA on structured EMR data by translating questions to SQL queries. These datasets allow researchers to handle unique challenges present in the healthcare domain. Table 3.2 shows a comparison of our ClinicalKBQA to these datasets for QA in healthcare. There are only a few works on KB in healthcare. SNOMED [32] is a KB with standard clinical terminologies for healthcare documentation. Rotmensch *et al.* [88] learnt a knowledge graph of symptom and disease from EMR by considering the importance measure between terms.

Table 3.2: Comparison of ClinicalKBQA with other datasets for QA in healthcare domain.

Dataset	**Data Source**	**QA Task**	**Answer Type**
BioASQ	Biomedical Articles	MRC	Text Span
CliCR	Clinical Reports	MRC	Text Span
PubMedQA	Biomedical research	MRC	Text Span
emrQA	Clinical Notes	MRC	Text Span
MIMICSQL	Structured Clinical Tables	Text-to-SQL	SQL Query
ClinicalKBQA	Clinical Notes	KBQA	KB Entity

3.3 The ClinicalKBQA Dataset

ClinicalKBQA consists of two subsets, i.e., ClinicalKB and QA pairs. In this section, we will explain how we created the clinical knowledge base and the question answering dataset.

3.3.1 ClinicalKB

The n2c2 challenge data provide fine-grained document-level expert annotations of clinical records for various NLP tasks in clinical domain. We leverage the annotations about seven tasks to build the clinical knowledge base.

- **Smoking status classification** [110]: Each clinical record is annotated with the smoking status from five possible categories (including current smoker, past smoker, non-smoker, smoker and unknown) along with the smoking-related facts mentioned in the records.
- **Identification of obesity and its co-morbidities** [108]: Each clinical record is annotated with obesity and co-morbidities using both textual judgments (explicitly) and intuitive judgments (implicitly).
- **Medication extraction** [111]: The medication-related information including medication name, dosage along with the mode, frequency, duration and reason of the administration, is annotated in each clinical record.
- **Analysis of relations of medical problems, tests and treatments** [112, 113]: The annotations for concept, assertion, and relation information are provided in each clinical record.
- **Co-reference resolution** [109]: Each clinical record is annotated with concept mentions that are referring to the same entity.

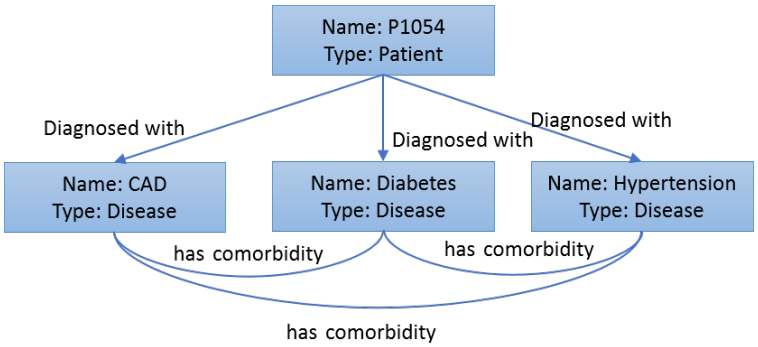


Figure 3.1: A subgraph example about diagnosed diseases and their comorbidity relationships in *Obesity* dataset.

- **Temporal information extraction and reasoning** [100]: The clinically significant events and temporal expressions are annotated along with the temporal relation between them in each clinical record.
- **Risk factors identification of heart disease** [97]: Each clinical record provides the annotation of medically relevant information about heart disease risk factors including the status of smoking, obesity, medication and hypertension.

The narrative blocks in clinical notes, such as family history, provide more detailed clinical information from different aspects and can be efficiently extracted with rule-based methods as additional annotations.

Grounded on domain expert annotated clinical notes in the n2c2 challenge data, we construct clinical KB by following two steps: (1) *Identify entities.* An entity is represented by its name and type. For example, {*name*: "ibuprofen", *type*: "medication"}. (2) *Build triples*, i.e., (subject, predicate, object). Here, both subject and object are entities, and predicate is a relation between them. For example, we can construct a triple ("P961115", "prescribed with", "ibuprofen") based on "a patient with ID P961115 has medication ibuprofen". In addition, we have also fixed some problems in the annotations during pre-processing, such as pronouns like "this/a/his/her" and irrelevant punctuation.

Subgraph Examples in ClinicalKB

We provide a subgraph example in Obesity dataset about diagnosed diseases for patient P1054 and their comorbidity relationships in Figure 3.1. Based on the clinical note of patient P1054, he/she has been diagnosed with three diseases, including CAD, Diabetes and Hypertension. Since the annotations in Obesity dataset focus on the comorbidities relations of different diseases, we include such comorbidity relation between these three diseases.

Figure 3.2 shows the relationships between patient P961115 and the prescribed medications

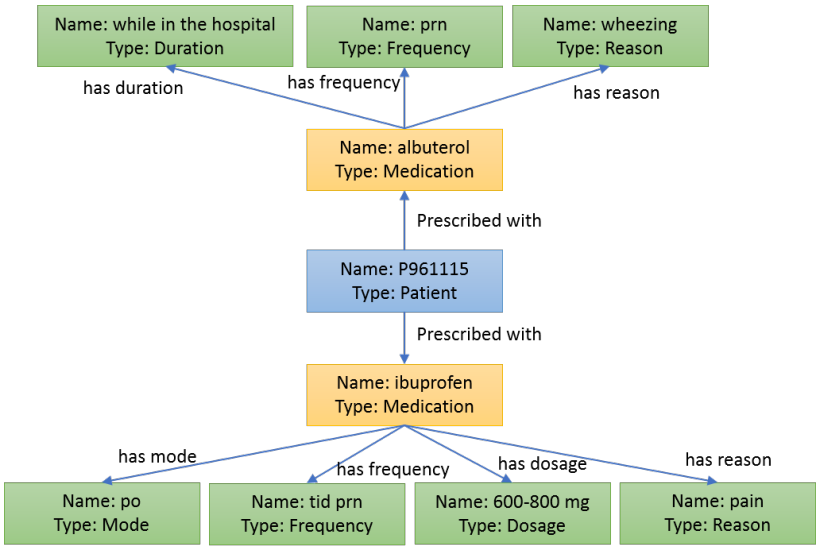


Figure 3.2: A subgraph example about prescribed medications along with their related information in *medication* dataset.

along with other detailed attribute information including dosage, frequency, duration, and reason. We observe that not all attribute information is available for each medication. For example, the duration is only mentioned for albuterol, while the mode and dosage are mentioned only for ibuprofen. We hope that these two subgraph examples can provide an overview for understanding about patient information covered in ClinicalKB. Detailed statistics about ClinicalKB are summarized in Table 3.3.

3.3.2 Question-Answer (QA) Pairs

Question Collection: We first collect a set of questions by polling real interests of physicians and considering existing clinical question resources, including emrQA and MIMICSQL, and further identify questions that can be answered by ClinicalKB. Compared with QA on structured tables [125] and clinical notes [76], we found that questions on ClinicalKB cover a wide range of topics (see Table 3.1). Some questions are not answerable by structured tables or a single clinical note. Take **Q8** as an example, "indicators of diseases" is usually not included in structured tables, and "all medical records" indicates answers cannot be found in a single note.

We then manually identified specific entities in the selected questions and replace them with generic placeholders to normalize and form question templates. In total, we generated a set of 322 question templates, including various paraphrases of questions with the same meanings. For example, the template for **Q3** in Table 3.1 is "*What does patient* |*Patient*| *take* |*Medication*| *for*?", where the generic placeholders |*Patient*| and |*Medication*| are the topic

Table 3.3: Statistics of ClinicalKB and QA pairs created based on the n2c2 dataset. Here, QuesLen, GoldAns and CandAns represent question length, gold-standard answers and candidate answers, respectively.

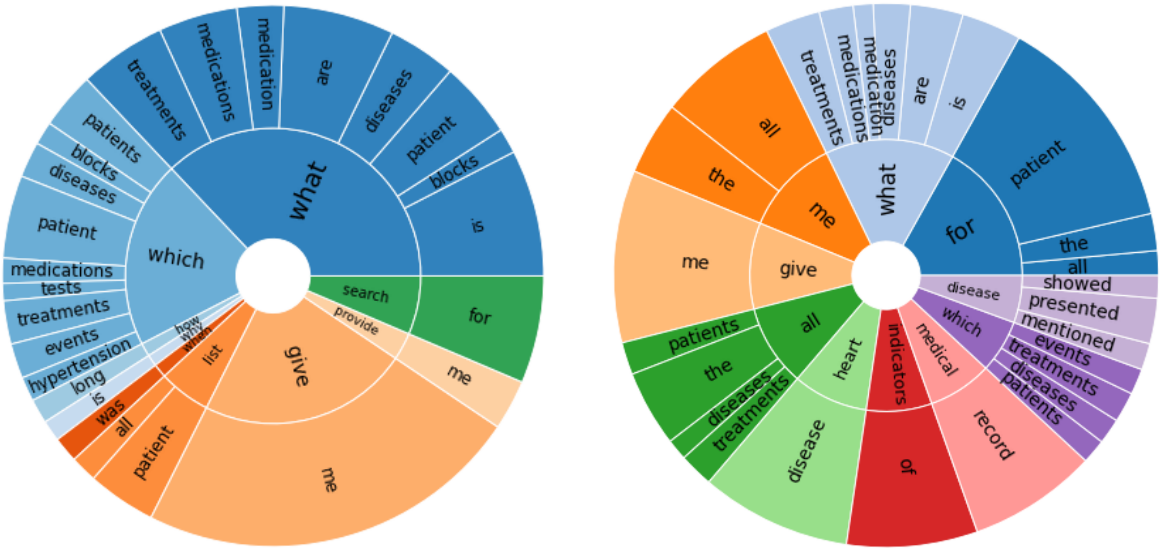
Metric	**Smoking**	**Obesity**	**Medications**	**Relations**	**Co-reference**	**Temporal**	**Risk**
# Patients	502	1,103	261	426	424	310	119
# Entities	6,160	17,861	28,821	20,031	1,581	127,772	6,984
# Entity types	49	42	46	7	7	20	15
# Triples	9,730	42,474	53,519	30,401	1,378	276,513	24,553
# Relations	5	8	14	11	7	13	11
# Question Templates	26	37	59	74	18	29	79
# QA pairs	600	1,126	1,847	2,389	444	626	1,920
Min/Max/Avg QuesLen	4/10/8	5/14/9	5/17/10	6/21/11	8/17/12	8/19/11	8/21/17
Min/Max/Avg # GoldAns	1/82/5	1/816/27	1/111/10	1/29/3	1/2/2	1/239/19	1/69/5
Min/Max/Avg # CandAns	5/2,665/999	3/8,686/2,261	2/6,240/68	4/679/79	3/6/4	5/1,543/175	2/74/17

entities of the question that need to be replaced by the corresponding ClinicalKB entities during question generation. We hope that the questions we collected from domain experts and the existing clinical question sources recognized by the community can provide a helpful resource of QA for researchers in both the medical domain and NLP community.

QA Pairs Generation: This step focuses on populating question templates and identifying corresponding answers. Since patient private information is de-identified in n2c2, we use patient IDs instead of names in patient-specific questions. Each question template may have multiple ways to populate. For example, the template of **Q3** mentioned previously can be populated with different combinations of $|Patient|$ and $|Medication|$. However, we do not need to enumerate all possible questions for it. In practice, we applied two constraints to limit repetitions: (1) Set a threshold to the total number of questions generated for each template. (2) Remove questions without answers. The corresponding answers to each question is simultaneously extracted from clinical notes when generating questions.

3.3.3 Data Analysis

Basic Statistics: The statistics of ClinicalKB and QA pairs are presented in Table 3.3. From this table, we can observe that our ClinicalKB covers seven important medical topics in n2c2. The total number of QA pairs is 8,952. We created more question templates and QA pairs for *Medications*, *Relations* and *Risk* because their annotations are more comprehensive. The average question length is 12 in terms of tokens. Each question has at least one gold-standard answer and a lot of questions have multiple answers. In this work, we refer to the collection of ClinicalKB and QA pairs as the ClinicalKBQA dataset. The number of entities in golden and candidate answers are 9 and 402 in average, respectively. The number of golden and candidate answers for questions in *Co-reference* is relatively small since the variety of annotated terms with the same meaning are small. More details about ClinicalKBQA are provided in Table 3.3.



(a) (b)

Figure 3.3: Distributions of questions by (a) the first two words in all questions and (b) the most common bigrams used in all questions.

Question Types: Our primary goal of knowledge base question answering on clinical notes is to extract patient information from unstructured clinical text. Therefore, all questions included in our ClinicalKBQA dataset are factoid questions which aim to seek reliable and concise medical history information about patients. We group the questions in ClinicalKBQA dataset into different types based on the starting words. The distributions of question types showed in Figure 3.3(a) are generated based on the most common first two starting words in all questions. Table 3.4 provides the quantitative percentage of various question types along with specific examples. It can be observed that the questions starting with "What", "List/Search/Give/Provide" and "Which" account for a large proportion of the dataset and aim to ask for detailed medical facts, such as prescribed medications and the smoking status. The questions starting with "Why" and "How" tend to be open-ended in many open-domain question answering datasets. However, in the ClinicalKBQA dataset, the "Why" and "How" types of questions are mainly included for retrieving attribute facts about medication, including prescribed reason, dosage, frequency, and duration. In addition, the question type "When" are included for extracting the admission and discharge time of patients.

In Figure 3.3(b), we also show a distribution of the most common bigrams used in all questions in ClinicalKBQA dataset. It provides an overview about the specific patient information that various questions aim to extract from clinical notes.

Question Coverage: Table 3.1 provides a comparison of questions that can be answered on different types of EHR data including clinical notes (CT), structured tables (ST) and

Table 3.4: Question types in ClinicalKBQA along with examples.

Question type	**Examples**	**Percentage**
What	What medications has patient P939003 ever been prescribed? What is the smoking status of patient P164? What is the dosage of colacefor patient P11995?	38.29%
List/Search/Give/Provide	List all comorbilities of Asthma for patient P1225. Search for all the coreferenced tests of blood cultureon P727. Give me all patients whose smoking status is current smoker. Provide me the discharge time of patient P76.	35.21%
Which	Which tests are conducted on patient P0161? Which tests are conducted on patient P0161? Which medications can be prescribed for preventing creatinine?	20.81%
Why	Why is patient P74976 prescribed glucotrol? Why is patient P280639 on coumadin? Why was ibuprofen originally prescribed for patient P961115?	2.21%
How much/often/long	How much aspirin does patient P920102 take per day? How often does patient P439766 take regular insulin? How long has patient P652612 been taking levofloxacin?	2.13%
When	When was patient P130 admitted? When was patient P32 discharged?	1.34%

knowledge base (KB). We can observe that knowledge base of patient clinical information is able to answer the basic questions that are answerable by QA on both clinical notes and structured tables. In addition, it has the ability to combine the advantages of free-text clinical notes and structured tables to handle more complex questions. For example, questions **Q7** in Table 3.1 cannot be answered based on structured data since the reason for taking medications are not clearly provided. For questions **Q8**, even if there are lab test information included in the structured data, the diseases that are actually revealed by each test are not specified. While for questions **Q5** and **Q6**, the machine reading comprehension on emrQA cannot provide answers since these questions are related to multiple clinical notes. ClinicalKB is able to integrates the information in different clinical notes or about different patients into a general network structure, which makes it feasible to handle more complexed questions about patients.

3.4 Modeling Knowledge Base Question Answering

3.4.1 Candidate Generation

Based on the statistics of ClincialKB provided in Table 3.3, we can see that the total number of entities and triples in ClincialKB is large and there are various entity types included. It will be computationally expensive for KBQA models to directly search answers from ClinicalKB. Candidate generation is commonly used in the open-domain problems to reduce

the candidate space for the input questions. Therefore, we first generate a candidate subgraph for each question in two steps: (1) We identify one of the entities in the question template as the topic entity (root), and collect all entities connected to it within 3-hop as a candidate subgraph. Each entity in the subgraph except the root is viewed as a candidate answer. For the ClinicalKBQA dataset, the answers to all questions are reachable within 3-hop of their topic entities. (2) We treat the remaining entities in the question as constraints to the candidate sub-graph, and further prune the graph to ensure that paths to the topic entity satisfy the constraints and include entities with expected answer type. For example, the answer type for **Q3** is "disease". Topic entities are "P961115" and "ibuprofen". If we treat "P961115" as root, then, one possible path is ["P961115", "*prescribed with*", "ibuprofen", "*has reason*", "right leg pain", "*has comorbidity*", "pain control"] since it has "ibuprofen". This path is further pruned to ["P961115", "*prescribed with*", "ibuprofen", "*has reason*", "right leg pain"] because answer type is "disease". We show the statistics of candidate answers in Table 3.3.

3.4.2 Attention-based Aspect Reasoning

Generally speaking, there are two groups of methods for KBQA task, i.e., semantic parsing-based [135] and information retrieval-based (IR-based) [133] methods. As an important category of information retrieval-based KBQA methods, embedding-based approaches [12, 45] usually map questions and answer candidates onto a common embedding space and directly calculate their matching scores. Then, ranking techniques are adopted to search answers from KB for given questions. We proposed an embedding-based End-to-End model on ClinicalKBQA dataset that incorporates an attention mechanism between question representations and aspect-level answer candidate representations to calculate matching scores. We will introduce this Attention-based Aspect Reasoning (AAR) approach for knowledge base question answering with more details as follows. Specifically, there are mainly four components in the AAR model: (1) Question Representation: the vector representations of the input question are obtained by a single-layer bidirectional Long short-term memory (LSTM). (2) Graph Representation: for each candidate answer, we considered four aspects based on the answer subgraph, including entity, entity type, path to topic entities, and context entities. Hereafter, we refer them as entity, type, path, and context, respectively. (3) Attention Mechanisms: this reasoning process allows us to capture the underlying dependencies between input question and different aspects of the candidate answers. (4) Scoring: first, distances (i.e., similarity scores) between the input question and the candidate answers in the hidden space are calculated; and then, predicted answers are selected via re-ranking based on the scores. Figure 3.4 shows the framework of our AAR model. In the following section, we will introduce more details of each component.

Question Representations: The question encoder is composed of a word-embedding layer followed by a bi-directional LSTM layer, which encodes a question $q = (q_1, q_2, \ldots, q_{|q|})$ into a sequence of hidden states $H^q = (h_1^q, h_2^q, \ldots, h_{|q|}^q)$, where q_i and h_i^q represent the ith token and

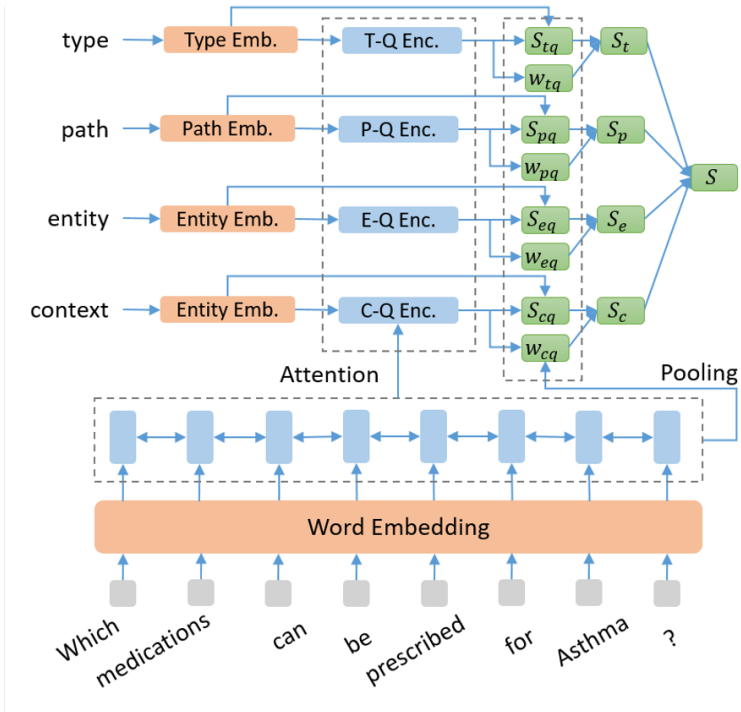


Figure 3.4: The overall framework of AAR model.

its corresponding hidden state, respectively. $|q|$ is the length of the input question.

Graph Representation: To encode candidate subgraphs of the knowledge base, we first convert each subgraph to a candidate answer set, where each element (i.e., node) in the set represents an entity in the subgraph, which has four aspects of information:

- Entity (h_k^e) represents the embedding of the node k in a knowledge base.
- Type (h_k^t) denotes the entity type of the node k. It provides important clue for finding an answer. For example, given a question such as "Give me all patients who have been diagnosed with heart failure.", the answer type should be "Patients".
- Path (h_k^p) represents a path from the topic entity node of a subgraph to the current candidate node. Thus, the path provides relationships between the topic entity and the candidate answer. In the above example, the topic entity is "heart failure".
- Context consists of all neighboring nodes of the current candidate node k. We encoded the context of each candidate answer $c_k = (c_{k_1}, c_{k_2}, \ldots, c_{k_c})$ as a list of hidden states $H_k^c = (h_{k_1}^e, h_{k_2}^e, \ldots, h_{k_c}^e)$, where c_{k_i} is the ith context node in the context of node k, and $h_{k_i}^e$ represents its corresponding entity embedding. For simplicity, in the following part of the section, we will use h^e, h^t, h^p and H^c to represent h_k^e, h_k^t, h_k^p and H_k^c, respectively.

Among these four aspects, the entity and context provide node-specific information, which help us identify more specific nodes in the graph. On the other hand, the type and path represent general information and help us filter out nodes in the graph that satisfy the constraints of the node type and path.

Attention Mechanisms: Now, we discuss the reasoning process over a candidate graph through attention mechanisms, which can discover underlying correlations between a question and different features/aspects of any candidate node. Here, we will use the attention between aspect "type" and the input question, namely type-to-question attention, to illustrate how the attention mechanism works. Given the question representation $H^q = (h_1^q, h_2^q, \ldots, h_{|q|}^q)$, and the type embedding h^t, the alignment score u^{t2q} and attention weight α^{t2q} are calculated as follows.

$$u_i^{t2q} = (h^t)^T tanh(W_{t2q} h_i^q + b_{t2q}) \tag{3.1}$$

$$\alpha_i^{t2q} = \frac{exp(u_i^{t2q})}{\sum_{j=1}^{|q|} exp(u_j^{t2q})} \tag{3.2}$$

where W_{t2q} and b_{t2q} are model parameters. Finally, the type-related question representation, namely type-to-question representation, is obtained by

$$r^{t2q} = \sum_{j=1}^{|q|} \alpha_j^{t2q} h_j^q \tag{3.3}$$

where, r^{t2q} can be considered as a question representation which incorporates type information. Similarly, we can also obtain such representations for other aspects, including path, entity, and context. Hereafter, they are denoted as r^{e2q}, r^{p2q} and r^{c2q}, respectively.

Scoring Answers: The prediction of answers is made based on the similarity score between the input question and each answer candidate, which is a weighted average score of distances between questions and different answer aspects of each candidate. For each aspect, we first calculate the similarity of its embedding and aspect-to-question representation as follows:

$$s_{tq} = (h^t)^T r^{t2q} \tag{3.4}$$

Since different aspects of candidate answers are not equally important to the final predictions, we also calculate the weight of each aspect as follows.

$$w_{tq} = (H_{avg}^q)^T r^{t2q} \tag{3.5}$$

where H_{avg}^q represents the question representation obtained by performing average-pooling over the sequence of hidden states of the question $H^q = (h_1^q, h_2^q, \ldots, h_{|q|}^q)$. Therefore, the final score of each candidate answer a can be obtained as follows:

$$S(q, a) = w_{eq} s_{eq} + w_{tq} s_{tq} + w_{pq} s_{pq} + w_{cq} s_{cq} \tag{3.6}$$

In the testing phase, candidate answers are ranked based on their scores.

Training: In the ClincalKBQA task, we treat the answer retrieval problem as a ranking problem and adopt a pair-wised strategy to train our model. Intuitively, ground truth answers should have higher scores than the other candidate answers. Therefore, during training, for each ground-truth answer node a (positive example), we randomly select a candidate node (not an answer) a' as a negative example. The training loss is a max-margin hinge loss and defined as follows:

$$L = min\frac{1}{|B^q|}\sum_{(a,a^{'})\in B^q}[\gamma + S(q,a) - S(q,a^{'})]_+ \tag{3.7}$$

where $S(q,a)$ and $S(q,a')$ are the final scores of nodes a and a', respectively. $\gamma \in (0,1)$ is a real number that indicates the margin between the positive and negative examples. $[\cdot]_+$ represents the hinge loss, which is defined by $max(0,\cdot)$. Here, B^q denotes a set of positive-negative example pairs $(a,a^{'})$, and $|B^q|$ is the batch size. Intuitively, the hinge loss function increases the margin between the positive and negative examples and allows us to select multiple answers from a set of candidate answers instead of the best answer only.

