COMMONSENSE KNOWLEDGE REPRESENTATION AND REASONING IN STATISTICAL SCRIPT LEARNING

by

I-Ta Lee

A Dissertation

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the degree of

Doctor of Philosophy

Computer Science

West Lafayette, Indiana

December 2020

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

ACKNOWLEDGMENTS

I have read so many acknowledgements, but this is the first time I get a chance to write my own one. Countless grateful sentences emerge in my head. I work so hard to make them shorter.

First, I would like to thank my advisor, Dan Goldwasser, for many inspiring and insightful conversations during the past 5 years. Me and my advisor are very different types of persons. Often see things differently. I consider things in a more practical and engineering way, while he sees things with a creative mind. Like most research works, often things do not work out at first. However, sometimes, if one idea works out, it becomes the most shining sparkle, an aggregation of ideas coming from different minds.

Thanks to my advisory committee members, Chris Clifton, Jennifer Neville, and Ninghui Li for offering wise advice and holding this work to a high standard. Thanks to Buster Dunsmore for guiding me to become a good teaching assistant and teacher. Thanks to my former advisors, Lun-Wei Ku, my advisor at Academia Sinica, Taiwan, who brought me into the NLP world; Shun-Ren Yang, my Master's advisor at National Tsing Hua University, Taiwan, who taught me general research methods and encouraged me to pursue my Ph. D.; Tein-Yaw Chung, Chin-Feng Fan, and Rung-Bin Lin, faculty members at Yuan Ze University, who were mentors of my professional career and prepared me solid knowledge in computer science.

Thanks to current and former PurdueNLP lab members: Maria Pacheco, Xiao Zhang, Chang Li, Kristen Johnson, Rajkumar Pujari, Shamik Roy, Nikhil Mehta, Maryam Davoodi, Younghun Lee, Tunazzina Islam, Abhishek Sharma, Prerit Gupta, and Mahak Goindani. I really appreciate the intellectual discussions and debates I had with you all. Especially thanks to Maria Pacheco. I still remember the orientation we attended when we started our Ph. D., and I will always remember every deadline we survived in these years. It was a pity that we could not have parties or hang out during my last semester, because of the COVID-19, but I look forward to meeting you around the world in the future. If you visit where I am, please do not hesitate to contact me.

Thanks to my father, Li-Te Lee, for taking care of the whole family while I am abroad. Thanks to my grandmother for being supportive for what I want to pursue, even though she never wants me to leave home. If I can make a wish when I completed my Ph. D., I hope you two staying healthy and happy forever. I also want to thank my close friend, Chia-Yen Chang. Without your encouragements, I would not be able to make the tough decision for studying abroad at my 28. If you never appeared in my life, I will not be the person who I am now. I wish you the best in all that you do.

Lastly, I want to thank myself for being so brave to make the decisions of leaving the country I lived for almost 30 years, jumping out of the comfort zone, and pursuing an interesting life. Thanks to myself for working so hard and not giving up when facing any frustrations or failures. When I look back, this journey is the adventure that I “failed” the most but rewarded the most. When I was a college student, I never thought I will pursue a Ph. D.. Life leads me here, and I am grateful for acquiring this invaluable experience. This Ph. D. will be the treasure of my career, gaining me confidence to attempt, to risk something, and to take challenges beyond my ability. Eventually, I hope all these can be used in helping people in need.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

A recent surge of research on commonsense knowledge has given the AI community new opportunities and challenges. Many studies focus on constructing commonsense knowledge representations from natural language data. However, how to learn such representations from large-scale text data is still an open question. This thesis addresses the problem through statistical script learning, which learns event representations from stereotypical event relationships using weak supervision. These event representations serve as an abundant source of commonsense knowledge to be applied in downstream language tasks. We propose three script learning models that generalize previous works with new insight. A feature-enriched model characterizes fine-grained and entity-based event properties to address specific semantics. A multi-relational model generalizes traditional script learning models which rely on one type of event relationship—co-occurrence—to a multi-relational model that considers typed event relationships, going beyond simple event similarities. A narrative graph model leverages a narrative graph to inform an event with a grounded situation to maintain a global consistency of event states. Also, pretrained language models such as BERT are used to further improve event semantics.