Inference: During the testing, for each input question, we first retrieve a set of candidate answers C^q from the corresponding knowledge base, and then calculate the score for each candidate answer $a \in C^q$. The best answer is obtained by

$$a_{best} = arg\ \max_{a\in C^q} S(q,a) \tag{3.8}$$

Usually, there are multiple answers for each question, therefore, the candidate answers whose scores are close to the highest score within a margin can also be considered as answers. This inference process can be formulated as follows:

$$f(q,a) = \begin{cases} 1 & S(q,a) > S(q,a_{best}) - \gamma \\ 0 & \text{otherwise} \end{cases} \tag{3.9}$$

Here $f(q,a) = 1$ indicates that node a is the answer to the question q.

3.5 Experiments and Analysis

3.5.1 Experimental Settings

We implemented both models and variants of AAR. Following prior work, we adopted micro-averaged precision, recall, and F1 score to evaluate different models. In our experiment, we split the data into training/development/testing sets with a proportion of 5952/1000/2000. We implemented the AAR model using Pytorch [79] and the best set of parameters are

Table 3.5: Performance results on ClinicalKBQA. # Ans denotes the number of answers predicted by models on testing set. Number of ground-truth answers is 16,251.

Models	**# Ans**	**Precision**	**Recall**	**Accuracy**	**Micro-F1**	**Macro-F1**
SGEmb (full)	2,726	0.7447	0.1249	0.5105	0.2139	0.6617
entity & *sub-graph*	3,153	0.3866	0.0750	0.2370	0.1256	0.3807
path	26,597	0.4870	0.7966	0.7260	0.6045	0.8583
AAR (full)	3,492	**0.8072**	0.1735	0.6525	0.2856	0.7973
entity & *context*	3,707	0.5964	0.1360	0.4725	0.2216	0.6645
type	120,128	0.1205	**0.8908**	0.3685	0.2123	0.6177
path	26,900	0.4780	0.7913	0.7665	0.5960	**0.9057**
type & *path*	16,297	0.6598	0.6616	**0.7745**	**0.6607**	0.8980

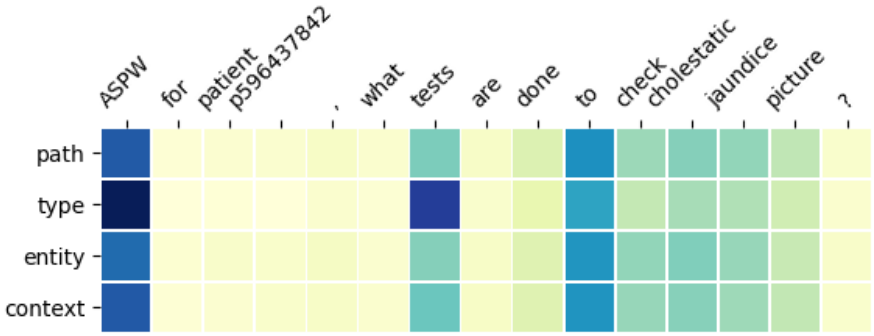
selected based on the development set. We set the size of embeddings for words, entities, entity types, and paths to topic entities to 300. We adopted a single layer Bi-LSTM with the hidden size 150. All parameters were trained using ADAM [59] optimizer with a constant learning rate of 0.0001 for 10 epochs. In addition, we compare the performance of AAR with a subgraph-based approach SGEmd [12], which first calculates embeddings of words, entities, and path to topic entities. Then, each question representation is obtained by applying average pooling to word embeddings. Answer candidates are represented by entities, paths to topic entities, and subgraphs. This method is known as subgraph embedding.

3.5.2 Experimental Results

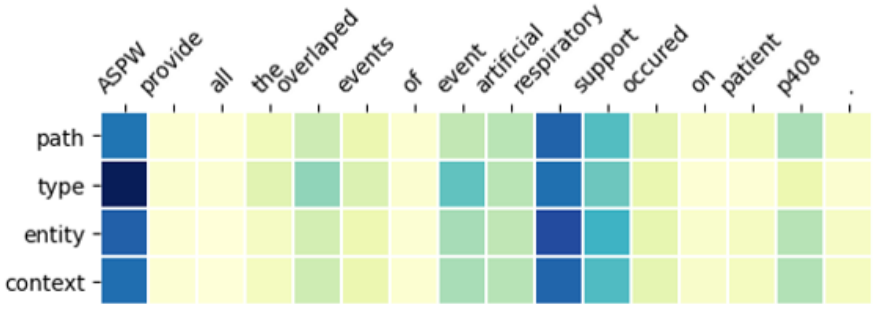
From Table 3.5, we can observe that AAR achieves better results than SGEmb, which is because AAR is equipped with a better encoder, attention mechanisms, and entity *type* information. To explore the impact of each aspect in our ClinicalKBQA, we studied models with only one aspect of information. SGEmb-*entity* & *sub-graph* and AAR-*entity* & *context*, which only leverage entity embeddings, achieve relatively higher precision and lower recall, and the number of predicted answers is much fewer than that of ground-truth. This is because different answer candidates have different entity embeddings and matching scores, which makes the model favor the answer with the highest score.

On the other hand, models that only consider *path* and *type* achieve relatively lower precision and higher recall since different candidates may share common *type* and *path* embeddings and have the same matching score. Thus, the number of predicted answers are more than that of the ground-truth. Models with only *path* information perform significantly better than other variants with only one aspect, which indicates *path* is the most significant factor for our ClinicalKBQA. Finally, AAR-*type* & *path* achieves the best accuracy and Micro-F1 score.

We have also shown the heat-map based on the attention mechanism for two input questions in Figure 3.5. In the first example, the model gives more weight to the aspect "type" among



(a)



(b)

Figure 3.5: Attention heatmaps generated by the cross-attention module. ASPW denotes weights for different aspects, which are *path*, *type*, *entity*, and *context*.

all four aspects, which indicates that the aspect "type" of candidate answers are the most important features for the final prediction. For aspect-towards-question attention, all four aspects capture the keywords "tests" and "to check cholestatic jaundice picture". These important keywords are serving as the query conditions to identify qualified candidate answers whose node "type" is "test" and can be used "to check cholestatic jaundice picture". For the second example, we can find that the aspect "type" is also the most important aspect for the candidate answers to satisfy the query conditions of "event" and "artificial respiratory support". This analysis of attention weights is helpful for us to explain how the AAR model identifies correct answers for an input question. It also provides us insights about the impactful aspects of candidate answers to match the input questions on the ClinicalKBQA dataset.

3.6 Summary

In this work, we introduced a dataset for question answering (QA) on ClinicalKB, namely ClinicalKBQA, which is composed of two subsets, i.e., ClinicalKB and QA pairs. ClinicalKB is built from expert annotated clinical notes; thus, it allows doctors to ask questions on a collection of notes for different patients. We have also introduced a procedure for generating answer candidate subgraphs from ClinicalKB for given questions. In addition, an attention-based aspect-level reasoning model is developed for KBQA on the new created dataset. Finally, we conducted experimental analysis and studied the significance of different aspects in providing accurate answers.

Chapter 4

Self-Supervised Learning of Contextual Embeddings for Link Prediction in Heterogeneous Networks

This chapter presents a novel self-supervised learning framework of contextual embeddings for link prediction task in heterogeneous networks. First, the introduction to this work is presented in Section 4.1. Section 4.2 provides an overview of related work about network embedding learning and differentiates our work from other existing work. In Section 4.3, we introduce the problem formulation and present the proposed SLiCE model. We describe the details of experimental analysis and show the comparison of our model with the state-of-the-art methods in Section 4.4. Finally, Section 4.5 concludes the discussion of the work.

4.1 Introduction

Representation learning for heterogeneous networks has gained a lot of attention in recent years [31, 18, 142, 114, 127, 1], where a low-dimensional vector representation of each node in the graph is used for downstream applications such as link prediction [145, 18, 1] or multi-hop reasoning [43, 23, 150]. Many of the existing methods focus on obtaining a *static* vector representation per node that is agnostic to any specific context and is typically obtained by learning the importance of all the node's immediate and multi-hop neighbors. However, we argue that nodes in a heterogeneous network exhibit a different behavior, based on different relation types and their participation in diverse network communities. Further, most downstream tasks such as link prediction are dependent on the specific contextual information related to the input nodes that can be extracted in the form of *task specific subgraphs*.

Incorporation of contextual learning has led to major breakthroughs in the natural language processing community [82, 27], in which the same word is associated with different concepts depending on the context of the surrounding words. A similar phenomenon can be exploited in graph structured data and it becomes particularly pronounced in heterogeneous networks where the addition of relation types as well as node and relation attributes leads to increased diversity in a node's contexts. Figure 4.1 provides an illustration of this problem for a healthcare network. Given two patients who are diagnosed with different diseases, we posit that the task of predicting the link ($Patient\ 2, diagnosed\ with, Hyperipidemia$) would perform better if the node representation is reflective of the common diseases, i.e., *Malignant Hypertension* and *Hypopotassemia*. This is in contrast to existing methods where patients embeddings would reflect information aggregation from all of their diseases, including the diseases *Diarrhea*, *Nausea with vomiting*, *Pulmonary collapse* and *Acute kidney failure*, which are not part of the common context.

Contextual learning of node representations in network data has recently gained attention with different notions of context emerging (see Table 4.1). In homogeneous networks, communities provide a natural definition of a node's participation in different contexts referred to as facets or aspects [131, 33, 68, 121, 77]. Given a task such as link prediction, inferring the cluster-driven connectivity between the nodes has been the primary basis for these approaches. However, accounting for higher-order effects over diverse meta-paths (defined as paths connected via heterogeneous relations) is demonstrated to be essential in representation learning and link prediction in heterogeneous networks [18, 142, 127, 51]. Therefore, contextual learning methods that primarily rely on the well-defined notion of graph clustering will be limited in their effectiveness for heterogeneous networks where modeling semantic association (via meta-paths or meta-graphs) is at least equal or more important than community structure for link prediction.

In this work, we seek to make a fundamental advancement over these categories that aims to contextualize a node's representation with regards to either a cluster membership or association with meta-paths or meta-graphs. We believe that the definition of a context needs to be expanded to subgraphs (comprising heterogeneous relations) that are task-specific and learn node representations that represent the collective heterogeneous context. With such a design, a node's embedding will change dynamically based on its participation in one input subgraph over another. Our experiments indicate that this approach has strong merit with link prediction performance, thus improving it by 10%-25% over many state-of-the-art approaches.

We propose shifting the node representation learning from a node's perspective to a subgraph point of view. Instead, of focusing on "what is the best representation for a node v", we seek to answer "what are the best collective node representations for a given subgraph g_c" and "how such representations can be potentially useful in a downstream application?" Our proposed framework SLiCE (which is an acronym for Self-supervised LearnIng of Contextual Embeddings), accomplishes this by bridging static representation learning methods using global information from the entire graph with *localized attention driven mechanisms to learn*

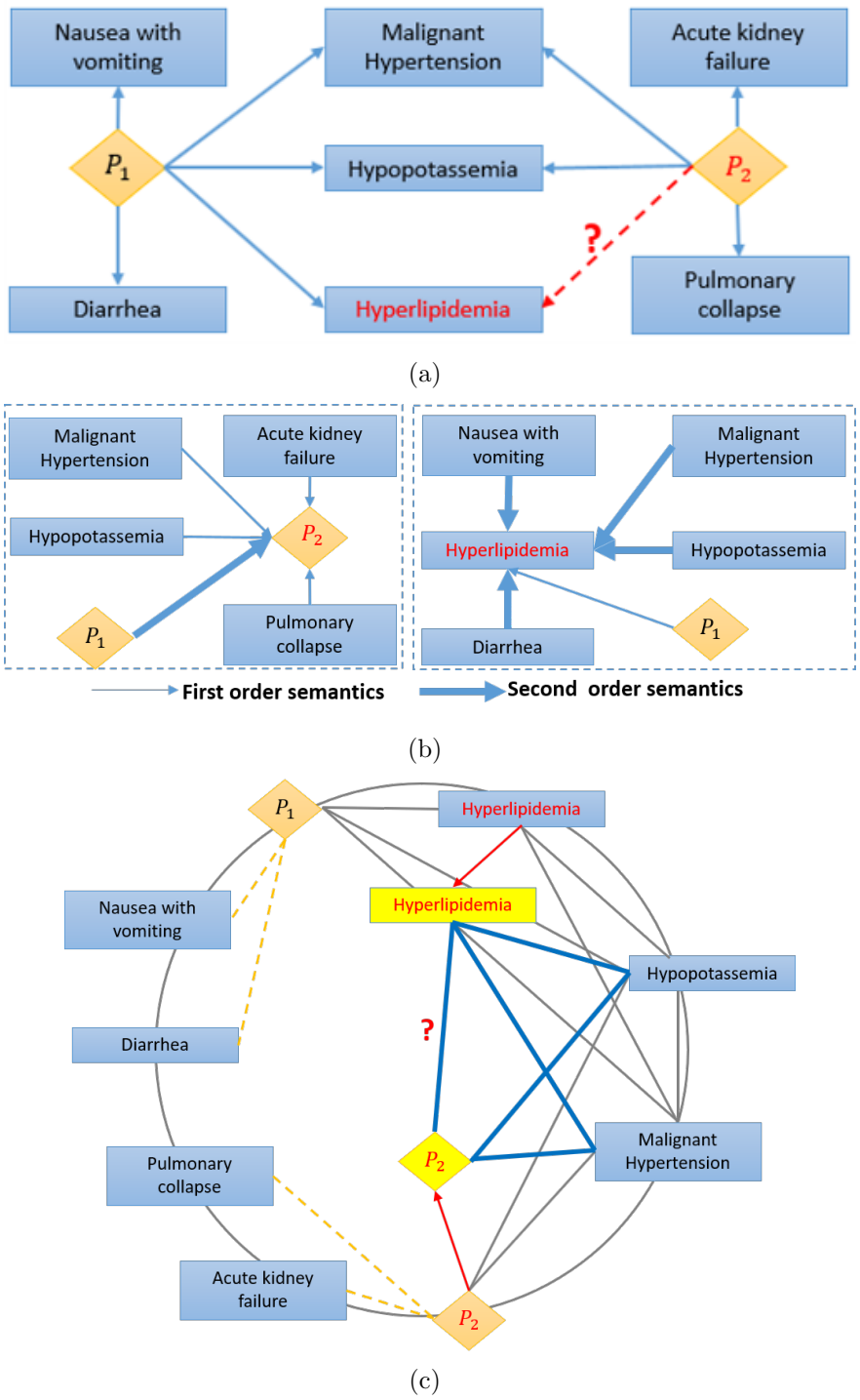


Figure 4.1: Subgraph driven contextual learning in a healthcare knowledge graph. (a) Patient nodes diagnosed with diverse diseases (participate in diverse contexts). (b) State-of-the-art methods aggregate global semantics for patients based on all diagnosed diseases. (c) Our approach uses context subgraph between patients to contextualize their node embeddings during link prediction.

Table 4.1: Comparison of representative approaches for learning heterogeneous network (HN) embeddings proposed in the recent literature from contextual learning perspective. Other abbreviations used: graph convolutional network (GCN), graph neural network (GNN), random walk (RW), skip-gram (SG). N/A stands for "Not Applicable".

Method	**Multi-Embeddings per Node**	**Context Scope**	**HN support**	**Learning Approach**	**Automated Learning of Meta-Paths/Graphs**
HetGNN [144], HetGAT [127]	N	N/A	Y	GNN	N
GTN [142], HGT [51]	N	N/A	Y	Transformer	Y
GAN [1], RGCN [89]	N	N/A	Y	GCN	N
Polysemy [68], MCNE [121]	Y	Per aspect	N	Extends SG/GCN, GNN	N
GATNE [18], CompGCN [114]	Y	Per relation	Y	HN-SG, GCN	N/A
SPLITTER [33], asp2vec [77]	Y	Per aspect	Y	Extends RW-SG	N
SLiCE (proposed)	Y	Per subgraph	Y	Self-supervision	Y

contextual node representations in heterogeneous networks. While bridging global and local information is a common approach for many algorithms, the primary novelty of SLiCE lies in learning an operator for contextual translation by learning higher-order interactions through self-supervised learning.

Contextualized Representations: Building on the concept of *translation-based embedding models* [13], given a node u and its embedding h_u computed using a global representation method, we formulate graph-based learning of contextual embeddings as performing a vector-space translation Δ_u (informally referred to as *shifting process*) such that $h_u + \Delta_u \approx \tilde{h}_u$, where $\tilde{h}_u$ is the contextualized representation of u. The key idea behind SLiCE is to learn the translation Δ_u where $u \in V(g_c)$. Figure 4.1(c) shows an illustration of this concept where the embedding of both Author1 and Author2 are shifted using the common subgraph with (Paper P1, Representation learning , NeurIPS, Paper P3) as context. We achieve this contextualization as follows: We first learn the higher-order semantic association (HSA) between nodes in a context subgraph. We do not assume any prior knowledge about important metapaths, and SLiCE learns important task specific subgraph structures during training (see Section 4.4.3). More specifically, we first develop a *self-supervised learning* approach that pre-trains a model to learn a HSA matrix on a context-by-context basis. We then fine-tune the model in a task-specific manner, where given a context subgraph g_c as input, we

encode the subgraph with global features and then transform that initial representation via a HSA-based non-linear transformation to produce contextual embeddings (see Figure 4.2).

Our Contributions: The main contributions of our work are:

- Propose *contextual embedding learning for graphs from single relation context to arbitrary subgraphs.*
- Introduce a *novel self-supervised learning approach to learn higher-order semantic associations between nodes* by simultaneously capturing the global and local factors that characterize a context subgraph.
- Show that SLiCE significantly outperforms existing static and contextual embedding learning methods using standard evaluation metrics for the task of link prediction.
- Demonstrate the interpretability, effectiveness of contextual translation, and the scalability of SLiCE through an extensive set of experiments and contribution of a new benchmark dataset.

4.2 Related Work

We begin with an overview of the state-of-the-art methods for representation learning in heterogeneous network and then follow with a discussion on the nascent area of contextual representation learning. Table 4.1 provides a summarized view of this discussion.

Node Representation Learning: Earlier representation learning algorithms for networks can be broadly categorized into two groups based on their usage of matrix factorization versus random walks or skip-gram-based methods. Given a graph G, matrix factorization based methods [17] seek to learn a representation Γ that minimizes a loss function of the form $||\Gamma^T\Gamma - P_V||^2$, where P_V is a matrix containing pairwise proximity measures for $V(G)$. Random walk based methods such as DeepWalk [81] and node2vec [40] try to learn representations that approximately minimize a cross-entry loss function of the form $\sum_{v_i,v_j \in V(G)} -log(p_L(v_j|v_i))$, where $p_L(v_j|v_i)$ is the probability of visiting a node v_j on a random walk of length L starting from node v_i. Node2vec based approach has been further extended to incorporate multi-relational properties of networks by constraining random walks to ones conforming to specific metapaths [31, 20]. Recent efforts [83] seek to unify the first two categories by demonstrating the equivalence of [81] and [40]-like methods to matrix factorization approaches.

Attention-based Methods: A newer category represents graph neural networks and their variants [60, 89]. Attention mechanisms that learn a distribution for aggregating information from a node's immediate neighbors is investigated in [116]. Aggregation of attention from *semantic neighbors*, or nodes that are connected via multi-hop metapaths

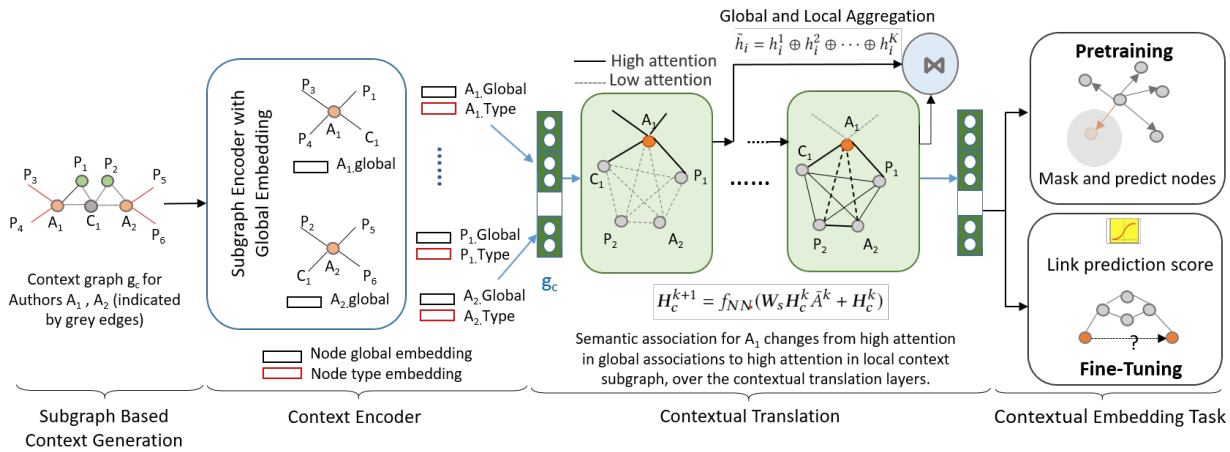


Figure 4.2: Overview of SLiCE architecture. Subgraph context is initialized using global features for each node. Each layer in SLiCE shifts the embedding of all nodes in g_c to emphasize the local dependencies in the contextual subgraph. The final embeddings for nodes in context subgraphs are determined as a function of output from last i layers to combine global with local contextual semantics for each node.

have been exhaustively investigated over the past few years and can be grouped by the underlying neural architectures such as graph convolutional networks [114], graph neural network [144, 127], and graph transformers [142, 51]. Extending the above methods from meta-paths to meta-graphs [48, 149] as a basis for sampling and learning has emerged as a new direction as well.

Contextual Representation Learning: Works such as [131, 33, 68, 121, 98, 7] study the "multi-aspect" effect. Typically, "aspect" is defined as a node's participation in a community or cluster in the graph or even being an outlier, and these methods produce a node embedding by accounting for its membership in different clusters. However, most of these methods are studied in detail for homogeneous networks. More recently, this line of work has evolved towards addressing finer issues such as inferring the context, addressing limitations with offline clustering, producing different vectors depending on the context as well as extension towards heterogeneous networks [73, 77].

Beyond these works, a number of newer approaches and objectives have also emerged. The authors of [1] compute a node's representation by learning the attention distribution over a graph walk context where it occurs. The work presented in [18] is a metapath-constrained random walk based method that *contextualizes* node representations per relation. It combines a base embedding derived from the global structure (similar to above methods) with a relation-specific component learnt from the metapaths. In a similar vein, [114] provides operators to adapt a node's embedding based on the associated relational context.

Key Distinctions of SLiCE: To summarize, the key differences between SLiCE and

existing works are as follows: (i) *Our contextualization objective* is formulated on the basis of a subgraph that distinguishes SLiCE from [18] and [114]. While subgraph-based representation learning objectives have been superficially investigated in the literature [147], they do not focus on generating contextual embeddings. (ii) From a *modeling perspective*, our self-supervised approach is distinct from both the metapath-based learning approaches outlined in the *attention-based methods section* (we learn important metapaths automatically without requiring any user supervision) as well as the clustering-centric multi-aspect approaches discussed in the *contextual representation learning* category.

4.3 The Proposed Framework

4.3.1 Problem Formulation

Before presenting our overall framework, we first briefly provide the formal definitions and notations that are required to comprehend the proposed approach.

Definition: (Heterogeneous Graph). We represent a heterogeneous graph as a 6-tuple $G = (V, E, \Sigma_V, \Sigma_E, \lambda_v, \lambda_e)$ where, V (alternatively referred to as $V(G)$) is the set of nodes and E (or $E(G)$) denotes the set of edges between the nodes. λ_v and λ_e are functions mapping the node (or edge) to its node (or edge) type Σ_V and Σ_E, respectively.

Definition: (Context Subgraph). Given a heterogeneous graph G, the context of a node v or node-pair (u, v) in G can be represented as the subgraph g_c that includes a set of nodes selected with certain criteria (e.g., k-hop neighbors for v or k-hop neighbors connecting (u, v)) along with their related edges. The context of the node or node-pair can be represented as $g_c(v)$ and $g_c(u, v)$.

Problem definition. Given a heterogeneous graph G, a subgraph g_c and the link prediction task T, compute a function $f(G, g_c, T)$ that maps each node v_i in the set of vertices in g_c, denoted as $V(g_c)$, to a real-valued embedding vector h_i in a low-dimensional space d such that $h_i \in \mathbb{R}^d$. We also require that S, a scoring function serving as a proxy for the link prediction task, satisfies the following: $S(v_i, v_j) \geq \delta$ for a positive edge $e = (v_i, v_j)$ in graph G and $S(v_i, v_j) < \delta$ if e is a negative edge, where δ is a threshold.

Overview of SLiCE. In this work, the proposed SLiCE framework mainly consists of the following four components: (1) ***Contextual Subgraph Generation and Encoding***: generating a collection of context subgraphs and transforming them into a vector representation. (2) ***Contextual Translation***: for nodes in a context subgraph, translating the global embeddings, which consider various heterogeneous attributes about nodes, relations and graph structure, to contextual embeddings based on the specific local context. (3) ***Model Pre-training***: learning higher order relations with the self-supervised contextualized node prediction task. (4) ***Model Fine-tuning***: the model is then tuned by the supervised

link prediction task with more fine-grained contexts for node pairs. Figure 4.2 shows the framework of the proposed SLiCE model. In the following sections, we will introduce more details layer-by-layer.

4.3.2 Context Subgraphs: Generation and Representation

Context Subgraph Generation

In this work, we generate the pre-training subgraphs G_C for each node in the graph using a random walk strategy. Masking and predicting nodes in the random walks helps SLiCE learn the global connectivity patterns in G during the pre-training phase. In the fine tuning phase, the context subgraphs for link prediction are generated using following approaches: (1) *Shortest Path* strategy considers the shortest path between two nodes as the context. (2) *Random* strategy, on the other hand, generates contexts following the random walks between two nodes, limited to a pre-defined maximum number of hops. Note that the context generation strategies are generic and can be applied for generating contexts in many downstream tasks such as link prediction [145], knowledge base completion [95] or multi-hop reasoning [23, 43].

In our experiments, context subgraphs are generated for each node v in the graph during pre-training and for each node-pair during fine-tuning. Each generated subgraph $g_c \in G_c$ is encoded as a set of nodes denoted by $g_c = (v_1, v_2, \cdots, v_{|V_c|})$, where $|V_c|$ represents the number of nodes in g_c. Different from the sequential orders enforced on graph sampled using pre-defined metapaths, the order of nodes in this set is not important. Therefore, our context subgraphs are not limited to paths, and can handle tree or star-shaped subgraphs.

Context Subgraph Encoder

We first represent each node v_i in the context subgraph as a low-dimensional vector representation by $h_i = \boldsymbol{W}_e f_{attr}(v_i)$, where $f_{attr}(.)$ is a function that returns a stacked vector containing the structure-based embedding of v_i and the embedding of its attributes. $\boldsymbol{W}_e$ is the learnable embedding matrix. We represent the input node embeddings in g_c as $\boldsymbol{H}_c = (h_1, h_2, \cdots, h_{|V_c|})$. It is flexible to incorporate the node and relation attributes (if available) for attributed networks [18] in the low-dimensional representations or initialize them with the output embeddings learnt from other global feature generation approaches that capture the multi-relational graph structure [40, 31, 127, 142, 114].

There are multiple approaches for generating global node features in heterogeneous networks (see "related work"). Our experiments show that the node embeddings obtained from random walk based skip-gram methods (RW-SG) produces competitive performance for link prediction tasks. Therefore, in the proposed SLiCE model, we mainly consider the pre-trained node representation vectors from node2vec for initialization of the node features.

4.3.3 Contextual Translation

Given a set of nodes V_c in a context subgraph g_c and their global input embeddings $\boldsymbol{H}_c \in \mathbb{R}^{d\times|V_c|}$, the primary goal of contextual learning is to *translate (or shift)* the global embeddings in the vector space towards their new positions that indicate the most representative roles of nodes in the structure of g_c. We consider this mechanism as a transformation layer and the model can include multiple such layers according to the higher-order relations contained in the graph. Before introducing the details of this contextual translation mechanism, we first provide the definition of the *semantic association matrix*, which serves as the primary indicator of the translation of embeddings according to specific contexts.

Definition: (Semantic Association Matrix). A semantic association matrix, denoted as $\bar{A}$, is an *asymmetric weight matrix* that indicates the high-order relational dependencies between nodes in the context subgraph g_c.

Note that the semantic association matrix will be asymmetric since the influences of two nodes on one another in a context subgraph tend to be different. The adjacency matrix of the context subgraph, denoted by A_{gc}, can be considered as a trivial candidate for $\bar{A}$, which includes the local relational information of context subgraph g_c. However, the goal of contextual embedding learning is to translate the global embeddings using the contextual information contained in the specific context g_c while keeping the nodes' connectivity through the global graph. Hence, instead of setting it to A_{gc}, we contextually learn the semantic associations, or more specifically the weights of the matrix $\bar{A}^k$ in each translation layer k by incorporating the connectivity between nodes through both local context subgraph g_c and global graph G.

Implementation of Contextual Translation: In the translation layer $k+1$, the semantic association matrix $\bar{A}^k \in \mathbb{R}^{|V_c|\times|V_c|}$ is updated by the transformation operation defined in Eq. (4.1). It is accomplished by performing message passing across all nodes in context subgraph g_c and updating the node embeddings $\boldsymbol{H}_c^k = (h_1^k, h_2^k, \cdots, h_{|V_c|}^k)$ to be $\boldsymbol{H}_c^{k+1}$.

$$\boldsymbol{H}_c^{k+1} = f_{NN}(\boldsymbol{W}_s \boldsymbol{H}_c^k \bar{A}^k + \boldsymbol{H}_c^k) \tag{4.1}$$

where f_{NN} is a non-linear function and the transformation matrix $\boldsymbol{W}_s \in \mathbb{R}^{d\times d}$ is the learnable parameter. The residual connection [47] is applied to preserve the contextual embeddings in the previous step. This allows us to maintain the global relations by passing the original global embeddings through layers while learning contextual embeddings. Given two nodes v_i and v_j in context subgraph g_c, the corresponding entry $\bar{A}_{ij}^k$ in semantic association matrix can be computed using the multi-head (with N_h heads) attention mechanism [115] in order to capture relational dependencies under different subspaces. For each head, we calculate $\bar{A}_{ij}^k$ as follows:

$$\bar{A}_{ij}^k = \frac{exp\left((\boldsymbol{W}_1 h_i^k)^T (\boldsymbol{W}_2 h_j^k)\right)}{\sum_{t=1}^{|V_c|} exp\left((\boldsymbol{W}_1 h_i^k)^T (\boldsymbol{W}_2 h_t^k)\right)} \tag{4.2}$$

where the transformation matrix $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ are learnable parameters. It should be noted that, different from the aggregation procedure performed across all nodes in the general graph G, the proposed translation operation is only performed within the local context subgraph g_c. The updated embeddings after applying the translation operation according to context g_c indicate the most representative roles of each node in the specific local context neighborhood. In order to capture the higher-order association relations within the context, we apply multiple layers of the transformation operation in Eq. (4.1) by stacking K layers as shown in Figure 4.2, where K is the largest diameter of the subgraphs sampled in context generation process.

By applying multiple translation layers, we are able to obtain multiple embeddings for each node in the context subgraph. In order to collectively consider different embeddings in the downstream tasks, we aggregate the node embeddings learnt from different layers $\{h_i^k\}_{k=1,\ldots,K}$ as the contextual embedding $\tilde{h}_i$ for each node as follows.

$$\tilde{h}_i = h_i^1 \oplus h_i^2 \oplus \cdots \oplus h_i^K \tag{4.3}$$

Given a context subgraph g_c, the obtained contextual embedding vectors $\{\tilde{h}_i\}_{i=1,2,\ldots,|V_c|}$ can be fed into the prediction tasks. In pre-training step, a linear projection function is applied on the contextual embeddings to predict the probability of masked nodes. For fine-tuning step, we apply a single layer feed-forward network with softmax activation function for binary link prediction.

4.3.4 Model Training Objectives

Self-supervised Contextual Node Prediction

Our model pre-training is performed by training the self-supervised contextualized node prediction task. More specifically, for each node in G, we generate the node context g_c with diameter (defined as the largest shortest pair between any pair of nodes) using the aforementioned context generation methods and randomly mask a node for prediction based on the context subgraph. The graph structure is left unperturbed by the masking procedure. Therefore, the pre-training is learnt by maximizing the probability of observing this masked node v_m based on the context g_c in the following form.

$$\theta = \arg\max_{\theta} \prod_{g_c \in G_C} \prod_{v_m \in g_c} p(v_m | g_c, \theta) \tag{4.4}$$

where θ represents the set of model parameters. The procedure for pre-training is given in Algorithm 1. In this algorithm, lines 2-8 generate context subgraphs for nodes in the graph and further applies a random masking strategy to process the data for pre-training. Lines 9-10 learn the pre-trained global node features and initialize them as the node embeddings in SLiCE. In lines 13-15, we apply the contextual translation layers on the context subgraphs,

Algorithm 1: Self-supervised Pre-training in SLiCE.

1 **Require:** Graph G with nodes set V and edges set E, embedding size d, context subgraph size m, No. of translation layers K.
2 Pre-training dataset $G_c^{pre} \leftarrow \emptyset$
3 **for *each*** $v \in V$ **do**
4 $\quad g_c^v =$GetContext(G, v, m) ▷ Generate context subgraph
5 $\quad g_c^v = Encode(g_c^v)$ ▷ Encode context as a sequence
6 $\quad v_m = Random(g_c^v)$ ▷ *Mask a node in* g_c^v *for prediction*
7 $\quad G_c^{pre}.Append(g_c^v, v_m)$
8 **end**
9 $\boldsymbol{H} \leftarrow EmbedFunction(G, d)$ ▷ Learn global embeddings
10 Initialize node embeddings as $\boldsymbol{H}^0 = \boldsymbol{H}$.
11 **while** *not converged* **do**
12 $\quad$ **for** g_c^v **to** G_c^{pre} **do**
13 $\quad\quad$ **for** $k = 1$ **to** K **do**
14 $\quad\quad\quad \boldsymbol{H}_c^{k+1} = f_{NN}(\boldsymbol{W}_s \boldsymbol{H}_c^k \bar{A}^k + \boldsymbol{H}_c^k)$ with $\bar{A}$ calculated by Eq. (4.2)
15 $\quad\quad$ **end**
16 $\quad\quad$ *Contextual embeddings* $\tilde{\boldsymbol{H}}_c = \boldsymbol{H}_c^1 \oplus \boldsymbol{H}_c^2 \oplus \cdots \oplus \boldsymbol{H}_c^K$
17 $\quad\quad$ *Update parameters with the contextualized node prediction task using the objective function in Eq. (4.4).*
18 $\quad$ **end**
19 **end**

aggregate the output of different layers as the contextual node embeddings in line 16 and update the model parameters with contextual node prediction task. In a departure from traditional skip-gram methods [77] that predicts a node from the path prefix that precedes it in a random walk, our random masking strategy forces the model to learn higher-order relationships between nodes that are arbitrarily connected by variable length paths with diverse relational patterns.

Fine-tuning with Supervised Link Prediction

The SLiCE model is further fine-tuned on the *contextualized link prediction task by generating multiple fine-grained contexts for each specific node-pair* that is under consideration for link prediction. Based on the predicted scores, this stage is trained by maximizing the probability of observing a positive edge (e_p) given context (g_{cp}), while also learning to assign low probability to negatively sampled edges (e_n) and their associated contexts (g_{cn}). The overall objective is obtained by summing up the training data subsets with positive edges (D_p) and negative edges (D_n). Algorithm 2 shows the process of fine-tuning step. In this algorithm, lines 2-7 generate context subgraphs of the node-pairs for link prediction task and process

Algorithm 2: Fine-tuning in SLICE with link prediction.

1 **Require:** Graph G with nodes set V and edges set E, list of edges E_{lp} for link prediction, global embeddings $\boldsymbol{H}$, pre-trained model parameters θ, context subgraph size m, No. of translation layers K.
2 Fine-tuning dataset $G_c^{fine} \leftarrow \emptyset$
3 **for *each*** $e \in E_{lp}$ **do**
4 $\quad g_c^e =$GetContext(G, e, m) $\triangleright$ Generate context subgraph
5 $\quad g_c^e = Encode(g_c^e)$ $\triangleright$ Encode context as a sequence
6 $\quad G_c^{fine}.Append(g_c^e)$
7 **end**
8 Initialize node embeddings as $\boldsymbol{H}^0 = \boldsymbol{H}$.
9 Set model parameters to pre-trained parameters θ in Algorithm 1.
10 **while** *not converged* **do**
11 $\quad$ **for** g_c^e **to** G_c^{fine} **do**
12 $\quad\quad$ **for** $k = 1$ **to** K **do**
13 $\quad\quad\quad \boldsymbol{H}_c^{k+1} = f_{NN}(\boldsymbol{W}_s \boldsymbol{H}_c^k \bar{A}^k + \boldsymbol{H}_c^k)$ with $\bar{A}$ calculated by Eq. (4.2)
14 $\quad\quad$ **end**
15 $\quad\quad$ *Contextual embeddings* $\tilde{\boldsymbol{H}}_c = \boldsymbol{H}_c^1 \oplus \boldsymbol{H}_c^2 \oplus \cdots \oplus \boldsymbol{H}_c^K$
16 $\quad\quad$ *Update fine-tuning parameters using Eq. (4.5).*
17 $\quad$ **end**
18 **end**

the data for fine-tuning in the same manner described in pre-training. Lines 8-18 perform the fine-tuning with link prediction task.

$$\mathcal{L} = \sum_{(e_p, g_{cp}) \in D_p} \log(P(e_p | g_{cp}, \theta)) + \sum_{(e_n, g_{cn}) \in D_n} \log(1 - P(e_n | g_{cn}, \theta)) \tag{4.5}$$

We compute the probability of the edge between two nodes $e = (v_i, v_j)$ as the similarity score $S(v_i, v_j) = \sigma(\tilde{h}_i^T \cdot \tilde{h}_j)$ [1], where $\tilde{h}_i$ and $\tilde{h}_j$ are contextual embeddings of v_i and v_j learnt based on a context subgraph, respectively. $\sigma(\cdot)$ represents sigmoid function.

4.3.5 Complexity Analysis

We assume that N_{cpn} denotes the number of context subgraphs generated for each node, N_{mc} represents the maximum number of nodes in any context subgraph, and $|V|$ represents the number of nodes in the input graph G. Then, the total number of context subgraphs considered in pre-training stage can be estimated as $|V| * N_{cpn}$ and the cost of iterating over all these subgraphs through multiple epochs will be $O(|V| * N_{cpn})$. Since the generated context subgraphs need to provide us with a good approximation of the total number of

edges in the entire graph, we approximate the total cost as $O(|V| * N_{cpn}) \approx O(N_E)$, where N_E is the number of edges in the training dataset. It can also be represented as $N_E = \alpha_T|E|$, where $|E|$ is the total number of edges in graph G and α_T represents the ratio of training split. The cost for each contextual translation layer in SLiCE model is $O(N_{mc}^2)$ since the dot product for calculating node similarity is the dominant computation and is quadratic to the number of nodes in the context subgraph. In this case, the total training complexity will be $O(|E|N_{mc}^2)$. The maximum number of nodes N_{mc} in context subgraphs is relatively small and it can be considered as a constant that does not depend on the size of the input graph. Therefore, the training complexity of SLiCE is approximately linear to the number of edges in the input graph.

4.3.6 Implementation Details

The proposed SLiCE model is implemented using PyTorch 1.3 [79]. The dimension of contextual node embeddings is set to 128 in SLiCE. We use a skip-gram based random walk approach to encode context subgraphs with global node features. Both pre-training and fine-tuning steps in SLiCE are trained for 10 epochs with a batch size of 128 using the cross-entropy loss function. The model parameters are trained with ADAM optimizer [59] with a learning rate of 0.0001 and 0.001 for pre-training and fine-tuning steps, respectively. The best model parameters are selected based on the development set. Both the number of contextual translation layers and number of self-attention heads are set to 4. We generate context subgraphs by performing random walks between node pairs by setting the maximum number of nodes in context subgraphs and the number of contexts generated for each node to be {6, 12} and {1, 5, 10}, respectively and report the best performance. In the fine-tuning stage, the subgraph with the largest prediction score is selected as the best context subgraph for each node-pair. The implementation of the SLiCE model is made publicly available at this website[1].

4.4 Experiments

In this section, we address following research questions (RQs) through experimental analysis:

1. **RQ1**: Does subgraph-based contextual learning improve the performance of downstream tasks such as link prediction?
2. **RQ2**: Can we interpret SLiCE's performance using semantic features of context subgraphs?

[1]https://github.com/pnnl/SLICE

Table 4.2: The basic statistics of the datasets used in this work.

Dataset	Amazon	DBLP	Freebase	Twitter	Healthcare
# Nodes	10,099	37,791	14,541	9,990	4,683
# Edges	129,811	170,794	248,611	294,330	205,428
# Relations	2	3	237	4	4
# Training (positive)	126,535	119,554	272,115	282,115	164,816
# Development	14,756	51,242	35,070	32,926	40,612
# Testing	29,492	51,238	40,932	65,838	40,612

3. **RQ3**: Where does contextualization help in graphs? How do we quantify the effect of the embedding shift from static to subgraph-based contextual learning?

4. **RQ4**: What is the impact of different parameters and components (including pre-trained global features and fine-tuning procedure) on the performance of SLiCE?

5. **RQ5**: Can we empirically verify SLiCE's scalability?

4.4.1 Experimental Settings

Datasets used

We use four public benchmark datasets covering multiple applications: *e-commerce* (Amazon), *academic graph* (DBLP), *knowledge graphs* (Freebase), and *social networks* (Twitter). We use the same data split for training, development, and testing as described in previous works [18, 1, 114]. In addition, we also introduce a new knowledge graph from the publicly available real-world Medical Information Mart for Intensive Care (MIMIC) III dataset[2] in the healthcare domain. We generated equal number of positive and negative edges for the link prediction task. Table 4.2 provides the basic statistics of all datasets. The details of each dataset is provided below.

- **Amazon[3]:** includes the co-viewing and co-purchasing links between products. The edge types, also_bought and also_viewed, represent that products are co-bought or co-viewed by the same user, respectively.

- **DBLP[4]:** includes the relationships between papers, authors, venues, and terms. The edge types include paper_has_term, published_at and has_author.

[2]https://mimic.physionet.org/
[3]https://github.com/THUDM/GATNE/tree/master/data
[4]https://github.com/Jhy1993/HAN/tree/master/data

- **Freebase**[5]**:** is a pruned version of FB15K with inverse relations removed. It includes links between people and their nationality, gender, profession, institution, place of birth/death, and other demographic features.
- **Twitter**[3]: includes links between tweets users. The edge types included in the network are re-tweet, reply, mention, and friendship/follower.
- **Healthcare:** includes relations between patients and their diagnosed medical conditions during each hospital admission along with relations to procedures and medications received. To ensure data quality, we use a 5-core setting, *i.e.*, retaining nodes with at least five neighbors in the knowledge graph. The codes for generating this healthcare knowledge graph from MIMIC III dataset are also available at[6].

Comparison Methods

We compare our model against the following state-of-the-art network embedding learning methods. The first four methods learn static embeddings and the remaining methods learn contextual embeddings.

- **TransE** [13] treats the relations between nodes as the translation operations in a low-dimensional embedding space.
- **RefE** [19] incorporates hyperbolic space and attention-based geometric transformations to learn the logical patterns of networks.
- **node2vec** [40] is a random-walk based method that was developed for homogeneous networks.
- **metapath2vec** [31] is an extension of node2vec that constrains random walks to specified metapaths in heterogeneous network.
- **GAN** *(Graph Attention Networks)* [1] is a graph attention network for learning node embeddings based on the attention distribution over the graph walk context.
- **GATNE-T** (*General Attributed Multiplex HeTerogeneous Network Embedding*) [18] is a metapath-constrained random-walk based method that learns relation-specific embeddings.
- **RGCN** *(Relational Graph Convolutional Networks)* [89] learns multi-relational data characteristics by assigning a different weight for each relation.
- **CompGCN** *(Composition-based Graph Convolutional Networks)* [114] jointly learns the embedding of nodes and relations for heterogeneous graph and updates a node representation with multiple composition functions.

[5]https://github.com/malllabiisc/CompGCN/tree/master/data_compressed
[6]https://github.com/pnnl/SLICE

- **HGT** *(Heterogeneous Graph Transformer)* [51] models the heterogeneity of graph by analyzing heterogeneous attention over each edge and learning dedicated embeddings for different types of edges and nodes. We adapt the released implementation of node classification task to perform link prediction task.
- **asp2vec** *(Multi-aspect network embedding)* [77] captures the interactions of the pre-defined multiple aspects with aspect regularization and dynamically assigns a single aspect for each node based on the specific local context.

Experimental Settings

All evaluations were performed using NVIDIA Tesla P100 GPUs. The results of SLiCE are evaluated under the parameter settings described in Section 4.3.6. The results of all baselines are obtained with their original implementations. Note that for all baseline methods, the parameters not specified here are under the default settings. We use the implementation provided in KGEmb[7] for both **TransE** and **RefE**. **node2vec**[8] is implemented by sampling 10 random walks with a length of 80. The original implementation of **metapath2vec**[9] is used by generating 10 walks for each node as well. We set the learning rate to be {0.1, 0.01, 0.001} and reported the best performance for **GAN**[10]. **GATNE-T** is implemented by generating 20 walks with a length of 10 for each node. The results of **CompGCN**[11] are obtained using the multiplicative composition of node and relation embeddings. We adapt the Deep Graph Library (DGL) based implementation[12] of **HGT** and **RGCN** to perform the link prediction task. We use the original implementation of **asp2vec**[13] for the evaluation. For all baselines, the dimension of embedding is set to 128.

4.4.2 Evaluation on Link Prediction

We evaluate the impact of contextual embeddings using the binary link prediction task, which has been widely used to study the structure-preserving properties of node embeddings [145, 20].

Table 4.3 provides the link prediction results of different methods on five datasets using micro-F1 score and AUCROC. The prediction scores for SLiCE are reported from the context subgraph generation strategy (shortest path or random) that produces the best score for each dataset on the validation set. Compared to the state-of-the-art methods, we observe that

[7]https://github.com/HazyResearch/KGEmb
[8]https://github.com/aditya-grover/node2vec
[9]0https://ericdongyx.github.io/metapath2vec/m2v.html
[10]https://github.com/google-research/google-research/tree/master/graph_embedding/watch_your_step
[11]https://github.com/malllabiisc/CompGCN
[12]https://github.com/dmlc/dgl/tree/master/examples/pytorch/hgt
[13]https://github.com/pcy1302/asp2vec

Table 4.3: Performance comparison of different models on link prediction task using micro-F1 score and AUCROC. The symbol "OOM" indicates out of memory. Here, SLiCE$_{w/o\ GF}$ and SLiCE$_{w/o\ FT}$ represent two variants of the proposed SLiCE method by removing the Global Feature (GF) initialization and without fine-tuning (FT), respectively. The symbol * indicates that the improvement is statistically significant over the best baseline based on two-sided t-test with p-value 10^{-10}.

Type	Methods	micro-F1 Score					AUCROC				
		Amazon	**DBLP**	**Freebase**	**Twitter**	**Healthcare**	**Amazon**	**DBLP**	**Freebase**	**Twitter**	**Healthcare**
Static	**TransE**	50.28	49.60	47.78	50.60	48.42	50.53	49.05	48.18	50.26	49.80
	RefE	51.86	49.60	50.25	48.55	47.96	51.74	48.50	50.41	49.28	50.73
	node2vec	88.06	86.71	83.69	72.72	71.92	94.48	93.87	89.77	80.48	79.42
	metapath2vec	88.86	44.58	77.18	66.73	62.64	95.42	38.41	84.33	72.16	69.11
Contextual	**GAN**	85.47	OOM	OOM	85.01	81.94	92.86	OOM	OOM	92.39	89.72
	GATNE-T	89.06	57.04	OOM	68.16	58.02	94.74	58.44	OOM	72.07	73.40
	RGCN	65.03	28.84	OOM	63.46	56.73	74.77	50.35	OOM	64.35	46.15
	CompGCN	83.42	40.10	65.39	40.75	39.84	90.14	34.04	72.01	39.86	38.03
	HGT	65.77	53.32	OOM	53.13	76.54	68.66	50.85	OOM	59.32	82.36
	asp2vec	94.89	78.82	90.02	88.29	85.46	98.51	92.51	**96.61**	95.00	92.97
	SLiCE$_{w/o\ GF}$	67.01	66.02	66.31	67.07	60.88	62.87	57.52	55.31	66.69	63.11
	SLiCE$_{w/o\ FT}$	94.99	89.34	90.01	82.19	81.58	98.66	96.07	96.33	90.38	89.51
	SLiCE (Ours)	**96.00***	**90.70***	**90.26**	**89.30***	**91.64***	**99.02***	**96.69***	96.41	**95.73***	**94.94***

SLiCE significantly outperforms both static and contextual embedding learning methods by 11.95% and 25.57% on average in F1-score, respectively. We attribute static methods superior performance, compared to relation based contextual learning methods (such as GATNE-T, RGCN, and CompGCN), to the ability to capture global network connectivity patterns. Relation based contextual learning methods limit node contextualization by emphasizing the impact of relations on nodes. We outperform all methods on F1-score, including asp2vec, a cluster-aspect based contextualization method. Asp2vec achieves a marginally better AUCROC score on Freebase, but SLiCE achieves a better F1-score.

SLiCE outperforms asp2vec on all other datasets (in F1-score and AUCROC measure), improving F1-score for DBLP by 13% owing to its ability to learn important metapaths without explicit specification. These results indicate that subgraph based contextualization is highly effective and is a strong candidate for advancing the state-of-the-art link prediction in a graph network.

4.4.3 SLiCE Model Interpretation

Here we study the impact of using different subgraph contexts on link prediction performance and demonstrate SLiCE's ability to learn important higher-order relations in the graph.

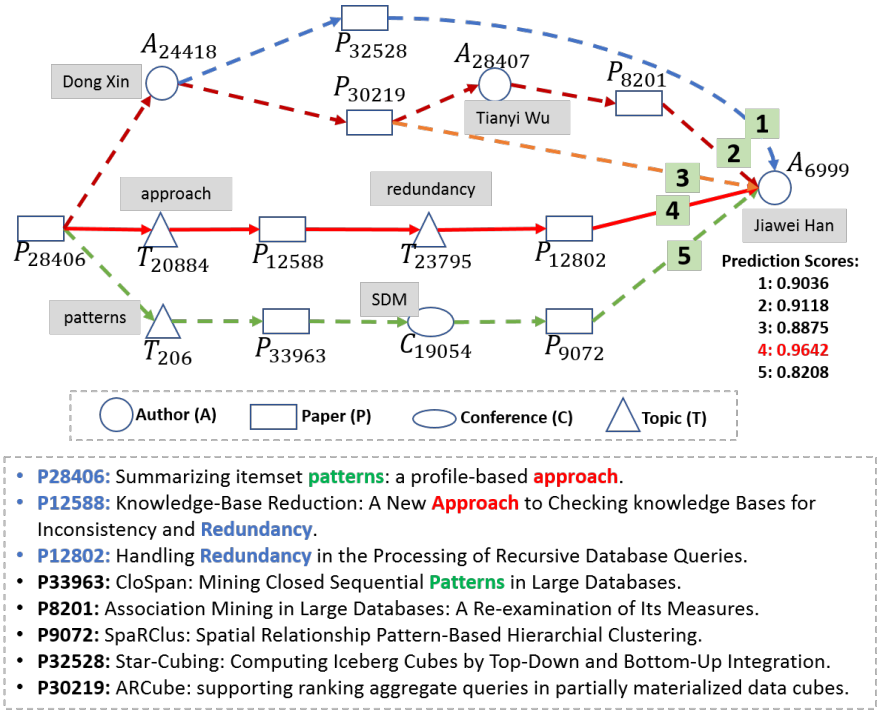


Figure 4.3: An example from DBLP. Relations in DBLP include paper-author, paper-topic and paper-conference. To predict the paper-author relationship between P_{28406} and A_{6999}, five context subgraphs are generated with a beam-search strategy. The $4th$ context subgraph (along with context subgraph 1, 2 and 3) contain closely related nodes and get high scores, while path 5 containing a generic conference node achieves lowest score.

Our analysis shows SLiCE' results are highly interpretable, and provide a way to perform explainable link prediction by learning the relevance of different context subgraphs connecting the query node pair.

Case Study for Model Interpretation

Figure 4.3 shows an example subgraph from DBLP dataset between "Jiawei Han" and "P_{28406}", a paper on frequent itemset mining. Paths 1 to 5 (shown in different legend) show different contexts subgraphs present between the two nodes. We observe that the link prediction score between Jiawei Han and paper P_{28406} varies, depending on the context subgraph provided to the model.

We also observe that SLiCE assigns high link prediction scores for all context subgraphs (paths 1-4) where all the nodes on the path share strong semantic association with other nodes in the context subgraph. The lowest score is achieved for path-5 which contains a conference node C_{19054} (SIAM Data Mining). As a conference publishes papers on multiple topics, we

hypothesize that this breaks the semantic association across nodes, and consequently lowers the probability of the link compared to other context subgraphs where all nodes are closely related.

It is important to note, that a node can share a semantic association with another node separated by multiple hops on the path, and thus would be associated via a higher-order relation. We explore this concept further in the following subsections.

Interpretation of Semantic Association Matrix

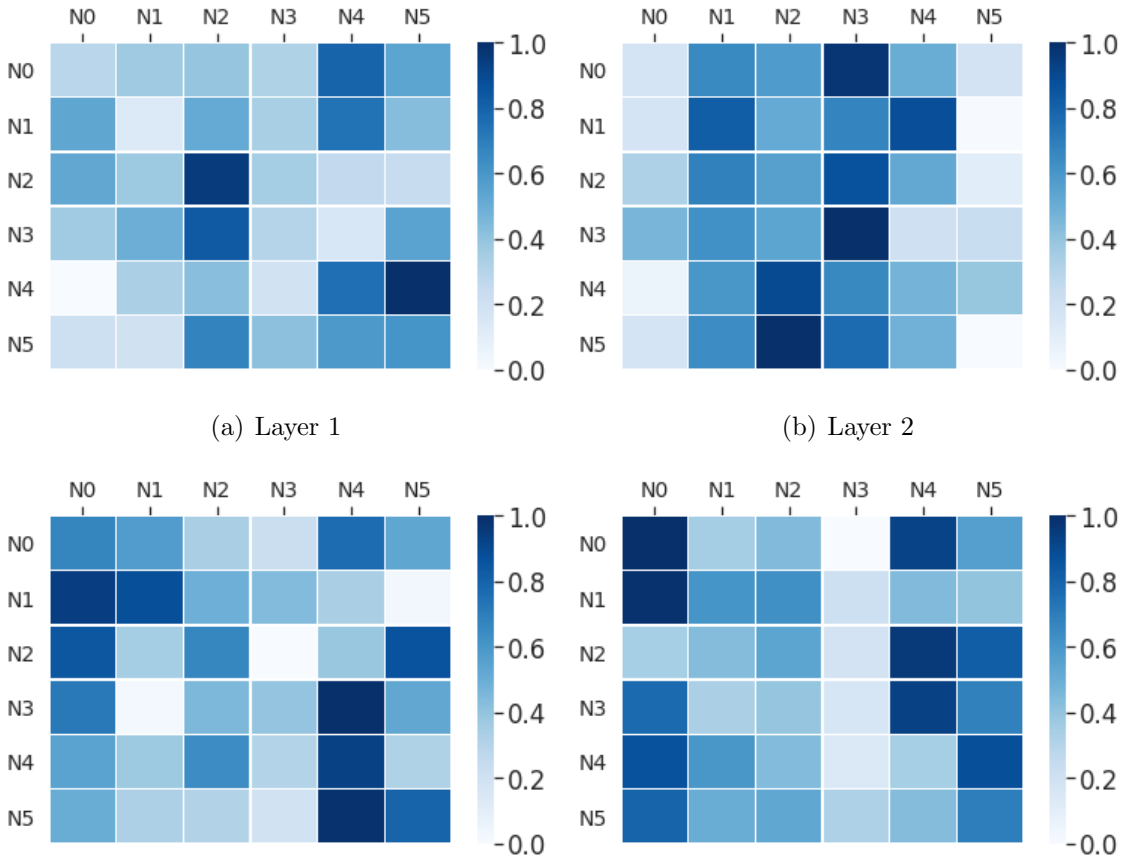
We provide the visualization of the semantic association matrix $\bar{A}^k_{ij}$ as defined in Eq. (4.1) to investigate how the node dependencies evolve through different layers in SLiCE. Given a node pair (v_i, v_j) in the context subgraph g_c, a high value of $\bar{A}^0_{ij}$, indicates a strong global dependency of node v_i on v_j. While a high value of $\bar{A}^k_{ij}(k \geq 1)$ (the association after applying more translation layers) indicates a prominent high-order relation in the subgraph context.

Figure 4.4 shows weights of semantic association matrix for the context generated for node pair (N0: *Summarizing itemset patterns: a profile-based approach (Paper)*, N1: *Jiawei Han (Author)*). Nodes in the context consist of N2: *Approach (Topic)*, N3: *Knowledge-base reduction (Paper)*, N4: *Redundancy (Topic)* and N5: *Handling Redundancy in the Processing of Recursive Database Queries (Paper).* We observe that at layer 1 (Figure 4.4(a)), the association between source node *N0* and target node *N1* is relatively low. Instead, they both assign high weights on *N4*. However, the dependencies between nodes are dynamically updated when applying more learning layers, consequently enabling us to identify higher-order relations. For example, the dependency of *N1* on *N0* becomes higher from layer 3 (Figure 4.4(c)) and *N0* primarily depends on itself without highly influenced by other nodes in layer 4 (Figure 4.4(d)). This visualization of semantic association helps to understand how the global embedding is translated into the localized embedding for contextual learning.

Symbolic Interpretation of Semantic Associations via Metapaths

Metapaths provide a symbolic interpretation of the higher-order relations in a heterogeneous graph. We analyze the ability of SLiCE to learn relevant metapaths that characterize positive semantic associations in the graph. We observe from Table 4.4 that SLiCE is able to match existing metapaths and also identify new metapath patterns for prediction of each relation type. For example, to predict the paper-author relationship, SLiCE learns three shortest metapaths, including "TPA" (authors publish with the same topic), "APA" (co-authors) and "CPA"(authors published in the same conference).

Interestingly, our learning suggests that longer metapath "APCPA", which is commonly used to sample academic graphs for co-author relationship, is not as highly predictive of a positive relationship, i.e., "all authors who publish in the same conference *do not* necessarily publish



(a) Layer 1

(b) Layer 2

(c) Layer 3

(d) Layer 4

Figure 4.4: Visualization of the semantic association matrix (after normalization) learnt from different layers on a DBLP subgraph for link prediction between paper N0 and author N1. An intense color indicates a higher association. Initially (layer 1), nodes N0 and N1 have low association. In layer 4, SLiCE learns higher semantic association from N1 to N0.

Table 4.4: Comparisons of metapaths learned by SLiCE with predefined metapaths on DBLP dataset for each relation type. Here, P, A, C, and T represent Paper, Author, Conference, and Topic, respectively.

Learning Methods	**Paper-Author**	**Paper-Conference**	**Paper-Topic**
Predefined [142]	APCPA, APA	-	-
SLiCE + Shortest Path	TPA, APA, CPA	TPC, APC, TPTPC	TPT, CPT, APT
SLiCE + Random	APA, APAPA	TPTPC, TPAPC	TPTPT, APTPT

together". Overall, the metapaths reported in Table 4.4 are consistent with the top ranked paths in Figure 4.3. These metapaths demonstrate SLiCE's ability to discover higher order semantic associations and perform interpretable link prediction in heterogeneous networks.

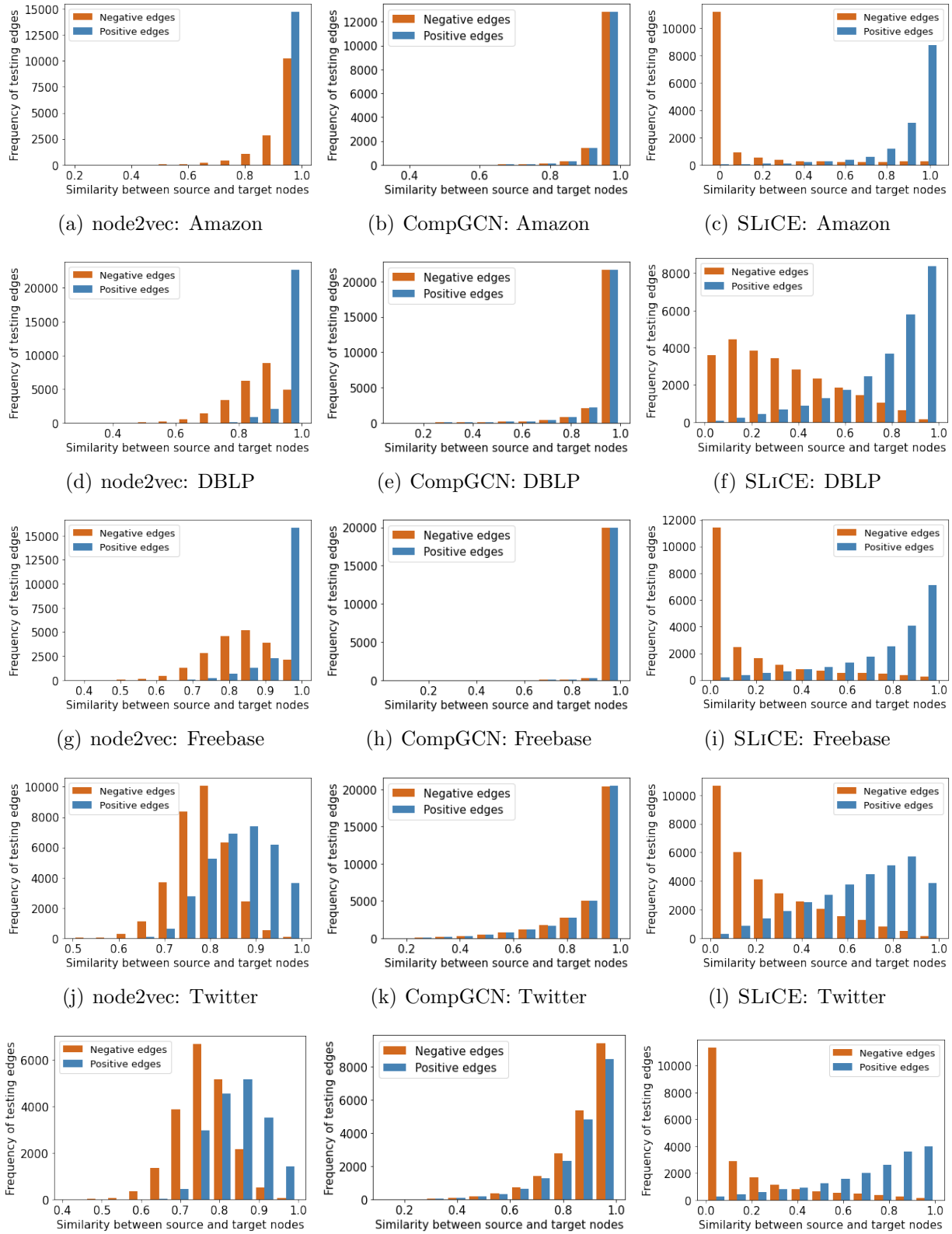
4.4.4 Effectiveness of Contextual Translation for Link Prediction

In this section, we study the impact of contextual translation on node embeddings. First, we evaluate the impact of contextualization in terms of the similarity (or distance) between the query nodes. Second, we analyze the effectiveness of contextualization as a function of the query node pair properties. The latter is especially relevant for understanding the performance boundaries of the contextual methods.

Impact of Contextual Translation on Embedding-based Similarity

Figure 4.5 provides the distribution of similarity scores for both positive and negative edges obtained by SLiCE. We compare against embeddings produced by node2vec [40] which is one of the best performing static embedding methods (see Table 4.3) and CompGCN [114] a relation based contextualization method. We observe that for node2vec and CompGCN, the distribution of similarity scores across positive and negative edges overlaps significantly for all datasets. This indicates that the embeddings learnt from global methods or relation specific contextualization cannot accurately differentiate the positive and negative edges in link prediction task.

On the contrary, SLiCE increases the margin between the distributions of positive and negative edges significantly. It brings node embeddings in positive edges closer and shifts nodes in negative edges farther away in the low-dimensional space. This indicates that the generated subgraphs provide informative contexts during link prediction and enhance embeddings such that they improve the discriminative capability of both positive and negative node-pairs.



(a) node2vec: Amazon (b) CompGCN: Amazon (c) SLiCE: Amazon

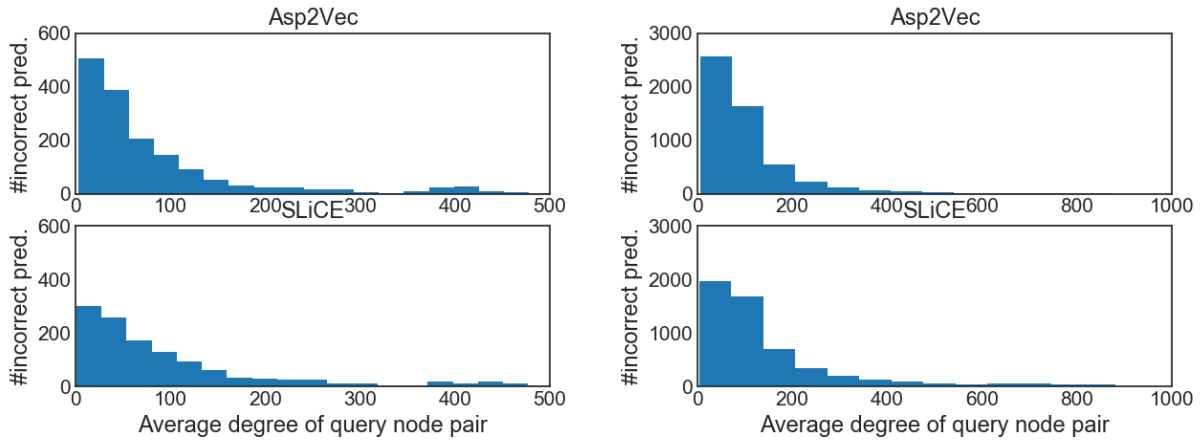
(d) node2vec: DBLP (e) CompGCN: DBLP (f) SLiCE: DBLP

(g) node2vec: Freebase (h) CompGCN: Freebase (i) SLiCE: Freebase

(j) node2vec: Twitter (k) CompGCN: Twitter (l) SLiCE: Twitter

(m) node2vec: Healthcare (n) CompGCN: Healthcare (o) SLiCE: Healthcare

Figure 4.5: Distributions of similarity scores of both positive and negative node-pairs obtained by node2vec, CompGCN, and SLiCE over five datasets.



(a) Amazon (b) Healthcare

Figure 4.6: Error analysis of contextual translation based link prediction method as a function of degree-based connectivity of query nodes.

Error Analysis of Contextual Methods as a function of Node Properties

We investigate the performance of SLiCE and the closest performing contextual learning method, asp2vec [77], as a function of query node pair properties. Given each method, we select all query node pairs drawn from both positive and negative samples that are associated with an incorrect prediction. Next, we compute the average degree of the nodes in each such pair. We opt for degree and ignore any type constraint for its simplicity and ease of interpretation. Figure 4.6 shows the distribution of these incorrect predictions as a function of average degree of query node pair. It can be seen that, for Amazon and Healthcare datasets, most of the incorrect predictions are concentrated around the query pairs with low and medium values of average degree. However, SLiCE has fewer errors than asp2vec for such node pairs. This can be attributed to the *aspect-oriented* nature of asp2vec, which maps each node to a fixed number of aspects. Since nodes in a graph may demonstrate varying degree of aspect-diversity, mapping a node with low diversity to more aspects (than it belongs to) reduces asp2vec's performance. SLiCE adopts a complementary approach, where it considers the subgraph context that connects the query nodes, leading to better contextual representations.

4.4.5 Study of SLiCE

Parameter Sensitivity

In Figure 4.7, we provide the link prediction performance with micro-F1 score on four datasets by varying four parameters used in SLiCE model, including number of heads, number of

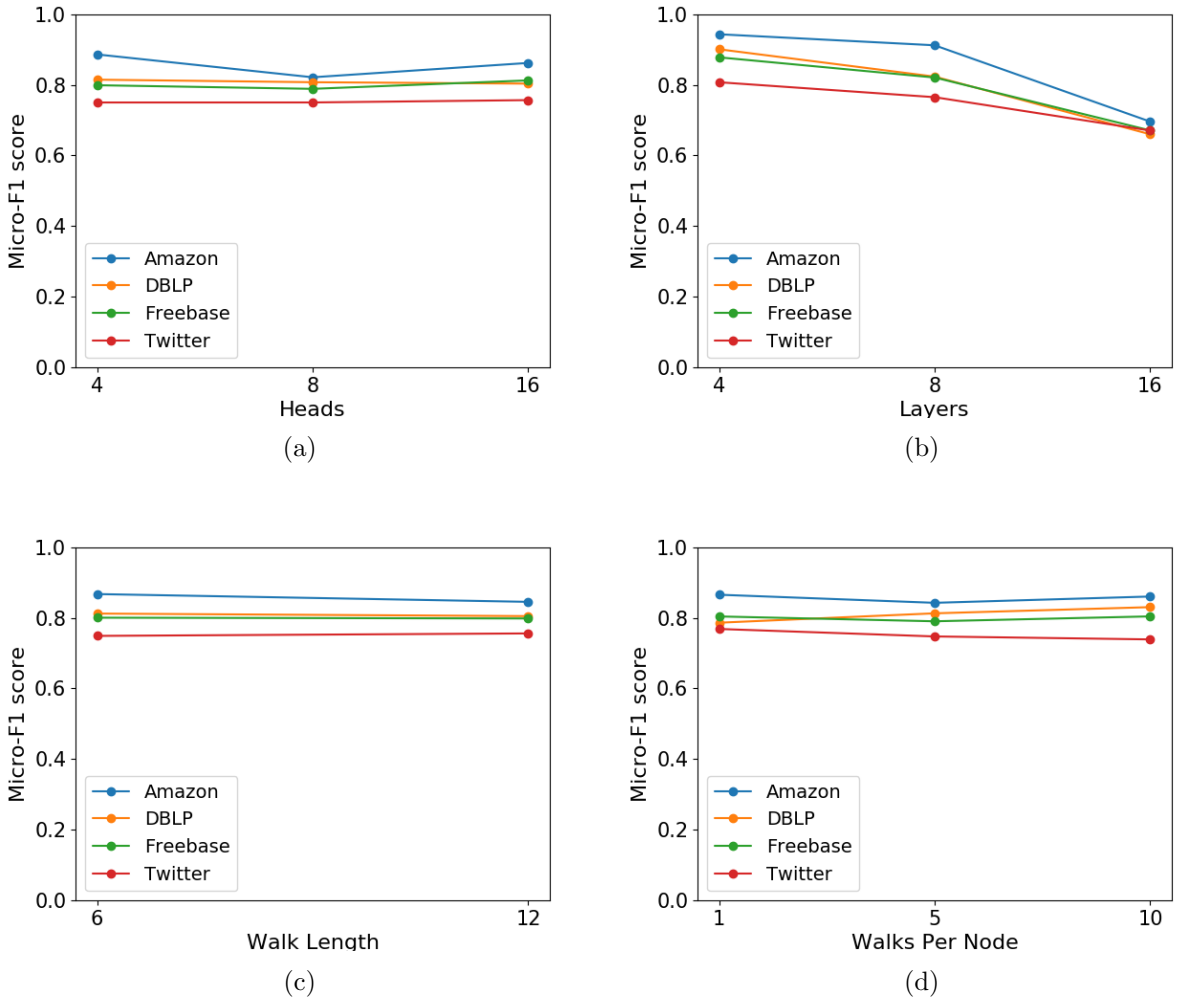


Figure 4.7: Micro-F1 scores for link prediction with different parameters in SLiCE on four datasets.

contextual translation layers, number of nodes in contexts (*i.e.*, walk length), and the number of (context) walks generated for each node in pre-training. The performance shown in these plots are the averaged performance by fixing one parameter and varying other three parameters.

On the other hand, when varying the parameter *number of layers*, we observe that applying four layers of contextual translation provides the best performance on all the datasets; the performance dropped significantly when stacking more layers. This indicates that four contextual translation layers are sufficient to capture the complex higher-order relations over various knowledge graphs. Based on these analyses, we set the default values for both number of heads and the number of layers to be 4, and generate one walk for each node with a length

of 6 in the pre-training step.

Effect of Pre-trained Global Features (GF)

To explore the impact of pre-trained global node features on the performance of SLiCE, we perform an ablation study by analyzing a variant of SLiCE with four contextual translation layers. More specifically, the pre-trained global embeddings are disabled in SLiCE, termed SLiCE$_{w/o\ GF}$. The results are provided in Table 4.3. We observe that without initialization using the pre-trained global embeddings, the model performance of SLiCE decreased on all five datasets in both metrics. The reason is that, compared to the random initialization, the global node features are able to represent the role of each node in the global structure of the knowledge graph. By further applying the proposed contextual translation layers, they can collaboratively provide contextualized node embeddings for downstream tasks like link prediction.

Effect of Fine-tuning (FT)

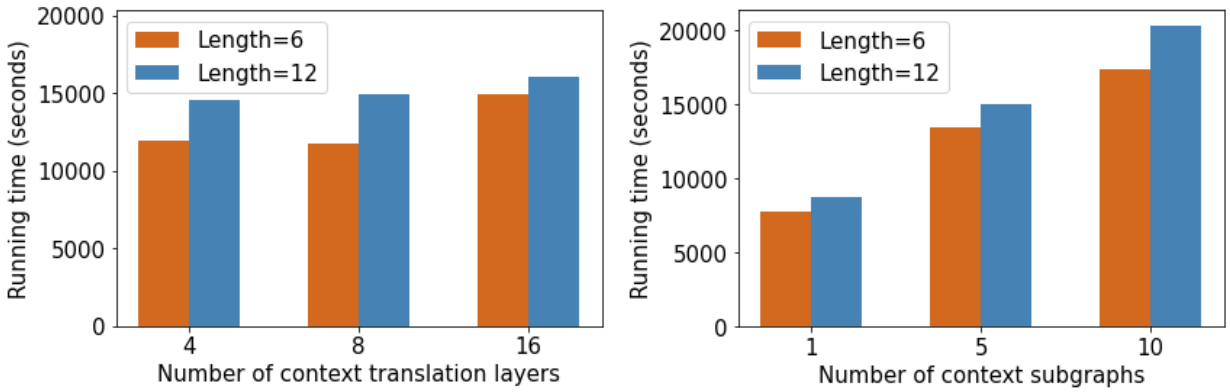
To investigate the effect of fine-tuning stage on learning the contextual node embeddings, we disable the fine tuning layer for supervised link prediction task, termed SLiCE$_{w/o\ FT}$ and show the results in Table 4.3. Compared to the baseline methods, it still achieves competitive performance. We attribute this to the effectiveness of capturing higher-order relations through the contextual translation layers. While compared to the full SLiCE model, the performance of SLiCE$_{w/o\ FT}$ degrades slightly on Amazon, DBLP, and Freebase datasets, but significantly decreases on both Twitter and MIMIC datasets. This can be attributed to the fact that supervised training with the link prediction task is able to learn the fine-grained contextual node embedding for link prediction task.

4.4.6 SLiCE Model Complexity

Contextual learning methods are known to have high computational complexity. In Section 4.3.5, we observed that the cost of training SLiCE is approximately linear to the number of edges. In this section, we provide an empirical evaluation of the model scalability in view of the prior analysis.

Time Complexity Analysis

In this subsection, we mainly investigate the impact of the following three parameters on the overall time complexity of the SLiCE model: (1) number of contextual translation layers, (2) number of context subgraphs, and (3) length of context subgraph.



(a) (b)

Figure 4.8: Analysis of the time complexity of SLiCE on Freebase dataset by varying (a) number of translation layers and (b) number of context subgraphs considered for each node.

- We study the scalability of the SLiCE model when the number of context translation layers are varied and the corresponding plots are provided in Figure 4.8(a). The x-axis and y-axis represent the number of layers and running time in seconds, respectively. The plots indicate that increasing the number of layers does not significantly increase the training time of the SLiCE model.
- In Figure 4.8(b), we demonstrate the impact of the number of context subgraphs on the time complexity. Increasing the number of context subgraphs generated for each node in pre-training and each node-pair for fine-tuning raises the number of training edges which further increases the training time of the model.

These two plots empirically verify the analysis about the model complexity, discussed in Section 4.3.5, that the proposed SLiCE model is approximately linear to the number of edges in the graph and does not depend on other parameters such as the number of contextual translation layers and the number of nodes in the graph. In addition, we also vary the length (number of nodes) of the context subgraph in both plots. The plot shows that even doubling the context length will not significantly increase the running time. This time complexity analysis, combined with the performance results in Table 4.3 and the parameter sensitivity analysis in Figure 4.7 can jointly provide the guidelines for parameter selection.

Context Subgraph Sampling Analysis

In the complexity analysis discussed in Section 4.3.5, we approximated the total number of training edges in the entire graph as $N_E \approx |V| * N_{cpn}$, where $|V|$ represents the number of nodes in graph G and N_{cpn} denotes the number of context subgraphs generated for each node. This estimation also provides us guidelines for determining the number of context subgraphs

Table 4.5: Estimation of the number of context subgraphs for each node in the knowledge graph.

Dataset	**Amazon**	**DBLP**	**Freebase**	**Twitter**	**Healthcare**
# Nodes ($\|V\|$)	10,099	37,791	14,541	9,990	4,683
# Edges ($\|E\|$)	129,811	170,794	248,611	294,330	205,428
# Contexts (N_{cpn})	7.74	2.71	10.26	17.67	26.32

for each node N_{cpn}. By incorporating $N_E = \alpha_T|E|$ into the approximation (where $|E|$ is the total number of edges in graph G and α_T is the ratio of training split), we can estimate the number of context subgraphs per node as $N_{cpn} = \alpha_T|E|/|V|$. Table 4.5 shows the estimated numbers (with $\alpha_T = 0.6$) for the five datasets used in this work. These estimations provide us an approximate range for the value of N_{cpn} during the context generation step. Based on this analysis, in our experiments, we generally consider 1, 5, and 10 for the value of N_{cpn} on all the five datasets in both pre-training and fine-tuning stages.

4.5 Summary

We introduce SLiCE framework for learning contextual subgraph representations. Our model brings together knowledge of structural information from the entire graph and then learns deep representations of higher-order relations in arbitrary context subgraphs. SLiCE learns the composition of different metapaths that characterize the context for a specific task in a drastically different manner compared to existing methods which primarily aggregate information from either direct neighbors or semantic neighbors connected via certain pre-defined metapaths. SLiCE significantly outperforms several competitive baseline methods on various benchmark datasets for the link prediction task. In addition to demonstrating SLiCE's interpretability and scalability, we provide a thorough analysis of the effect of contextual translation for node representations. In summary, we show SLiCE's subgraph-based contextualization approach is effective and distinctive over competing methods.

Chapter 5

Tensor-based Temporal Multi-Task Survival Analysis

This chapter presents a tensor-based temporal multi-task learning framework for survival analysis. Section 5.1 provides the introduction and motivation to this work. In Section 5.2, the background and related work about survival analysis and multi-task learning is briefly reviewed. Section 5.3 introduces the details about the proposed temporal multi-task learning algorithm along with the optimization method used to solve the model. The prediction performance of the proposed MTMT algorithm is evaluated using the real-world datasets in Section 5.4. Finally, Section 5.5 concludes the discussion of this work.

5.1 Introduction

The primary goal of survival analysis is to predict time to the event of interest on longitudinal data obtained during a particular observation period [65]. Survival analysis methods have been successfully applied in many real-world applications, such as healthcare [86], social networks [24], reliability [72] and customer lifetime value [143]. One unique phenomenon, known as *censoring*, in the longitudinal data is the occurrence of the event may not always be observed for all instances during the observation due to non-occurrence of the event by the end of the observation or losing follow-up during the observation. Then in the longitudinal data, the event occurrence time of the instances which experienced the event of interest and the censoring time of the censored instances will be recorded as the observation time, respectively. This implies that people need to wait for a long time to collect the training data with sufficient event occurrences in longitudinal studies.

Most of these existing survival analysis methods are developed to solve the survival problem with a single specific event of interest at a given time point. In addition, the effect of the number of events in the dataset on the model performance is not considered in the standard

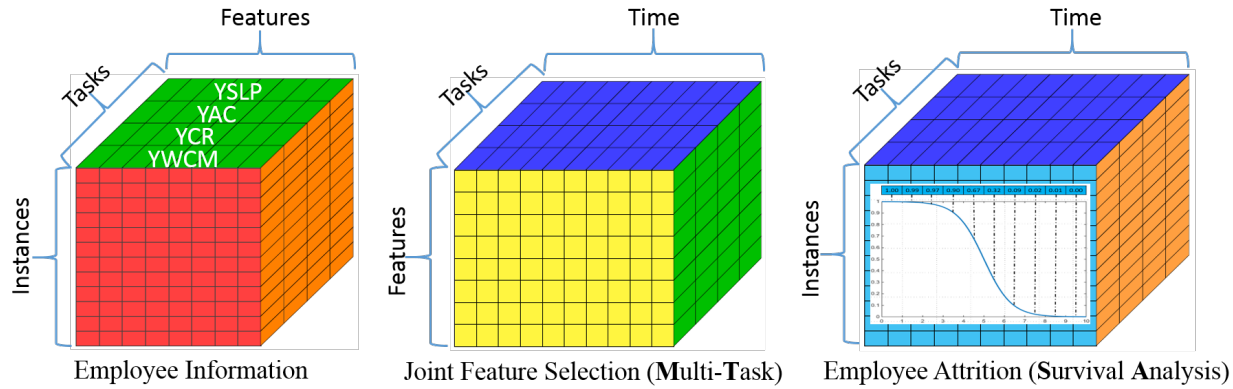


Figure 5.1: An illustration of various components in the proposed MTMT framework using Employee Attrition dataset.

survival analysis methods, even if the percentage of censoring is very high [64]. However, in the real-world applications, the data about several related longitudinal studies is available and can be modeled simultaneously. For example, in the human resource management domain, the information of employees before leaving, such as years stayed in the company and years worked with the manager, can be easily extracted. In other words, predicting the occurrence of multiple events of interest simultaneously is needed in the real-world applications. In addition, it is also important if we can dynamically follow up the survival status of the instances in each longitudinal study.

A naive way to solve this problem is to simply apply the standard survival analysis method independently to each task at each specific time point. However, it often leads to a sub-optimal solution since the underlying dependencies between these tasks and the correlations of each single task over time are ignored. On one hand, the dependencies between tasks (called *inter-task correlations*) are good ways to handle the insufficiency of event occurrences in each single task to perform the time to event prediction. On the other hand, the survival status of instances at the adjacent time points should be similar. Thus, for each single task, the problems at different time points should also have some correlations (called *intra-task temporal smoothness*). Thus, both types of correlations will need to be incorporated into a model that can jointly analyze multiple tasks by leveraging the limited number of events available for each single task.

In this work, we will formalize a temporal (**M**ultiple **T**ime points) **M**ulti-**T**ask learning (MTMT) framework using tensor representation for survival analysis to overcome the weakness of the standard survival analysis methods and incorporate the two types of correlations (mentioned above) in the training process [126]. Multi-task learning method is used in the literature to learn common feature subsets across all tasks [65, 153]. More specifically, in our work, multi-task learning will be dynamically applied to several related survival tasks and the common feature subset will be the unified feature representation across all tasks and

all time points. Since there are multiple tasks and multiple time points in our problem, a 3-way tensor will also be introduced in this work for effective representation. Therefore, we extend the standard $\ell_{2,1}$ norm which is usually used on a matrix to the $\ell_{F,1}$ norm (which is appropriate for the tensor representation) to incorporate the inter-task correlations in the proposed MTMT model. Another contribution of the MTMT algorithm is that it performs the multi-task learning dynamically. In this case, the survival status of the instances about multiple events of interest can be monitored over time, then, the survival function for the problem in each task can also be obtained. If the standard multi-task learning formulation is used, it will not be able to handle the censored data in survival problems. For solving the survival analysis problems, we will formulate a multi-way indicator tensor which can incorporate the censored instances depending on their survival status over time for each task. In this case, at each time point, one given instance will be involved in the training process as long as it is not censored in at least one task. Figure 5.1 illustrates the main components of the proposed MTMT method on the Employee Attrition dataset in more detail.

In MTMT, we incorporate both the censoring constraint and two types of correlations by applying regularizations in the loss function of multi-task learning formulation and the Alternating Direction Methods of Multipliers (ADMM) algorithm [15]. In addition, the proximal gradient decent algorithm [8] with backtracking linear search is extended to optimize the $\ell_{F,1}$ regularized problems. The proposed MTMT method will be evaluated on both the employee attrition dataset provided by IBM Watson Analytics and the Medical Information Mart for Intensive Care III (MIMIC III) dataset [55, 39] and compared with the state-of-the-art survival analysis and multi-task learning methods.

The main contributions of this work are summarized as follows:

- Propose a multiple time points multi-task learning (MTMT) model for survival analysis which has the ability to analyze several survival problems jointly and predict the survival probability at each time point dynamically.
- Develop an ADMM based algorithm to optimize the proposed MTMT problem.
- Evaluate the performance of the proposed temporal multi-task learning method using multiple real-world datasets and compare with several state-of-the-art methods.

5.2 Related Work

In this section, we discuss the related works on survival analysis, multi-task learning and temporal modeling. We will also distinguish our contribution in this work from other existing algorithms in the literature.

5.2.1 Survival Analysis

Survival analysis is a branch of statistics which aims to handle censored data and then predict the time to the event of interest [123]. Typically, survival analysis methods can be grouped into three categories: non-parametric methods, Cox-based methods and parametric methods.

Kaplan-Meier (KM) method [57] is the most commonly used non-parametric survival analysis method due to its reasonable ability to estimate the survival function, which is one of the most important functions in survival problems. Generally, the KM method estimates the survival probability at a given time point during the observation period as the product of the survival rate at this time point and the same estimate at the previous time.

The Cox proportional hazards model [25] is well studied in both statistics and data mining fields due to its flexibility in choosing the baseline hazard function and its ability to provide a more consistent estimation. Cox model is built on a proportional hazard assumption and it estimates the parameters using the partial likelihood function. In order to adapt Cox model to the survival problem with high-dimensional features, the regularized Cox models [104, 94, 118] are proposed to perform the feature selection and identify the most significant features to the outcome variable.

Different from the non-parametric and Cox-based methods, parametric methods assume that the survival time (or its logarithm) of all instances follow a certain theoretical distribution [62]. The commonly used distributions include exponential, weibull, logistic, log-logistic and log-normal distribution. The accelerated failure time (AFT) model [128] is one of the widely used parametric methods since it assumes that the logarithm of the survival times follows one theoretical distribution. In addition, several linear models, including the Buckley-James (BJ) [16] and Tobit regression [106] are proposed based on the standard linear regression to handle the survival analysis problems.

Other researchers have proposed deep survival analysis methods [58, 85, 3, 37] to further tackle the non-linear relationships between covariates and survival time. DeepSurv is proposed as a treatment recommender system that combines the power of both Cox proportional hazards model and deep neural networks and modelling the interactions between covariates and the effect of treatment [58]. The hierarchical generative deep survival analysis approach in [85] is proposed to handle the sparsity and heterogeneity in EHR data using a sequence of multi-layer probability models. It is worth noting that all the survival analysis methods mentioned above are suitable for the problems involving only one event of interest or single time point. Moreover, the number of event occurrences will affect the performance of the standard survival analysis methods, especially when the number of events is insufficient at the early stage of the observation period. An alternative way to compensate for this insufficiency is to make effective use of the event information that is available in other survival problems about a related event of interest. The goal of this work is to solve the survival analysis problem with multiple events of interest jointly and improve the prediction performance of each single problem. An intuitive way to solve the survival problem with more than one

event of interest is to simply apply the standard survival algorithm independently on each of the events of interest. However, the correlations between these multiple tasks will be ignored. In this work, we dynamically consider the survival problem with multiple correlated events of interest by using the multi-task learning method.

5.2.2 Multi-task Learning

Multi-task learning (MTL) method aims at improving the prediction performance of each task by analyzing multiple tasks jointly and learning shared features across all tasks. Compared to the methods which learn each task independently, it has been shown both theoretically [34, 4] and empirically [65, 153, 148, 28] that multi-task learning simultaneously improves the performance of each task. Multi-task learning has been widely used in many real-world applications, including biomedical informatics [65, 153] and computer vision [148, 28]. However, there is not much work on multi-task learning in the field of survival analysis. The authors in [65] developed a linear regression based multi-task learning formulation (MTLSA) to predict the survival status at each time point. In their method, the survival problem at each time point is considered as one single task and the $\ell_{2,1}$ norm penalty is applied in order to learn a shared feature subset across different tasks. However, only one event of interest is considered in MTLSA and each of the multiple tasks is formulated at each time point, which cannot solve the problems with multiple events of interest as described in our work. In addition, deep multi-task Gaussian processes are applied in a non-parametric Bayesian model to solve survival problems with competing risks [3]. A neural multi-task logistic regression model is proposed in [37] to offset the linearity problem in traditional survival models. These multi-task learning based methods are able to analyze different survival analysis problems jointly, however, they do not consider the correlations between the problems at different time points.

In our work, we develop a new multi-task learning method to jointly analyze several related survival analysis problems and dynamically perform such multi-task learning to monitor the survival status of instances in each individual task. In other words, our proposed method provides the ability to generate the survival function for each event of interest by predicting the survival probability for each instance over time. To optimize the proposed MTMT method, several types of constraints, including censoring, inter-task correlations and intra-task temporal smoothness, are considered in the objective function to incorporate the unique properties of our survival problem. To the best of our knowledge, this work is the first work about dynamically applying the multi-task learning in the field of survival analysis and also the first work to solve the survival analysis problem using tensor representation.

5.3 Proposed Model

In this section, we will first describe the problem formulation in the form of a joint multi-task learning and survival analysis framework by considering multi-task problems dynamically. Then, an ADMM based optimization algorithm will be proposed to solve the joint optimization problem. The convergence and complexity analysis of the proposed algorithm will also be discussed at the end of this section.

5.3.1 The MTMT Framework

Survival analysis methods are well known for their ability to handle the censored instances in the data in order to predict the time to event of interest. Generally, the standard survival analysis algorithm is proposed to estimate the time to a single event of interest. More specifically, it solves the single task (single event of interest) problem (eg. task k), in which the instance i in survival data is recorded by a triplet (X_i^k, T_i^k, Y_i^k) [65], where $X_i^k \in \mathbb{R}^{1\times P}$ is a vector which represents the feature vector; Y_i^k is a binary variable which indicates the event status, i.e., $Y_i^k = 1$ if an event is observed on instance i and the corresponding time (O_i^k) for the event is recorded as T_i^k, and $Y_i^k = 0$ if instance i is censored during the observation time period and then its censored time (C_i^k) will be recorded as T_i^k [65]. In other words, for each instance in survival data, only the survival time or the censored time can be observed during the observation time period, i.e.,

$$T_i^k = \begin{cases} O_i^k & \text{if } Y_i^k = 1 \\ C_i^k & \text{if } Y_i^k = 0 \end{cases} \tag{5.1}$$

It should be noted that the event time O_i^k for the censored instance is a latent value since there is no information about the event status after the censored time C_i^k. One of the main goals of survival analysis is to measure the survival probability at each time point and estimate the time to the event of interest for each instance.

However, there are certain disadvantages for the standard survival analysis methods. A sufficient number of event occurrences in the training data must be collected by waiting for a long period of time, otherwise having fewer event occurrences may affect the prediction performance of the survival analysis algorithm, especially in the early stage of the observation [35]. It will be challenging but beneficial for the prediction of each single problem if we can make the best use of the event information in multiple correlated regular survival analysis problems and analyze these individual survival analysis problems in a joint manner. More specifically, the event information in other correlated tasks can be used to compensate for the insufficiency of the event occurrence in a single task and improve the prediction performance. However, due to the new occurrence of censoring or events on some instances, the values of the event label for these instances will be updated at the corresponding event time or censored time. It will be vital to monitor the survival status of each instance dynamically,

which cannot be solved by the existing survival analysis algorithms. These disadvantages motivate the need for developing an algorithm which can analyze multi-task survival analysis problems dynamically.

In this work, the temporal multi-task learning framework is proposed to estimate the survival probabilities of the instances in K tasks over J time points. Specifically, each event of interest is considered as a single task, and we simultaneously analyze all the tasks by considering both the inter-task correlations and the intra-task time smoothness. Figure 5.1 shows various components used in the proposed MTMT framework. In the input data, the data record for instance i ($i = 1, \cdots, N_k$) in task k ($k = 1, \cdots, K$) at each time point t_j ($j = 1, \cdots, J$) can be obtained as a triplet $(\mathbf{X}_{i:k}, \mathbf{T}_{ijk}, \mathbf{Y}_{ijk})$ using the following three procedures:

1. $\mathbf{X}_{i:k} \in \mathbb{R}^{1\times P}$ represents the feature vector for instance i in task k. Due to the difficulty and limitation to collect the feature values over time, the static feature values are commonly used in the survival analysis literature.

2. $\mathbf{Y}_{ijk}$ is the binary variable indicating the survival status of instance i at time point t_j in task k. It is obtained as follows:

$$\mathbf{Y}_{ijk} = \begin{cases} 0 & \text{if } Y_i^k = 1 \text{ and } t_j \geq O_i^k \\ 1 & \text{Otherwise} \end{cases} \tag{5.2}$$

   where $\mathbf{Y}_{ijk} = 1$ indicates that there is no event observed for instance i at t_j in task k and $\mathbf{Y}_{ijk} = 0$ indicates that instance i did not survive after t_j in task k. Eq. (5.2) indicates that, for an instance having an event during the observation time in task k, it will survive until the event time O_i^k, which is the event time of this instance in the task. After the event time O_i^k, it will be labeled as 0 in the data collected at each time point which is the unique property of the non-recurrent event survival analysis problem. The instances without an event occurrence during the observation time will be labeled as 1 over all time points. It should be noted that these instances have definitely survived at each time point until the censored time, however, the survival status after the censoring time is unknown even if we label it as 1. These unknown records will be carefully handled by an indicator formulation (to be introduced in Subsection 5.3.2).

3. $\mathbf{T}_{ijk}$ is obtained according to the value of Y_i as follows:

$$\mathbf{T}_{ijk} = \begin{cases} O_i^k & \text{if } \mathbf{Y}_{ijk} = 0 \\ \min(t_j, C_i^k) & \text{if } \mathbf{Y}_{ijk} = 1 \text{ and } Y_i^k = 0 \\ \min(t_j, O_i^k) & \text{if } \mathbf{Y}_{ijk} = 1 \text{ and } Y_i^k = 1 \end{cases} \tag{5.3}$$

It should be noted that each instance is not required to be observed for all tasks. If instance i is not observed in task k, then all values in the triplet $(\mathbf{X}_{i:k}, \mathbf{T}_{ijk}, \mathbf{Y}_{ijk})$ at each time point t_j will be set to 0. This representation leads to equal number of instances in each of the tasks.

Based on the data transformation procedures, the entire data for task k at all the time points can be represented as $(\mathbf{X}_{::k}, \mathbf{T}_{::k}, \mathbf{Y}_{::k})$, where $\mathbf{X}_{::k} \in \mathbb{R}^{N_k \times P}$, $\mathbf{T}_{::k} \in \mathbb{R}^{N_k \times J}$ and $\mathbf{Y}_{::k} \in \mathbb{R}^{N_k \times J}$ represent matrices of the features, survival status and event time, respectively. Following the three procedures, the survival problem in each task can be transformed into a sequence of similar problems ordered by time. Our goal is to analyze the $J \times K$ problems jointly in order to completely utilize the event information available in each task, the inter-task correlations and the intra-task temporal smoothness. It can be observed that this proposed temporal (multi-time points) multi-task problem (MTMT) can be considered as a generalized framework for solving the survival analysis problems. Other existing problems in the literature can be considered as the special cases of MTMT as described below:

1. The standard survival analysis problem is a special case of MTMT with single task at single time point (STST), in which $J = 1$ and $K = 1$.
2. The MTLSA method proposed in [65] can be considered as a special case of MTMT with single task at multi-time points (MTST), in which $K = 1$.
3. Besides STST and MTST problems, MTMT can also incorporate the method in [122] as a multiple tasks at a single time point (STMT), in which $J = 1$.

All the three problems can be derived from our generalized MTMT framework. Therefore, it is important to find a way to solve the MTMT algorithm efficiently. A commonly used approach to solve the formulated temporal multi-task learning problems is to optimize $J \times K$ linear regression problems as follows:

$$\min_{\mathbf{B}} \sum_{j=1}^{J} \sum_{k=1}^{K} \frac{1}{2} ||\mathbf{Y}_{:jk} - \mathbf{X}_{::k}\mathbf{B}_{:jk}||_2^2 + R(\mathbf{B}) \tag{5.4}$$

where $\mathbf{X}_{::k} \in \mathbb{R}^{N_k \times P}$ is the feature matrix of the kth task, and $\mathbf{Y}_{:jk} \in \mathbb{R}^{N_k \times 1}$ and $\mathbf{B}_{:jk} \in \mathbb{R}^{P \times 1}$ represent the response vector and the estimated coefficient vector in the kth task at time point t_j, respectively. It should be noted that the estimated coefficients across all the tasks and all the time points can also be represented as a 3-way tensor $\mathbf{B} \in \mathbb{R}^{P \times J \times K}$ since we assume that the features in different tasks are homogeneous in this work. The term $R(\mathbf{B})$ in Eq. (5.4) denotes the regularization which is used to avoid the over-fitting of the model and to incorporate other constraints for the parameters.

5.3.2 Constraints and Regularization Function

There are mainly two challenges in the proposed MTMT algorithm. (i) As defined in Eq. (5.2), we only know that the censored instances survive until the censoring time, while the survival status after the censoring time is not available. Either treating them as survival or non-survival may introduce some bias into the model. (ii) *The goal of the MTMT*

algorithm is to incorporate the inter-task correlations and intra-task temporal smoothness into the algorithm in order to estimate the survival probabilities over time and increase the prediction performance of each single task. It is critically important to find a reasonable way to incorporate both properties into the algorithm. In this subsection, we will discuss and determine suitable regularization terms to incorporate these characteristics into the proposed model.

Censoring

Censoring is one of the main characteristics of survival data, i.e., the components for the censored instances in the survival status vector $\mathbf{Y}_{:jk}$ for task k at time point t_j is latent since we do not know the exact time until which the instances survive beyond the censoring time [86]. The censoring problem in the temporal multi-task learning algorithm is more complex due to the dynamically changing behavior of the survival status in each task. More specifically, the censoring time or event time, for one instance, in different tasks may be different, which also leads to the difference in survival status at different time points in different tasks. Thus it is important to handle this censored information in order to jointly utilize the data in each task and improve the prediction in each single task. We formulate a 3-way indicator tensor to handle these censored instances in the training process.

For the task k at time point t_j, each component indicator $\mathbf{S}_{ijk}$ defined in Eq. (5.5) is used to indicate the extent of contribution of the survival information about instance i in the model.

$$\mathbf{S}_{ijk} = \begin{cases} 0 & \text{if } \mathbf{Y}_{ijk} = 1 \text{ and } t_j > C_i^k \\ 1 & \text{Otherwise} \end{cases} \tag{5.5}$$

Then the objective function in Eq. (5.4) can be updated to handle the censoring information as:

$$\min_{\mathbf{B}} \sum_{j=1}^{J} \sum_{k=1}^{K} \frac{1}{2} ||\mathbf{S}_{:jk} \odot (\mathbf{Y}_{:jk} - \mathbf{X}_{::k}\mathbf{B}_{:jk})||_2^2 + R(\mathbf{B}) \tag{5.6}$$

where the symbol $\odot$ which represents the component-wise multiplication of two vectors is defined as $a \odot b = \sum_i a_i b_i$. By applying the indicator tensor $\mathbf{S}$, one instance will contribute to the training process only if it is not censored in at least one of the tasks, no matter if it is survived or not in these tasks.

Overfitting

In order to avoid the overfitting of the problem, a ℓ_F regularization [124] on the coefficient tensor $\mathbf{B}$ defined in Eq. (5.7) will be used.

$$||\mathbf{B}||_F^2 = \sum_{k=1}^{K} ||\mathbf{B}_{::k}||_F^2 \tag{5.7}$$

Inter-task correlations

One of the goals of multi-task learning is to compensate for the insufficiency of the event occurrences in each task and to learn shared features among all the tasks by jointly learning the models. In other words, multi-task learning aims at identifying the most important and common features that contribute to the survival of the instances across all the tasks. The $\ell_{2,1}$ norm on a matrix is commonly used to learn the shared features since it tends to result in similar sparsity patterns for the parameters in all tasks [65]. Here, we extend the $\ell_{2,1}$ norm on the matrix to the $\ell_{F,1}$ norm, defined in Eq. (5.8), on the tensor $\mathbf{B}$, which will lead to shared features across all tasks at all time points.

$$||\mathbf{B}||_{F,1} = \sum_{p=1}^{P} \sqrt{||\mathbf{B}_{p::}||_F^2} \tag{5.8}$$

where $||\mathbf{B}_{p::}||_F^2 = \sum_{k=1}^{K} \sum_{j=1}^{J} \mathbf{B}_{pjk}^2$.

Intra-task temporal smoothness

One of the main objectives of the MTMT algorithm is to dynamically perform the multi-task learning. This is necessary because the survival status of each instance in each task keeps changing over time. By dynamically learning the multi-task problem, we can monitor the survival status for each instance over time, incorporate the updated survival status in the training process and track the most significant features across all the tasks. According to the definition of the indicator tensor $\mathbf{S}$ in Eq. (5.5), there exits a high temporal dependency between the problems at different time points. In this work, an intra-task temporal smoothness regularization term defined in Eq. (5.9) is introduced to reduce the parameter deviations estimated at adjacent time points [153].

$$||\mathbf{B}||_{\text{ts}} = \sum_{k=1}^{K} \sum_{j=1}^{J-1} \|\mathbf{B}_{:(j+1)k} - \mathbf{B}_{:jk}\|_2^2 \tag{5.9}$$

It should be noted that similar to fused lasso [105], Eq. (5.7) and Eq. (5.9) are designed to prevent overfitting and encourage intra-task temporal smoothness, respectively. However, fused lasso introduces sparsity to parameters and differences of parameters for adjacent time points, which is not required in our problem. Therefore, we only consider regularizers with Euclidean norm to decay weights (i.e., model parameters) and parameter differences, instead of introducing sparsity.

After incorporating all the constraints into Eq. (5.4), we can solve the proposed temporal

multi-task learning framework by minimizing the following objective function:

$$\begin{aligned}\min_{\mathbf{B}} \sum_{k=1}^{K}\sum_{j=1}^{J}\left(\frac{1}{2}\|\mathbf{S}_{:jk}\odot(\mathbf{Y}_{:jk}-\mathbf{X}_{::k}\mathbf{B}_{:jk})\|_2^2\right)+\frac{1}{2}\alpha\|\mathbf{B}\|_F^2 \\ +\beta||\mathbf{B}||_{F,1}+\frac{1}{2}\gamma||\mathbf{B}||_{\text{ts}}\end{aligned} \tag{5.10}$$

where α, β and γ are the positive parameters controlling the importance of the overfitting term, inter-task correlation terms and intra-task time smoothness term, respectively.

5.3.3 Optimization

The optimization problem in Eq. (5.10) incorporates four unique constraints into our problem. In this section, we apply the alternating direction method of multipliers (ADMM) method [15] and the proximal methods to optimize it since the solution to this problem is not trivial. First, we introduce a sequence of new constraints $\mathbf{M}_{:jk}=\mathbf{X}_{::k}\mathbf{B}_{:jk}$ for $(k=1,\cdots,K,\ j=1,\cdots,J)$ and update the problem in Eq. (5.10) as follows:

$$\begin{aligned}\min_{\mathbf{B}} \sum_{k=1}^{K}\sum_{j=1}^{J}\left(\frac{1}{2}\|\mathbf{S}_{:jk}\odot(\mathbf{Y}_{:jk}-\mathbf{M}_{:jk})\|_2^2\right) \\ +\frac{1}{2}\alpha\|\mathbf{B}\|_F^2+\beta||\mathbf{B}||_{F,1}+\frac{1}{2}\gamma||\mathbf{B}||_{\text{ts}} \\ \text{s.t. } \mathbf{M}_{:jk}=\mathbf{X}_{::k}\mathbf{B}_{:jk},(k=1,\cdots,K,\ j=1,\cdots,J)\end{aligned} \tag{5.11}$$

By combining the linear and quadratic terms in the augmented Lagrangian and scaling the dual variable, the objective function can be written as:

$$\begin{aligned}\min_{\mathbf{B}} \sum_{k=1}^{K}\sum_{j=1}^{J}\left(\frac{1}{2}\|\mathbf{S}_{:jk}\odot(\mathbf{Y}_{:jk}-\mathbf{M}_{:jk})\|_2^2\right)+\frac{1}{2}\alpha\|\mathbf{B}\|_F^2 \\ +\sum_{k=1}^{K}\sum_{j=1}^{J}\left(\frac{1}{2}\sigma\|\mathbf{M}_{:jk}-\mathbf{X}_{::k}\mathbf{B}_{:jk}+\mathbf{u}_{:jk}\|_2^2-\frac{1}{2}\sigma\|\mathbf{u}_{:jk}\|_2^2\right) \\ +\beta||\mathbf{B}||_{F,1}+\frac{1}{2}\gamma||\mathbf{B}||_{\text{ts}}\end{aligned} \tag{5.12}$$

where $\mathbf{u}$ is the scaled variable tensor and $\sigma>0$ is the penalty parameter. Then we can have the scaled form of ADMM as follows:

$$\begin{aligned}\mathbf{M}^{n+1} := \arg\min_{\mathbf{M}} \sum_{k=1}^{K}\sum_{j=1}^{J}\left(\frac{1}{2}\|\mathbf{S}_{:jk}\odot(\mathbf{Y}_{:jk}-\mathbf{M}_{:jk})\|_2^2\right. \\ \left.+\frac{1}{2}\sigma\|\mathbf{M}_{:jk}-\mathbf{X}_{::k}\mathbf{B}_{:jk}^{n}+\mathbf{u}_{:jk}^{n}\|_2^2\right)\end{aligned} \tag{5.13}$$

$$\begin{aligned}\mathbf{B}^{n+1} := \arg\min_{\mathbf{B}} \sum_{k=1}^{K}\sum_{j=1}^{J} \Big(&\frac{1}{2}\sigma\|\mathbf{M}_{:jk}^{n+1} - \mathbf{X}_{::k}\mathbf{B}_{:jk} + \mathbf{u}_{:jk}^{n}\|_2^2 \\ &+ \frac{1}{2}\alpha\|\mathbf{B}_{:jk}\|_2^2 + \frac{1}{2}\gamma(1-\delta(j=J))||\mathbf{B}_{:(j+1)k} - \mathbf{B}_{:jk}||_2^2\Big) \\ &+ \beta||\mathbf{B}||_{F,1}\end{aligned} \tag{5.14}$$

$$\mathbf{u}_{::k}^{n+1} := \mathbf{u}_{::k}^{n} + \mathbf{M}_{::k}^{n+1} - \mathbf{X}_{::k}\mathbf{B}_{::k}^{n+1} \tag{5.15}$$

Thus, next we need to optimize the two problems given in Eq. (5.13) and Eq. (5.14) in order to solve the problem in Eq. (5.11). Here, the symbol n in the superscript represent the number of iterations.

Step 1: Update $\mathbf{M}^{n+1}$ given $\mathbf{B}^{n}$ and $\mathbf{u}^{n}$ (solve Eq. (5.13)).

The first order and second order derivative of the problem in Eq. (5.13) with respect to $\mathbf{M}_{ijk}$ calculated in Eq. (5.16) and Eq. (5.17) indicates that the objective function in Eq. (5.13) is strictly convex.

$$\frac{\partial L}{\partial \mathbf{M}_{ijk}} = -\mathbf{S}_{ijk}(\mathbf{Y}_{ijk} - \mathbf{M}_{ijk}) + \sigma(\mathbf{M}_{ijk} - \mathbf{X}_{i:k}\mathbf{B}_{:jk}^{n} + \mathbf{u}_{ijk}^{n}) \tag{5.16}$$

$$\frac{\partial^2 L}{(\partial \mathbf{M}_{ijk})^2} = \mathbf{S}_{ijk} + \sigma > 0 \tag{5.17}$$

Thus, $\mathbf{M}_{::k}$ can be updated by Eq. (5.18) in each iteration for each task k.

$$\mathbf{M}_{::k}^{n+1} \leftarrow \frac{\mathbf{S}_{::k} \odot \mathbf{Y}_{::k} + \sigma(\mathbf{X}_{::k}\mathbf{B}_{::k}^{n} - \mathbf{u}_{::k}^{n})}{\mathbf{S}_{::k} + \sigma} \tag{5.18}$$

Step 2: Update $\mathbf{B}^{n+1}$ given $\mathbf{M}^{n+1}$ and $\mathbf{u}^{n}$ (solve Eq. (5.14)).

The objective function in Eq. (5.14) can be separated into two terms, a smooth term and a non-smooth term, as shown in Eq. (5.19).

$$L(\mathbf{B}) = g(\mathbf{B}) + \beta\|\mathbf{B}\|_{F,1} \tag{5.19}$$

where $g(\mathbf{B})$ is a smooth convex function defined as:

$$\begin{aligned}g(\mathbf{B}) := \sum_{k=1}^{K}\sum_{j=1}^{J} \Big(&\frac{1}{2}\sigma\|\mathbf{M}_{:jk} - \mathbf{X}_{::k}\mathbf{B}_{:jk} + \mathbf{u}_{:jk}\|_2^2 \\ &+ \frac{1}{2}\alpha\|\mathbf{B}_{:jk}\|_2^2 + \frac{1}{2}\gamma(1-\delta(j=J))||\mathbf{B}_{:(j+1)k} - \mathbf{B}_{:jk}||_2^2\Big)\end{aligned} \tag{5.20}$$

Algorithm 3: FISTA for $\ell_{F,1}$ Constrained Optimization.

Input: Predictor tensor ($\mathbf{X}$);
Proximal survival status tensor ($\mathbf{M}$);
Scaled variable tensor ($\mathbf{u}$);
Output: Regression coefficient tensor ($\bar{\mathbf{B}}$)

Initialize: $\mathbf{B}^0$ random real numbers, $\mathbf{B}^{-1} = \mathbf{B}^0$;
$m = 1$, $\xi^0 = 1$, $\mu^{-1} = 0$, $\mu^0 = 1$;
repeat
  $\mathbf{A}^m = \mathbf{B}^{m-1} + \frac{\mu^{m-2}-1}{\mu^{m-1}}(\mathbf{B}^{m-1} - \mathbf{B}^{m-2})$;
  repeat
    Calculate $\mathbf{B}^m = \pi_{\mathcal{Z}}(\mathbf{A}^m - \frac{1}{\xi^m} g^{'}(\mathbf{A}^m))$;
    Calculate $\Delta\mathbf{B}^m = \mathbf{B}^m - \mathbf{A}^m$;
    Calculate $h(\mathbf{A}^m, \mathbf{B}^m) = g(\mathbf{A}^m) + \sum_{k,j} \left(\frac{\partial g(\mathbf{B})}{\partial \mathbf{B}_{:jk}}\right)^T \Delta\mathbf{B}^m_{:jk} + \frac{\xi^m}{2}||\Delta\mathbf{B}^m||^2_F$;
    if $g(\boldsymbol{B}^m) \leq h(\boldsymbol{A}^m, \boldsymbol{B}^m)$ **then**
      break ;
    end
    Set $\xi^m = 2\xi^{m-1}$;
  until *Convergence*;
  $\mu^m = \frac{1+\sqrt{1+4\mu^2_{m-1}}}{2}$;
  $m = m + 1$;
until *Convergence*;
$\bar{\mathbf{B}} = \mathbf{B}^m$.

and the first order derivative of $g(\mathbf{B})$ can be obtained as:

$$\begin{aligned}\frac{\partial g(\mathbf{B})}{\partial \mathbf{B}_{:jk}} :&= \sigma \mathbf{X}^T_{::k}(\mathbf{M}_{:jk} - \mathbf{X}_{::k}\mathbf{B}_{:jk} + \mathbf{u}_{:jk}) \\ &+ \alpha \mathbf{B}_{:jk} + \gamma(1 - \delta(j = J))(\mathbf{B}_{:jk} - \mathbf{B}_{:(j+1)k}) \\ &+ \gamma(1 - \delta(j = 1))(\mathbf{B}_{:jk} - \mathbf{B}_{:(j-1)k})\end{aligned} \tag{5.21}$$

Then, the optimization problem for Eq. (5.14) is equivalent to the following $\ell_{F,1}$-ball constrained smooth convex optimization problem.

$$\min_{\mathbf{B} \in \mathcal{Z}} \quad g(\mathbf{B}) \tag{5.22}$$

where $\mathcal{Z} = \{\mathbf{B} \in \mathbb{R}^{P \times J \times K} \ |z \geq \|\mathbf{B}\|_{F,1}\}$ and $z \in \mathbb{Z}_+$ is the radius of the $\ell_{F,1}$-ball. The minimization problem in Eq. (5.22) can be solved by a proximal gradient decent algorithm with backtracking linear search [8], which is based on the fast iterative shrinkage-thresholding

algorithm (FISTA) framework provided in Algorithm 3. Within the iteration m, we set the initial searching point as $\mathbf{A}^m = \mathbf{B}^{m-1} + \eta^m(\mathbf{B}^{m-1} - \mathbf{B}^{m-2})$, where η^m is an non-negative parameter. Then, the approximated solution $\mathbf{B}^m$ can be obtained as:

$$\mathbf{B}^m = \pi_{\mathcal{Z}}(f(\mathbf{A}^m)) \tag{5.23}$$

where $f(\mathbf{A}^m) = \mathbf{A}^m - \frac{1}{\xi}g'(\mathbf{A}^m)$ with parameter ξ, and g' denotes the derivative (See line 6 of Algorithm 3). The Euclidean projection operator [67] $\pi_{\mathcal{Z}}$ is defined as:

$$\begin{aligned} \pi_{\mathcal{Z}}(f(\mathbf{A})) =& \arg\min_{\mathbf{B}} ||\mathbf{B} - f(\mathbf{A})||_F^2 \\ & \text{s.t. } \|\mathbf{B}\|_{F,1} \leq z \end{aligned} \tag{5.24}$$

To solve this $\ell_{F,1}$ constrained optimization problem, we adopt the method introduced in [67], which is originally proposed for $\ell_{2,1}$ constrained optimization problem. In this work, we extend this method to the higher dimension where $\mathbf{B}$ is a tensor.

Theorem 1 *For a given β, the primal optimal point $\boldsymbol{B}$ is given by*

$$\boldsymbol{B}_{p::} = \begin{cases} \frac{||f(\boldsymbol{A}_{p::})||_F - \beta}{||f(\boldsymbol{A}_{p::})||_F} f(\boldsymbol{A}_{p::}) & \beta > 0, ||f(\boldsymbol{A}_{p::})||_F > \beta \\ 0 & \beta > 0, ||f(\boldsymbol{A}_{p::})||_F \leq \beta \\ f(\boldsymbol{A}_{p::}) & \beta = 0 \end{cases} \tag{5.25}$$

Proof: To prove this theorem, we first convert $\mathbf{B}$ to a giant matrix $\underline{B} \in R_+^{P \times Q}$, where $Q = JK$. Each element of $\underline{B}$ is given by $\underline{B}_{pq} = \mathbf{B}_{pjk}$, where $q = k \times J + j$. In the same way, we can create the giant matrices $\underline{M}, \underline{u}$ and $\underline{A}$ for $\mathbf{M}, \mathbf{u}$ and $\mathbf{A}$, respectively. It is easy to prove that the value of $g(\mathbf{B})$ will remain the same by using the giant matrix representation, and there is a giant matrix $\frac{\partial g}{\partial \underline{B}_{:q}}$ corresponding to $\frac{\partial g(\mathbf{B})}{\partial \mathbf{B}_{:jk}}$, since they are both vectors. Therefore, the problem can be solved by the following Euclidean projection

$$\begin{aligned} \underline{B} =& \arg\min_{\underline{B}} ||\underline{B} - (\underline{A} - \frac{1}{\xi}\frac{\partial g}{\partial \underline{B}})||_F^2 \\ & \text{s.t. } \|\underline{B}\|_{2,1} \leq z \end{aligned} \tag{5.26}$$

where $\|\underline{B}\|_{2,1} = \sum_{p=1}^{P} \|\underline{B}\|_2$. It is obvious that $\|\underline{B}\|_{2,1} = \|\mathbf{B}\|_{F,1}$. Finally, comparing Eq. (5.24) with Eq. (5.26), we can conclude that Theorem 1 based on the giant matrix representation is equivalent to Theorem 5 in [67].

The details of the proposed model is shown in Algorithm 4. In the algorithm, $\mathbf{M}$ is initialized to be the survival status tensor $\mathbf{Y}$ and $\mathbf{B}$ is randomly initialized. Within each iteration, $\mathbf{M}$ and $\mathbf{u}$ are updated in lines 3-5 and lines 7-9, respectively. $\mathbf{B}$ is updated in line 6 using Algorithm 3. Algorithm 3 is based on the framework of FISTA algorithm, where the Euclidean projection is conducted in line 6.

Algorithm 4: MTMT Algorithm

Input: Predictor tensor ($\mathbf{X}$); Survival status tensor ($\mathbf{Y}$); Indicator tensor ($\mathbf{S}$); regularization parameters ($\alpha,\ \beta,\ \gamma,\ \sigma$);

Output: Regression coefficient tensor ($\hat{\mathbf{B}}$);

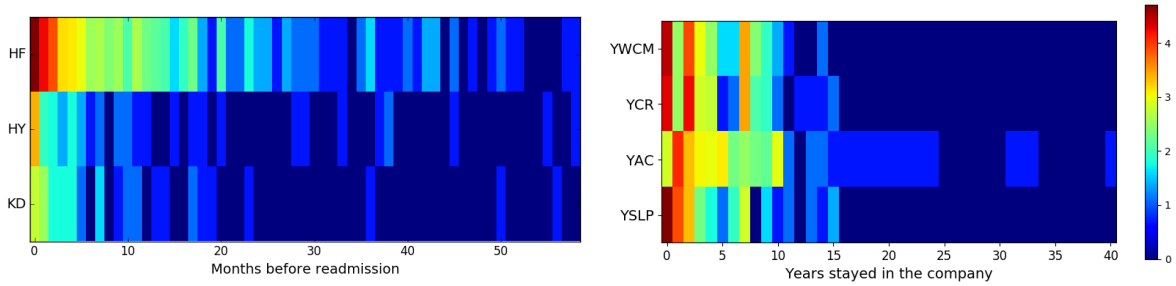
```
Initialize n = 0, M^0 = Y, u^0 = 0, and randomly initialize B^0;
repeat
    for k = 1, K do
        Compute M^{n+1}_{::k} using Eq. (5.18);
    end
    Compute B using Algorithm 1;
    for k = 1, K do
        Compute u^{n+1}_{::k} = u^n_{::k} + M^{n+1}_{::k} - X_{::k} B^{n+1}_{::k};
    end
    n = n + 1;
until Convergence;
B̂ = B^n.
```

5.3.4 Complexity Analysis

For the ADMM algorithm, the time complexity in each iteration is determined by updating $\mathbf{M}$ and $\mathbf{B}$. Firstly, it takes $O(NPJ)$ float point operations to calculate $\mathbf{M}$ using Eq. (5.18), where $N = \max_k(N_k)$. When updating $\mathbf{B}$, within each iteration, the time complexity for evaluating the values of $g(\mathbf{B})$ and its derivative is $O(NPJK)$, and we need to perform $O(PJK)$ operations on the Euclidean projections, i.e., Eq. (5.25). Thus, the time complexity for updating $\mathbf{B}$ is $O(\frac{1}{\sqrt{\epsilon}}(NPJK + PJK)) = O(\frac{1}{\sqrt{\epsilon}}NPJK)$, where ϵ is the desired accuracy. Therefore, the overall time complexity is $O(\frac{1}{\sqrt{\epsilon}}NPJK)$. Since our proposed model trains all the tasks in a single run, it is more convincing to evaluate the average time complexity for each task, which is $O(\frac{1}{\sqrt{\epsilon}}NPJ)$.

5.4 Experimental Results

In this section, we first introduce the two real-world datasets used for the model evaluation, Medical Information Mart for Intensive Care III (MIMIC III) [55, 39] and Employee attrition dataset obtained from IBM Waston Analytics, and then conduct various experiments on the datasets. We compare the performance of the proposed MTMT model with the state-of-the-art baseline methods. In addition, we also analyze the shared features selected by the proposed method and provide the parameter sensitivities.



(a) MIMIC III (b) Employee Attrition

Figure 5.2: Distribution of events over time on both datasets.

5.4.1 Datasets Description

Two real-world datasets will be used in our evaluation.

- **MIMIC III Dataset:** The Medical Information Mart for Intensive Care III (MIMIC III) dataset used in this work is a public dataset provided by PhysioNet[1]. It includes de-identified health-related datasets with more than 40,000 ICU patients from the Beth Isreal Deaconess Medical Center between 2001 and 2012. The information available in this dataset includes: demographics, diagnosis, laboratory test results, procedures and medications, etc. In addition, the dataset shows a good temporal resolution, which is a helpful property for researchers to find the latent relationships between various diseases. In the experiments, we analyze the readmission problems for three diseases, including Heart Failure (HF), Hypertensive (HY) and Kidney Failure (KD). A correlated pattern between the three tasks can be observed in Figure 5.2(a) through the distribution of patient readmission (shown as the logarithm of the number of patients) over time provided.

- **Employee attrition dataset:** The employee attrition dataset provided by IBM Waston Analytics[2] is also used to evaluate the performance of the proposed algorithm. The dataset spans 40 years and consists of 1470 employees attrition information. The features provided in the dataset can be categorized into four types: demographical, education related, job related and salary related information. Based on the attrition status and the time information provided, we formulate four related problems and generate four longitudinal datasets. (i) *Years At Company (YAC)*: How many years have the employee stayed at the company before leaving? (ii) *Years With Current Manager (YWCM)*: How many years have the employee worked with the current manager before leaving? (iii) *Years in Current Role (YCR)*: How many years have the employee worked in the current role? (iv) *Years Since Last Promotion (YSLP)*: How many years have the employee stayed in the company since the last promotion? The distribution of employee attrition (shown as the logarithm

[1]https://mimic.physionet.org/
[2]https://www.ibm.com/communities/analytics/watson-analytics-blog/hr-employee-attrition/

Table 5.1: Details of the dataset used in our experiments.

Dataset		**# instances**	**# features**	**# censored**
MIMIC III	HF	1811	118	1333
	HY	731	118	657
	KD	1209	118	1143
Employee Attrition	YWCM	1207	46	1055
	YCR	1226	46	1062
	YAC	1426	46	1205
	YSLP	889	46	762

of the number of employees) over time provided in Figure 5.2(b) indicates a correlated pattern between the four tasks, especially in the first 15 years.

There are three main steps in data preparation for the two survival problems. (i) We extracted features from the original dataset after generating dummy variables for the categorical features and removing the features with zero-variance. (ii) The longitudinal dataset for each survival task was obtained as the triplet (X, T, Y) by collecting the features, event status and the corresponding event time for all subjects. (iii) The entire dataset of each problem represented by a tensor is obtained by combining the data in individual tasks.

More details about the datasets are provided in Table 5.1. In the table, "# instances", "# features" and "# censoring" indicate the number of instances, features, and censored instances, respectively. For the Employee Attrition problem, the event of interest in the formulated survival problem is employee attrition. Therefore, the censored instances correspond to the employees who are still working in the company at the observation time, while the uncensoring indicates that the employee has left the company during the observation time period. For the MIMIC III dataset, the event of interest is the patient readmission and the censored instances are the patients who are not readmitted to the hospital during the observation period. We implemented the proposed MTMT algorithm in Python on the two datasets using 5-fold cross validation. Our codes are publicly available at `https://github.com/wangpinggl/MTMT`.

5.4.2 Comparison Methods

To evaluate the performance of the proposed algorithm, we compare it with the following commonly used survival analysis methods.

- **Cox-based methods:** The Cox-based models [25] assume that all the instances share the same baseline hazard function. The Lasso-Cox and the EN-Cox method are developed for the high-dimensional survival problems. DeepSurv [58] is a Cox proportional hazards

deep neural network method for modelling the dependencies between covariates and events. In our experiments, Cox model and the regularized Cox models are trained using the *survival* [103] and *fastcox* [132] package in R, respectively. DeepSurv is implemented with Keras[3]. The batch size and epoch are set to be 100 and 30, respectively. The RMSprop optimizer with hyper-parameter $\rho = 0.9$, $\epsilon = 10^{-8}$ and learning rate of 10^{-5} is adopted to train the model parameters. The dimension of the hidden states is set to be 32.

- **Parametric survival methods:** The survival time or the logarithm of the survival time is assumed to follow a theoretical distribution in parametric methods. In the experiments, the parametric survival methods learned through *survival* package with Exponential and Weibull distributions are compared with the proposed method.

- **Linear methods:** We also compare the proposed method with the standard linear regression optimized using ordinary least square (OLS) method since the multi-task framework in this work is based on linear regression method. It should be noted that the standard linear regression method is trained only using the subjects who experienced events since it cannot handle the censored data in longitudinal studies. Tobit model and Buckley-James (BJ) regression method are two linear regression methods adapted to solve survival problems and trained in *survival* and *bujar* package in R, respectively.

- **Multi-task learning methods:** We also compare with the standard multi-task learning method regularized with lasso and $\ell_{2,1}$ implemented in the MALSAR package [152] written in MATLAB. The multi-task learning model for survival analysis (MTLSA) in [65] and its variant MTLSA.V2 which formulate a sequence of standard survival analysis problems over time for the original problem with single event of interest are implemented using the MTLSA package[4].

5.4.3 Performance Evaluation

A common approach to evaluate survival analysis methods is to consider the relative risk between two comparable instances instead of the absolute survival times for each instance using the time-dependent AUC [46], which is also known as concordance index (C-index) for survival analysis problems. Two instances are considered to be comparable if their survival times fall in one of the following: (i) both of them are uncensored; (ii) the observed event time of the uncensored instance is smaller than the censoring time of the censored instance [96]. Two comparable instances are considered to be concordant if the predicted probability of event for the instance which has event earlier is higher than that of the other instance.

Consider two instances i and j, we can use an indicator function to represent the comparability of this pair of instances as $C_{ij} = I(Y_i = 1 \; and \; T_i < T_j)$, which indicates that $C_{ij} = 1$, if the

[3]https://github.com/KITMILTU/DeepSurv-Keras
[4]https://github.com/MLSurvival/MTLSA

Table 5.2: Performance evaluation using time-dependent AUC (along with their standard deviations) on Employee Attrition dataset.

Methods		**YWCM**	**YCR**	**YAC**	**YSLP**	**Average**
Cox-based	Cox	0.8473 (0.0022)	0.8371 (0.0020)	0.8741 (0.0013)	0.7820 (0.0031)	0.8351 (0.0015)
	Lasso-Cox	0.8478 (0.0011)	0.8418 (0.0004)	0.8696 (0.0015)	0.7841 (0.0065)	0.8358 (0.0013)
	EN-Cox	0.8474 (0.0018)	0.8441 (0.0003)	0.8691 (0.0008)	0.7961 (0.0067)	0.8392 (0.0009)
	DeepSurv	0.5335(0.0597)	0.4964 (0.0546)	0.5591 (0.0522)	0.5508 (0.0513)	0.5350(0.0278)
Parametric	Exponential	0.8325 (0.0026)	0.8275 (0.0021)	0.8616 (0.0012)	0.7753 (0.0039)	0.8242 (0.0013)
	Weibull	0.8543 (0.0020)	0.8411 (0.0021)	0.8698 (0.0011)	0.7799 (0.0038)	0.8363 (0.0016)
Linear	OLS	0.7329 (0.0007)	0.7584 (0.0016)	0.8202 (0.0010)	0.6623 (0.0006)	0.7435 (0.0043)
	Tobit	0.8463 (0.0022)	0.8410 (0.0022)	0.8727 (0.0010)	0.7722 (0.0034)	0.8331 (0.0018)
	BJ	0.8494 (0.0000)	0.8312 (0.0006)	0.8636 (0.0033)	0.7760 (0.0045)	0.8301 (0.0015)
Multi-task linear	Multi-Lasso	0.7061 (0.0011)	0.7557 (0.0010)	0.7900 (0.0017)	0.6894 (0.0052)	0.7353 (0.0021)
	Multi-$\ell_{2,1}$	0.7941 (0.0025)	0.8128 (0.0003)	0.8326 (0.0012)	0.7751 (0.0107)	0.8037 (0.0006)
	MTLSA	0.8176 (0.0007)	0.8315 (0.0003)	0.8524 (0.0019)	0.7923 (0.0037)	0.8235 (0.0007)
	MTLSA.V2	0.8327 (0.0011)	0.8432 (0.0006)	0.8649 (0.0022)	0.7972 (0.0042)	0.8345 (0.0008)
	MTMT	**0.8628 (0.0045)**	**0.8455 (0.0009)**	**0.8750 (0.0009)**	**0.8151 (0.0010)**	**0.8496 (0.0005)**

two instances are comparable and $C_{ij} = 0$, otherwise. Thus, the total number of comparable pairs can be obtained as

$$N_C = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} C_{ij} \tag{5.27}$$

where N represents the number of instances in the testing dataset. We can also represent the concordance of the two instances i and j similarly using the indicator function as $c_{ij} = I(s_i < s_j \; and \; C_{ij} = 1)$, where s_i and s_j represent the estimated survival probabilities for instances i and j, respectively. This indicates that $c_{ij} = 1$, if instances i and j are concordant and $c_{ij} = 0$, otherwise. Thus, the total number of concordant pairs can be calculated as

$$N_c = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} c_{ij} \tag{5.28}$$

Therefore, the time-dependent AUC can be calculated as:

$$AUC_t = N_c / N_C \tag{5.29}$$

Table 5.2 and Table 5.3 show the performance results of different algorithms using time-dependent AUC on the Employee Attrition and MIMIC III datasets, respectively. The results in bold represent the best performance. It can be observed that the proposed MTMT

Table 5.3: Performance evaluation using time-dependent AUC (along with their standard deviations) on MIMIC III dataset.

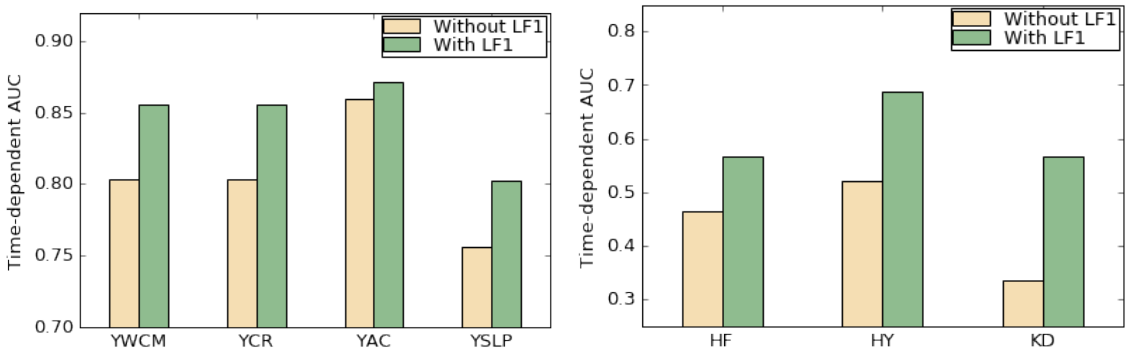
Methods		**HF**	**HY**	**KD**	**Average**
Cox-based	Cox	0.5499 (0.0010)	0.6809 (0.0127)	0.5112 (0.0052)	0.5807 (0.0079)
	Lasso-Cox	0.5534 (0.0004)	0.6764 (0.0110)	0.5395 (0.0081)	0.5898 (0.0057)
	EN-Cox	0.5552 (0.0007)	0.6783 (0.0127)	0.4897 (0.0048)	0.5744 (0.0092)
	DeepSurv	0.5102 (0.0240)	0.5209 (0.0311)	**0.6018 (0.1317)**	0.5464 (0.0537)
Parametric	Exponential	0.5567 (0.0011)	0.6166 (0.0191)	0.5688 (0.0079)	0.5807 (0.0010)
	Weibull	0.5479 (0.0008)	0.5703 (0.0177)	0.5419 (0.0351)	0.5534 (0.0002)
Linear	OLS	0.5354 (0.0007)	0.4671 (0.0286)	0.3181 (0.0366)	0.4402 (0.0123)
	Tobit	0.5578 (0.0011)	0.5946 (0.0029)	0.5531 (0.0198)	0.5685 (0.0005)
	BJ	0.5571 (0.0006)	0.6862 (0.0115)	0.2452 (0.0237)	0.4962 (0.0514)
Multi-task	Multi-Lasso	0.5220 (0.0015)	0.6803 (0.0126)	0.5268 (0.0186)	0.5764 (0.0081)
	Multi-$\ell_{2,1}$	0.5412 (0.0012)	0.6859 (0.0098)	0.5342 (0.0146)	0.5871 (0.0073)
	MTLSA	0.5348 (0.0010)	0.6528 (0.0096)	0.5587 (0.0097)	0.5821 (0.0039)
	MTLSA.V2	0.5404 (0.0010)	0.6630 (0.0116)	0.5633 (0.1228)	0.5889 (0.0042)
	MTMT	**0.5657 (0.0166)**	**0.6868 (0.0007)**	0.5674 (0.0178)	**0.6066 (0.0048)**

algorithm outperforms other state-of-the-art methods for all tasks on Employee Attrition dataset and two tasks on MIMIC III dataset. For other tasks, MTMT also achieves similar time-dependent AUC value compared to the best performance. The last column in Table 5.2 and Table 5.3 is the averaged time-dependent AUC across all the tasks for each algorithm. The results show that the MTMT algorithm outperforms other baseline methods on average for each task on the two problems. These promising results indicate that the proposed multi-task framework can increase the prediction performance of each task on average.

5.4.4 Feature Selection

One of the main goals of multi-task learning methods is to learn the shared features across all the tasks. In this section, we evaluate the effect of the $\ell_{F,1}$ norm on the tensor, which is mainly used to perform the common feature selection across all tasks over time. The bar plots in Figure 5.3(a) and Figure 5.3(b) show the effect of the $\ell_{F,1}$ norm on the prediction performance using time-dependent AUC. We can observe that the MTMT model regularized by $\ell_{F,1}$ norm (in green) outperforms the model without $\ell_{F,1}$ norm (in yellow) on all the tasks. This indicates that the new proposed $\ell_{F,1}$ norm can perform effectively for feature selection.

Table 5.4 and Table 5.5 provide the top-10 common features selected across different tasks on



(a) Employee Attrition (b) MIMIC III

Figure 5.3: Effect of $\ell_{F,1}$ norm on feature selection for both (a) employee attrition and (b) MIMIC III.

Table 5.4: The top-10 common features selected across four tasks on the Employee Attrition dataset.

Feature Name	**Score**
Total Working Years	2.2011
Job Level	0.8613
Performance Rating	0.7617
Job Involvement	0.4067
Number of Companies Worked	0.4012
Over Time	0.3537
Work Life Balance	0.3077
Job Role: Sales Executive	0.2938
Environment Satisfaction	0.2591
Job Role: Sales Representative	0.2508

the Employee Attrition dataset and MIMIC III dataset, respectively, based on the averaged $\ell_{F,1}$ norm over 5-fold of the tensor slices $\mathbf{B}_{i::}$ (as given in Section 5.3) for each feature. The higher score indicates the greater impact on the final prediction.

According to the results provided in Table 5.4, we can observe that the feature *Total Working Years* has the highest impact on the employee attrition. In addition, all these selected features are job related features, which means that these features mainly dominate the employee attrition. It is worth noting that two dummy features about "Job Role" among the selected features belong to the sales department. It indicates that the turnover rate in the sales department is relatively high compared to other departments. The selected common features

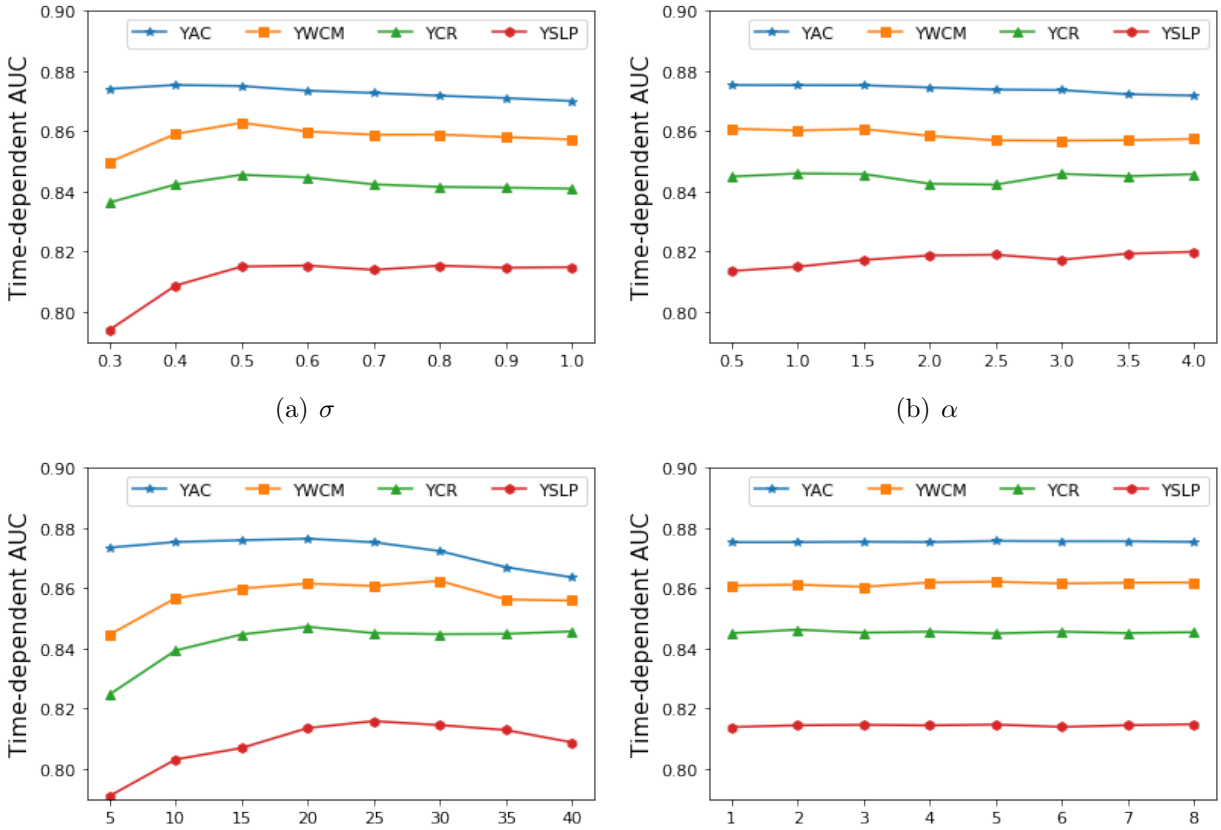
Table 5.5: The top-10 common features selected across three tasks on MIMIC III dataset.

Feature Name	**Score**
Support Systems	11.1720
Non Invasive Blood Pressure Systolic	11.0395
Braden Mobility	11.0392
Urine Appearance	10.9794
Position	10.9227
Creatinine	10.8996
Dorsal PedPulse R	10.8814
Marital status	10.8604
Pain Present	10.8504
Blood Urea Nitrogen	10.8452

among all the problems can provide guidance to reduce the attrition in a company. Based on the results in Table 5.5 on MIMIC III dataset, we can observe that the feature *Support Systems* affects the patient readmission most. It can also be observed that the variance of the scores for the top-10 features on MIMIC III dataset is relatively small, which indicates that these features affect the three tasks at the same level.

5.4.5 Parameter Sensitivity and Convergence Analysis

There are mainly four parameters in the proposed MTMT method, including (i) σ, which is used in the ADMM algorithm, (ii) the weight for the ℓ_F norm α, (iii) the weight of the inter-task correlation β and (iv) the weight for the intra-task time smoothness γ. In this section, we provide the parameter sensitivity for each of these parameters in order to show the robustness of the proposed MTMT algorithm. Figure 5.4 shows the parameter sensitivity of the four parameters in their best performance range based on our experiments on the Employee Attrition dataset. Each of the prediction performances are stable across all the four tasks when varying each parameter. The parameters used in our experiments are $\sigma = 0.5$, $\alpha = 0.5$, $\beta = 25$ and $\gamma = 2$. Figure 5.5 shows the convergence of MTMT model on both datasets. The convergence threshold is set to 0.01. It took around 300 seconds for MTMT model to converge on both datasets.



(a) σ (b) α

(c) β (d) γ

Figure 5.4: Parameter sensitivity of the four parameters used in the proposed method on Employee Attrition dataset.

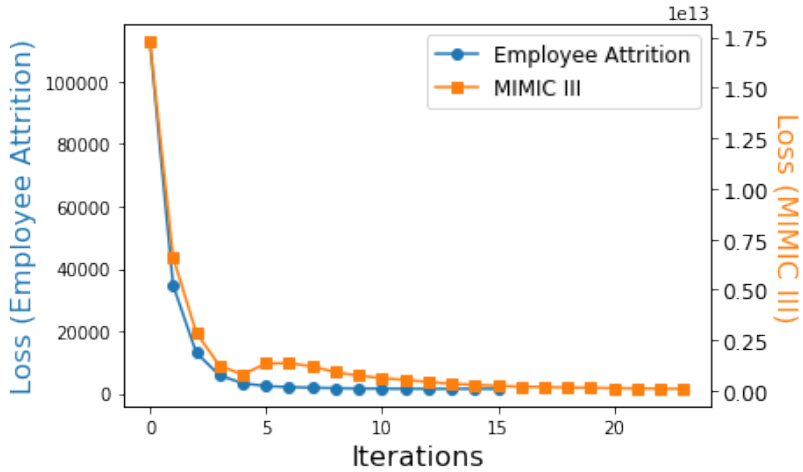


Figure 5.5: Convergence of MTMT model on both datasets.

5.5 Summary

Standard event prediction models are commonly used to make predictions for a single specific event at a given time point instead of predicting the occurrence of multiple events of interest simultaneously in a dynamic setting. To avoid sub-optimal solutions that are obtained by simply applying the standard survival analysis method independently to each task at specific time points, we formulated a temporal multi-task learning framework MTMT for survival analysis and optimized the problem using ADMM method. Our MTMT method demonstrates a superior performance on two real-world datasets compared to other state-of-the-art models. The qualitative results show that the common features selected by MTMT method can provide an important guideline for the real-world applications.

Chapter 6

Conclusions and Future Work

In this chapter, we provide a summary of this dissertation and discuss the future research directions.

6.1 Conclusions

Electronic Health Records (EHR) provides important medical history information of patients. Efficient usage of EHR data can assist doctors with their clinical decision making. This dissertation aims to develop advanced and innovative machine learning methods for automatic question answering and knowledge discovery on EHR. Specifically, we focused on four important research tasks about EHR, including (1) question-to-SQL generation task for question answering on multi-relational structured tables in EHR, (2) knowledge base question answering for retrieving important medical information from unstructured clinical notes, (3) contextual embedding learning for heterogeneous networks to analyze complex relationships of entities in EHR, and (4) temporal multi-task survival analysis to modeling several survival problems jointly and dynamically.

The task of question-to-SQL generation aims to retrieve answers from the structured tables in EHR by predicting the corresponding SQL query of the input question. To address this challenging problem, we first created a large-scale Question-SQL pair dataset, namely MIMICSQL, for the question-to-SQL generation task in the healthcare domain. Further, a TRanslate-Edit Model for Question-to-SQL (TREQS) generation task on MIMICSQL dataset is proposed. Given an input question, TREQS model first generates the output SQL query directly and then edits it by applying an attentive-copying mechanism and a recover technique. We conducted extensive experimental studies and provided both the qualitative and quantitative analysis. The results demonstrated that TREQS model can handle the unique challenges in healthcare and is effective and robust to randomly asked questions.

The task of clinical knowledge base question answering is proposed to tackle the limitations of machine reading comprehension methods in handling complex questions about unstructured clinical notes. We first built a clinical knowledge base ClinicalKB from expert annotated clinical notes, so that it can handle questions on a collection of notes for different patients. Further, we introduced the procedures to create the dataset for knowledge base question answering on ClinicalKB. We also introduced a procedure for generating answer candidate subgraphs from ClinicalKB for given questions and developed an attention-based aspect-level reasoning model for KBQA task. Both quantitative and qualitative analysis is provided to demonstrate its effectiveness in identifying answers and interpreting the results.

The task of contextual embedding learning for heterogeneous networks is studied to analyze the underlying complex relationships of entities. We proposed a self-supervised learning framework SLiCE for subgraph-based contextual representations, which incorporates knowledge contained in both the entire graph and localized arbitrary context subgraphs. Different from many existing methods which primarily aggregate information from either direct neighbors or semantic neighbors connected via certain pre-defined metapaths, SLiCE learns the contextual representations by characterizing the context for a specific task. Extensive experimental studies on both healthcare network and other public network benchmark datasets for the link prediction task shows that SLiCE significantly outperforms several competitive baseline methods.

The task of temporal survival analysis aims to study the survival problems with multiple events of interest simultaneously in a dynamic setting. A temporal multi-task learning framework MTMT is formulated to dynamically follow up the survival status of the instances in each longitudinal study. The proposed MTMT model is optimized using ADMM method. An extensive set of experiments are conducted to show that the proposed MTMT method demonstrates a superior performance on two real-world datasets compared to other state-of-the-art models. It also overcome the limitations of data insufficiency for a single event. The qualitative results show that the common features selected by MTMT method can provide an important guideline for the real-world applications.

6.2 Future Research Directions

6.2.1 Multi-modal Question-Answering Systems

Most existing question answering works in healthcare considers a single data source, such as structured tables or unstructured clinical notes. This limits their ability to handle comprehensive questions due to the differences of information coverage in different data sources. For example, structured tables in EHR tend to include the ICD codes of the diseases, procedures, and medications. While more detailed information about hospital process is usually specified in the clinical notes in the form of free-text. Therefore, we are interested in

building an integrated and interactive Question Answering (QA) system on EHRs that can seek answers jointly from a multi-modal data source, including tables, knowledge graphs and narrative records, for clinical activity related questions. In this case, we can incorporate the answer information returned from different sources into the final answers by taking advantages of different question answering methods. Many research questions arise in this project. For example, what kind of questions can be answered by a multimodal QA system compared with traditional systems? How does the model choose the data source from which the answer can be derived? This research will benefit the development of artificial intelligence-based healthcare dialogue systems.

6.2.2 Unsupervised and Weakly-supervised Learning for Medical Events Extraction

As many clinical notes are collected every day, it is challenging and time consuming for doctors to identify useful information from the large set of free-text clinical notes and assist their decision making in clinical practice. Recently, the task of turning the unstructured clinical notes into structured data and integrating them into a knowledge base about medical events is attracting great attention. However, it is a challenging task to extract useful medical entities along with their complex relations from various narrative medical information. It also requires great efforts from domain experts to collect a large set of human annotations. Therefore, we plan to develop unsupervised and weakly-supervised methods that can extract medical events from clinical notes when limited annotations are available. These methods are more applicable in practice because only a small number of annotations can lead to desired performance. This problem involves several NLP tasks, including name entity recognition, entity linking, and relations extraction (i.e., events extraction). The constructed knowledge bases can benefit many downstream applications, such as QA systems. The proposed methods can also be applied to other domains, such as e-commerce and social media.

6.2.3 Contextual Recommendation in Healthcare

Recently, knowledge graph has gained considerable attention in order to provide personalized recommendation in many open-domain problems, such as social networks and e-commerce, since it is able to incorporate various relations between users, items and their attribute information. We are interested in adopting this strategy to provide medication and treatment recommendation for patients in the healthcare domain. However, there are still several challenges. For example, how to construct a suitable clinical knowledge graph in healthcare domain for medical recommendation? What attribute information can be considered for patients, medication, and treatment? In addition, in order to support for personalized precision medicine, it requires the contextualized recommendation for the medication and treatment in healthcare domain in order to consider various useful information, such as

patients' previous medical history, current health status, and medical information from other related patients. To tackle these challenges, we plan to extend the contextual embedding learning strategy to learn contextual embeddings for entities in the clinical knowledge graph instead of using a single static representation. The contextual embeddings will incorporate both the structure information of the global network and the semantic interactions in the local context.

Bibliography

[1] S. Abu-El-Haija, B. Perozzi, R. Al-Rfou, and A. A. Alemi. Watch your step: Learning node embeddings via graph attention. In *NeurIPS*, 2018.

[2] A. Abujabal, M. Yahya, M. Riedewald, and G. Weikum. Automated template generation for question answering over knowledge graphs. In *Proceedings of the 26th international conference on world wide web*, pages 1191–1200, 2017.

[3] A. M. Alaa and M. van der Schaar. Deep multi-task gaussian processes for survival analysis with competing risks. In *Proceedings of the 30th Conference on Neural Information Processing Systems*, 2017.

[4] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

[5] S. J. Athenikos and H. Han. Biomedical question answering: A survey. *Computer methods and programs in biomedicine*, 99(1):1–24, 2010.

[6] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 4th International Conference on Learning Representations*, 2015.

[7] S. Bandyopadhyay, S. V. Vivek, and M. Murty. Outlier resistant unsupervised deep architectures for attributed network embedding. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, 2020.

[8] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

[9] A. Ben Abacha and P. Zweigenbaum. Medical question answering: translating medical questions into sparql queries. In *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium*, pages 41–50. ACM, 2012.

[10] B. Bogin, J. Berant, and M. Gardner. Representing schema structure with graph neural networks for text-to-sql parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4560–4565, 2019.

[11] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. ACM, 2008.

[12] A. Bordes, S. Chopra, and J. Weston. Question answering with subgraph embeddings. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 615–620, 2014.

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

[14] A. Bordes, J. Weston, and N. Usunier. Open question answering with weakly supervised embedding models. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 165–180. Springer, 2014.

[15] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[16] J. Buckley and I. James. Linear regression with censored data. *Biometrika*, pages 429–436, 1979.

[17] S. Cao, W. Lu, and Q. Xu. Grarep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 2015.

[18] Y. Cen, X. Zou, J. Zhang, H. Yang, J. Zhou, and J. Tang. Representation learning for attributed multiplex heterogeneous network. In *SIGKDD*, 2019.

[19] I. Chami, A. Wolf, D.-C. Juan, F. Sala, S. Ravi, and C. Ré. Low-dimensional hyperbolic knowledge graph embeddings. In *Annual Meeting of the Association for Computational Linguistics*, 2020.

[20] H. Chen, H. Yin, W. Wang, H. Wang, Q. V. H. Nguyen, and X. Li. Pme: projected metric embedding on heterogeneous networks for link prediction. In *SIGKDD*, 2018.

[21] Y. Chen, L. Wu, and M. J. Zaki. Bidirectional attentive memory networks for question answering over knowledge bases. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2913–2923, 2019.

[22] W. Cui, Y. Xiao, H. Wang, Y. Song, S.-w. Hwang, and W. Wang. Kbqa: learning question answering over qa corpora and knowledge bases. In *Proceedings of the VLDB Endowment*, pages 565–576, 2019.

[23] R. Das, A. Neelakantan, D. Belanger, and A. McCallum. Chains of reasoning over entities, relations, and text using recurrent neural networks. In *EACL*, 2017.

[24] V. S. Dave, M. Al Hasan, B. Zhang, and C. K. Reddy. Predicting interval time for reciprocal link creation using survival analysis. *Social Network Analysis and Mining*, 8(1):16, 2018.

[25] C. R. David. Regression models and life tables. *Journal of the Royal Statistical Society*, 34(2):187–220, 1972.

[26] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[27] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pages 4171–4186, 2019.

[28] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *ICML*, volume 32, pages 647–655, 2014.

[29] L. Dong and M. Lapata. Language to logical form with neural attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 33–43, 2016.

[30] L. Dong and M. Lapata. Coarse-to-fine decoding for neural semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 731–742, 2018.

[31] Y. Dong, N. V. Chawla, and A. Swami. metapath2vec: Scalable representation learning for heterogeneous networks. In *23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017.

[32] K. Donnelly. SNOMED-CT: The advanced terminology and coding system for ehealth. *Studies in health technology and informatics*, 121:279–290, 2006.

[33] A. Epasto and B. Perozzi. Is a single embedding enough? learning node representations that capture multiple social contexts. In *WWW*, 2019.

[34] T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6:615–637, 2005.

[35] M. J. Fard, P. Wang, S. Chawla, and C. K. Reddy. A bayesian perspective on early stage event prediction in longitudinal data. *IEEE Transactions on Knowledge and Data Engineering*, 28(12):3126–3139, 2016.

[36] C. Finegan-Dollak, J. K. Kummerfeld, L. Zhang, K. Ramanathan, S. Sadasivam, R. Zhang, and D. Radev. Improving text-to-sql evaluation methodology. pages 351–360. Association for Computational Linguistics, 2018.

[37] S. Fotso. Deep neural networks for survival analysis based on a multi-task framework. *arXiv preprint arXiv:1801.05512*, 2018.

[38] S. Gehrmann, Y. Deng, and A. Rush. Bottom-up abstractive summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 4098–4109, 2018.

[39] A. L. Goldberger, L. A. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley. Physiobank, physiotoolkit, and physionet. *Circulation*, 101(23):e215–e220, 2000.

[40] A. Grover and J. Leskovec. node2vec: Scalable feature learning for networks. In *SIGKDD*, 2016.

[41] J. Guo, Z. Zhan, Y. Gao, Y. Xiao, J.-G. Lou, T. Liu, and D. Zhang. Towards complex text-to-sql in cross-domain database with intermediate representation. In *ACL*, 2019.

[42] T. Guo and H. Gao. Table2answer: Read the database and answer without sql. *arXiv preprint arXiv:1902.04260*, 2019.

[43] W. Hamilton, P. Bajaj, M. Zitnik, D. Jurafsky, and J. Leskovec. Embedding logical queries on knowledge graphs. In *NeurIPS*, 2018.

[44] X. Han, S. Cao, X. Lv, Y. Lin, Z. Liu, M. Sun, and J. Li. Openke: An open toolkit for knowledge embedding. In *Proceedings of EMNLP*, pages 139–144, 2018.

[45] Y. Hao, Y. Zhang, K. Liu, S. He, Z. Liu, H. Wu, and J. Zhao. An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 221–231, 2017.

[46] F. E. Harrell, K. L. Lee, R. M. Califf, D. B. Pryor, and R. A. Rosati. Regression modelling strategies for improved prognostic prediction. *Statistics in medicine*, 3(2):143–152, 1984.

[47] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[48] Y. He, Y. Song, J. Li, C. Ji, J. Peng, and H. Peng. Hetespaceywalk: a heterogeneous spacey random walk for heterogeneous information network embedding. In *CIKM*, 2019.

[49] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[50] J. Hoffart, F. M. Suchanek, K. Berberich, E. Lewis-Kelham, G. De Melo, and G. Weikum. Yago2: exploring and querying world knowledge in time, space, context, and many languages. In *Proceedings of the 20th international conference companion on World wide web*, pages 229–232, 2011.

[51] Z. Hu, Y. Dong, K. Wang, and Y. Sun. Heterogeneous graph transformer. In *Proceedings of The Web Conference 2020*, pages 2704–2710, 2020.

[52] S. Iyer, I. Konstas, A. Cheung, J. Krishnamurthy, and L. Zettlemoyer. Learning a neural semantic parser from user feedback. In *Proceedings of the 55th Annual Meeting of the ACL*, pages 963–973, 2017.

[53] Q. Jin, B. Dhingra, Z. Liu, W. Cohen, and X. Lu. Pubmedqa: A dataset for biomedical research question answering. 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)*, pages 2567–2577, 2019.

[54] Q. Jin, Z. Yuan, G. Xiong, Q. Yu, C. Tan, M. Chen, S. Huang, X. Liu, and S. Yu. Biomedical question answering: A comprehensive review. *arXiv preprint arXiv:2102.05281*, 2021.

[55] A. E. Johnson, T. J. Pollard, L. Shen, L.-w. H. Lehman, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. A. Celi, and R. G. Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3, 2016.

[56] K. Kafle, B. Price, S. Cohen, and C. Kanan. Dvqa: Understanding data visualizations via question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5648–5656, 2018.

[57] E. L. Kaplan and P. Meier. Nonparametric estimation from incomplete observations. *Journal of the American statistical association*, 53(282):457–481, 1958.

[58] J. L. Katzman, U. Shaham, A. Cloninger, J. Bates, T. Jiang, and Y. Kluger. Deepsurv: personalized treatment recommender system using a cox proportional hazards deep neural network. *BMC medical research methodology*, 18(1):24, 2018.

[59] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2015.

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

[61] T. Kwiatkowski, E. Choi, Y. Artzi, and L. Zettlemoyer. Scaling semantic parsers with on-the-fly ontology matching. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1545–1556, 2013.

[62] E. T. Lee and J. Wang. *Statistical methods for survival data analysis*, volume 476. John Wiley & Sons, 2003.

[63] M. Lee, J. Cimino, H. R. Zhu, C. Sable, V. Shanker, J. Ely, and H. Yu. Beyond information retrieval—medical question answering. In *AMIA annual symposium proceedings*, volume 2006, page 469. American Medical Informatics Association, 2006.

[64] Y. Li, V. Rakesh, and C. K. Reddy. Project success prediction in crowdfunding environments. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, pages 247–256. ACM, 2016.

[65] Y. Li, J. Wang, J. Ye, and C. K. Reddy. A multi-task learning formulation for survival analysis. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1715–1724. ACM, 2016.

[66] C.-Y. Lin. ROUGE: A package for automatic evaluation of summaries. *Text Summarization Branches Out*, 2004.

[67] J. Liu, S. Ji, and J. Ye. Multi-task feature learning via efficient l 2, 1-norm minimization. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*, pages 339–348, 2009.

[68] N. Liu, Q. Tan, Y. Li, H. Yang, J. Zhou, and X. Hu. Is a single vector enough? exploring node polysemy for network embedding. In *SIGKDD*, 2019.

[69] D. Lukovnikov, N. Chakraborty, J. Lehmann, and A. Fischer. Translating natural language to sql using pointer-generator networks and how decoding order matters. *arXiv preprint arXiv:1811.05303*, 2018.

[70] Y. Luo, Q. Wang, B. Wang, and L. Guo. Context-dependent knowledge graph embedding. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1656–1661, 2015.

[71] T. Luong, H. Pham, and C. D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, 2015.

[72] M. R. Lyu. *Handbook of software reliability engineering*, volume 222. IEEE computer society press CA, 1996.

[73] J. Ma, P. Cui, K. Kuang, X. Wang, and W. Zhu. Disentangled graph convolutional networks. In *International Conference on Machine Learning*, 2019.

[74] B. McCann, N. S. Keskar, C. Xiong, and R. Socher. The natural language decathlon: Multitask learning as question answering. *arXiv preprint arXiv:1806.08730*, 2018.

[75] R. Nallapati, B. Zhou, C. dos Santos, Ç. glar Gulçehre, and B. Xiang. Abstractive text summarization using sequence-to-sequence RNNs and beyond. *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, 2016.

[76] A. Pampari, P. Raghavan, J. Liang, and J. Peng. emrqa: A large corpus for question answering on electronic medical records. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2357–2368, 2018.

[77] C. Park, C. Yang, Q. Zhu, D. Kim, H. Yu, and J. Han. Unsupervised differentiable multi-aspect network embedding. In *SIGKDD*, 2020.

[78] P. Pasupat and P. Liang. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, pages 1470–1480, 2015.

[79] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. *Proceedings of the 31st Conference on Neural Information Processing Systems*, pages 1–43, 2017.

[80] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing*, pages 1532–1543, 2014.

[81] B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk: Online learning of social representations. In *SIGKDD*, 2014.

[82] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep contextualized word representations. In *Proceedings of NAACL-HLT*, pages 2227–2237, 2018.

[83] J. Qiu, Y. Dong, H. Ma, J. Li, K. Wang, and J. Tang. Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In *WSDM*, 2018.

[84] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, 2016.

[85] R. Ranganath, A. Perotte, N. Elhadad, and D. Blei. Deep survival analysis. *Proceedings of Machine Learning for Healthcare*, 2016.

[86] C. K. Reddy and Y. Li. A review of clinical prediction models. *Healthcare Data Analytics*, 36:343–378, 2015.

[87] K. Roberts and B. G. Patra. A semantic parsing method for mapping clinical questions to logical forms. In *AMIA Annual Symposium Proceedings*, volume 2017, pages 1478–1487. American Medical Informatics Association, 2017.

[88] M. Rotmensch, Y. Halpern, A. Tlimat, S. Horng, and D. Sontag. Learning a health knowledge graph from electronic medical records. *Scientific reports*, 7(1):1–11, 2017.

[89] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, and M. Welling. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pages 593–607. Springer, 2018.

[90] A. See, P. J. Liu, and C. D. Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1073–1083, 2017.

[91] T. Shi, Y. Keneshloo, N. Ramakrishnan, and C. K. Reddy. Neural abstractive text summarization with sequence-to-sequence models. *arXiv preprint arXiv:1812.02303*, 2018.

[92] T. Shi, K. Tatwawadi, K. Chakrabarti, Y. Mao, O. Polozov, and W. Chen. Incsql: Training incremental text-to-sql parsers with non-deterministic oracles. *arXiv preprint arXiv:1809.05054*, 2018.

[93] T. Shi, P. Wang, and C. K. Reddy. Leafnats: An open-source toolkit and live demo system for neural abstractive text summarization. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 66–71, 2019.

[94] N. Simon, J. Friedman, T. Hastie, R. Tibshirani, et al. Regularization paths for cox's proportional hazards model via coordinate descent. *Journal of statistical software*, 39(5):1–13, 2011.

[95] R. Socher, D. Chen, C. D. Manning, and A. Ng. Reasoning with neural tensor networks for knowledge base completion. In *NeurIPS*, 2013.

[96] H. Steck, B. Krishnapuram, C. Dehing-oberije, P. Lambin, and V. C. Raykar. On ranking in survival analysis: Bounds on the concordance index. In *Advances in neural information processing systems*, pages 1209–1216, 2008.

[97] A. Stubbs, C. Kotfila, H. Xu, and Ö. Uzuner. Identifying risk factors for heart disease over time: Overview of 2014 i2b2/uthealth shared task track 2. *Journal of biomedical informatics*, 58:S67–S77, 2015.

[98] F.-Y. Sun, M. Qu, J. Hoffmann, C.-W. Huang, and J. Tang. vgraph: A generative model for joint community detection and node representation learning. In *NeurIPS*, 2019.

[99] H. Sun, H. Ma, X. He, W.-t. Yih, Y. Su, and X. Yan. Table cell search for question answering. In *Proceedings of the 25th International Conference on World Wide Web*, pages 771–782. International World Wide Web Conferences Steering Committee, 2016.

[100] W. Sun, A. Rumshisky, and O. Uzuner. Evaluating temporal relations in clinical text: 2012 i2b2 challenge. *Journal of the American Medical Informatics Association*, 20(5):806–813, 2013.

[101] S. Suster and W. Daelemans. Clicr: a dataset of clinical case reports for machine reading comprehension. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1551–1563, 2018.

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

[103] T. Therneau. A package for survival analysis in s. R package version 2.37-4. 2013.

[104] R. Tibshirani. The lasso method for variable selection in the cox model. *Statistics in Medicine*, 16(4):385–395, 1997.

[105] R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(1):91–108, 2005.

[106] J. Tobin. Estimation of relationships for limited dependent variables. *Econometrica: journal of the Econometric Society*, 26(1):24–36, 1958.

[107] G. Tsatsaronis, G. Balikas, P. Malakasiotis, I. Partalas, M. Zschunke, M. R. Alvers, D. Weissenborn, A. Krithara, S. Petridis, D. Polychronopoulos, et al. An overview of the bioasq large-scale biomedical semantic indexing and question answering competition. *BMC bioinformatics*, 16(1):138, 2015.

[108] Ö. Uzuner. Recognizing obesity and comorbidities in sparse data. *Journal of the American Medical Informatics Association*, 16(4):561–570, 2009.

[109] O. Uzuner, A. Bodnari, S. Shen, T. Forbush, J. Pestian, and B. R. South. Evaluating the state of the art in coreference resolution for electronic medical records. *Journal of the American Medical Informatics Association*, 19(5):786–791, 2012.

[110] Ö. Uzuner, I. Goldstein, Y. Luo, and I. Kohane. Identifying patient smoking status from medical discharge records. *Journal of the American Medical Informatics Association*, 15(1):14–24, 2008.

[111] Ö. Uzuner, I. Solti, and E. Cadag. Extracting medication information from clinical text. *Journal of the American Medical Informatics Association*, 17(5):514–518, 2010.

[112] Ö. Uzuner, I. Solti, F. Xia, and E. Cadag. Community annotation experiment for ground truth generation for the i2b2 medication challenge. *Journal of the American Medical Informatics Association*, 17(5):519–523, 2010.

[113] Ö. Uzuner, B. R. South, S. Shen, and S. L. DuVall. 2010 i2b2/va challenge on concepts, assertions, and relations in clinical text. *Journal of the American Medical Informatics Association*, 18(5):552–556, 2011.

[114] S. Vashishth, S. Sanyal, V. Nitin, and P. Talukdar. Composition-based multi-relational graph convolutional networks. In *International Conference on Learning Representations*, 2020.

[115] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[116] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. 2017.

[117] O. Vinyals, M. Fortunato, and N. Jaitly. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700, 2015.

[118] B. Vinzamuri and C. K. Reddy. Cox regression with correlation based regularization for electronic health records. In *2013 IEEE International Conference on Data Mining (ICDM)*, pages 757–766. IEEE, 2013.

[119] C. Wang, M. Brockschmidt, and R. Singh. Pointing out sql queries from text. *Technical report.*, 2018.

[120] C. Wang, K. Tatwawadi, M. Brockschmidt, P.-S. Huang, Y. Mao, O. Polozov, and R. Singh. Robust text-to-sql generation with execution-guided decoding. *arXiv preprint arXiv:1807.03100*, 2018.

[121] H. Wang, T. Xu, Q. Liu, D. Lian, E. Chen, D. Du, H. Wu, and W. Su. Mcne: An end-to-end framework for learning multiple conditional network representations of social network. In *SIGKDD*, 2019.

[122] L. Wang, Y. Li, J. Zhou, D. Zhu, and J. Ye. Multi-task survival analysis. In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 485–494. IEEE, 2017.

[123] P. Wang, Y. Li, and C. K. Reddy. Machine learning for survival analysis: A survey. *ACM Computing Surveys (CSUR)*, 51(6):110, 2019.

[124] P. Wang, K. K. Padthe, B. Vinzamuri, and C. K. Reddy. Crisp: Consensus regularized selection based prediction. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 1019–1028. ACM, 2016.

[125] P. Wang, T. Shi, and C. K. Reddy. Text-to-sql generation for question answering on electronic medical records. In *Proceedings of The Web Conference 2020*, pages 350–361, 2020.

[126] P. Wang, T. Shi, and C. K. Reddy. A novel tensor-based temporal multi-task survival analysis model. *IEEE Transactions on Knowledge and Data Engineering*, 33(9):3311–3322, 2021.

[127] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, and P. S. Yu. Heterogeneous graph attention network. In *WWW*, 2019.

[128] L.-J. Wei. The accelerated failure time model: a useful alternative to the cox regression model in survival analysis. *Statistics in medicine*, 11(14-15):1871–1879, 1992.

[129] X. Xu, C. Liu, and D. Song. Sqlnet: Generating structured queries from natural language without reinforcement learning. *arXiv preprint arXiv:1711.04436*, 2017.

[130] N. Yaghmazadeh, Y. Wang, I. Dillig, and T. Dillig. Sqlizer: query synthesis from natural language. *Proceedings of the ACM on Programming Languages*, 1, OOPSLA:1–26, 2017.

[131] L. Yang, Y. Guo, and X. Cao. Multi-facet network embedding: Beyond the general solution of detection and representation. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[132] Y. Yang and H. Zou. A cocktail algorithm for solving the elastic net penalized cox's regression in high dimensions. *Statistics and its Interface*, 6(2):167–173, 2012.

[133] X. Yao and B. Van Durme. Information extraction over structured data: Question answering with freebase. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 956–966, 2014.

[134] S. Yavuz, I. Gur, Y. Su, and X. Yan. What it takes to achieve 100 percent condition accuracy on wikisql. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1702–1711, 2018.

[135] S. W.-t. Yih, M.-W. Chang, X. He, and J. Gao. Semantic parsing via staged query graph generation: Question answering with knowledge base. *Proceedings of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 1321–1331, 2015.

[136] P. Yin, Z. Lu, H. Li, and B. Kao. Neural enquirer: Learning to query tables with natural language. *Proceedings of 2016 NAACL Human-Computer Question Answering Workshop*, pages 29–35, 2016.

[137] H. Yu, M. Lee, D. Kaufman, J. Ely, J. A. Osheroff, G. Hripcsak, and J. Cimino. Development, implementation, and a cognitive evaluation of a definitional question answering system for physicians. *Journal of biomedical informatics*, 40(3):236–251, 2007.

[138] T. Yu, Z. Li, Z. Zhang, R. Zhang, and D. Radev. Typesql: Knowledge-based type-aware neural text-to-sql generation. In *Proceedings of NAACL-HLT 2018*, pages 588–594, 2018.

[139] T. Yu, M. Yasunaga, K. Yang, R. Zhang, D. Wang, Z. Li, and D. Radev. Syntaxsqlnet: Syntax tree networks for complex and cross-domaintext-to-sql task. In *EMNLP*, 2018.

[140] T. Yu, R. Zhang, K. Yang, M. Yasunaga, D. Wang, Z. Li, J. Ma, I. Li, Q. Yao, S. Roman, et al. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921, 2018.

[141] X. Yue, B. J. Gutierrez, and H. Sun. Clinical reading comprehension: A thorough analysis of the emrqa dataset. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4474–4486, 2020.

[142] S. Yun, M. Jeong, R. Kim, J. Kang, and H. J. Kim. Graph transformer networks. In *NeurIPS*, 2019.

[143] V. A. Zeithaml, K. N. Lemon, and R. T. Rust. *Driving customer equity: How customer lifetime value is reshaping corporate strategy.* Simon and Schuster, 2001.

[144] C. Zhang, D. Song, C. Huang, A. Swami, and N. V. Chawla. Heterogeneous graph neural network. In *SIGKDD*, 2019.

[145] M. Zhang and Y. Chen. Link prediction based on graph neural networks. In *Advances in Neural Information Processing Systems*, pages 5165–5175, 2018.

[146] R. Zhang, T. Yu, H. Er, S. Shim, E. Xue, X. V. Lin, T. Shi, C. Xiong, R. Socher, and D. Radev. Editing-based sql query generation for cross-domain context-dependent questions. 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)*, pages 5341–5352, 2019.

[147] R. Zhang, Y. Zou, and J. Ma. Hyper-sagnn: a self-attention based graph neural network for hypergraphs. In *ICLR*, 2020.

[148] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja. Robust visual tracking via multi-task sparse learning. In *IEEE conference on Computer vision and pattern recognition (CVPR)*, pages 2042–2049, 2012.

[149] W. Zhang, Y. Fang, Z. Liu, M. Wu, and X. Zhang. mg2vec: Learning relationship-preserving heterogeneous graph representations via metagraph embedding. *IEEE TKDE*, 2020.

[150] Y. Zhang, H. Dai, Z. Kozareva, A. J. Smola, and L. Song. Variational reasoning for question answering with knowledge graph. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[151] V. Zhong, C. Xiong, and R. Socher. Seq2sql: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103*, 2017.

[152] J. Zhou, J. Chen, and J. Ye. *MALSAR: Multi-Task Learning via StructurAl Regularization*. Arizona State University, 2011.

[153] J. Zhou, L. Yuan, J. Liu, and J. Ye. A multi-task learning formulation for predicting disease progression. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 814–822. ACM, 2011.