Our three script learning models do not rely on annotated datasets, as the cost of creating these at large scales is unreasonable. Based on weak supervision, we extract events from large collections of textual data. Although noisy, the learned event representations carry profound commonsense information, enhancing performance in downstream language tasks.

We evaluate their performance with various intrinsic and extrinsic evaluations. In the intrinsic evaluations, although the three models are evaluated in terms of various aspects, the shared core task is Multiple Choice Narrative Cloze (MCNC) [1], [2], which measures the model's ability to predict what happens next, out of five candidate events, in a given situation. This task facilitates fair comparisons between script learning models for commonsense inference. The three models were proposed in three consecutive years, from 2018 to 2020, each outperforming the previous year's model as well as the competitors' baselines. Our best model outperforms EventComp [2], a widely recognized baseline, by a large margin in MCNC: i.e., absolute accuracy improvements of 9.73% (53.86% $\rightarrow$ 63.59%). In the extrinsic

evaluations, we use our models for implicit discourse sense classification (IDSC), a challenging task in which two argument spans are annotated with an implicit discourse sense; the task is to predict the sense type, which requires a deep understanding of common sense between discourse arguments. Moreover, in an additional work we touch on a more interesting group of tasks about psychological commonsense reasoning. Solving these requires reasoning about and understanding human mental states such as motivation, emotion, and desire. Our best model, an enhancement of the narrative graph model, combines the advantages of the above three works to address entity-based features, typed event relationships, and grounded context in one model. The model successfully captures the context in which events appear and interactions between characters' mental states, outperforming previous works.

The main contributions of this thesis are as follows:

- We identify the importance of entity-based features for representing commonsense knowledge with script learning.

- We create one of the first, if not the first, script learning models that addresses the multi-relational nature between events.

- We publicly release contextualized event representations (models) trained on large-scale newswire data.

- We develop a script learning model that combines entity-based features, typed event relationships, and grounded context in one model, and show that it is a good fit for modeling psychological common sense.

To conclude, this thesis presents an in-depth exploration of statistical script learning, enhancing existing models with new insight. Our experimental results show that models informed with the new knowledge aspects significantly outperform previous works in both intrinsic and extrinsic evaluations.

1. INTRODUCTION

"Common sense in an uncommon degree is what the world calls wisdom."

—Samuel Taylor Coleridge (1772–1834)

This thesis is about advancing statistical script learning for representing commonsense knowledge. We present three novel techniques for enhancing event representation learning. *Feature-enriched event representation* characterizes entity-based properties; *multi-relational script learning* diversifies inter-event relationships; and *narrative graph contextualization* strengthens the global consistency of the learned representations. These methods augment innovative aspects, including entity-based features, typed event relationships, and graphical context, to learn event representations. The resulting representations yield significant improvements over previously dominant approaches in intrinsic evaluations, and the improvements can be transferred to downstream tasks or applications that require commonsense knowledge to solve. One additional work that combines the three techniques illustrates a good fit for modeling psychological common sense. More importantly, these methods are all based on a weakly supervised setting, which requires only automatically acquired noisy labels. There is thus no need for expensive human labeling efforts, and the methods can be adapted to various corpora or languages.

The experiments presented in this thesis constitute thorough studies of representation quality for script learning. Our goal is to answer four key questions:

1. *Are noisily extracted events and properties useful for representing events?*
2. *Can we acquire different types of event relationships at a large scale and how do we model them?*
3. *Can we contextualize event representations, just like modern language models contextualize words, to improve the representations?*
4. *What are good applications for script learning models with the above properties?*

The empirical studies, including intrinsic and extrinsic evaluations, will be described in detail in later chapters.

In this chapter, to set the stage for defining our formal problems and novel methods, we introduce the historical and technical background in the context of cognitive science and natural language processing (NLP). This background ensures that as we share these exciting breakthroughs, readers are on the same page as we are. We then summarize the goals of this thesis to shape the three main technical contributions. Lastly, we present the road map of this thesis.

1.1 Human Cognitive System and Commonsense Knowledge

Daniel Kahneman, the Nobel Prize winning psychologist and economist, presented his findings of the human cognitive system in his Nobel Prize lecture "Maps of Bounded Rationality" [3]. Figure 1.1 shows three cognitive systems: *perception*, *intuition*, and *reasoning*.

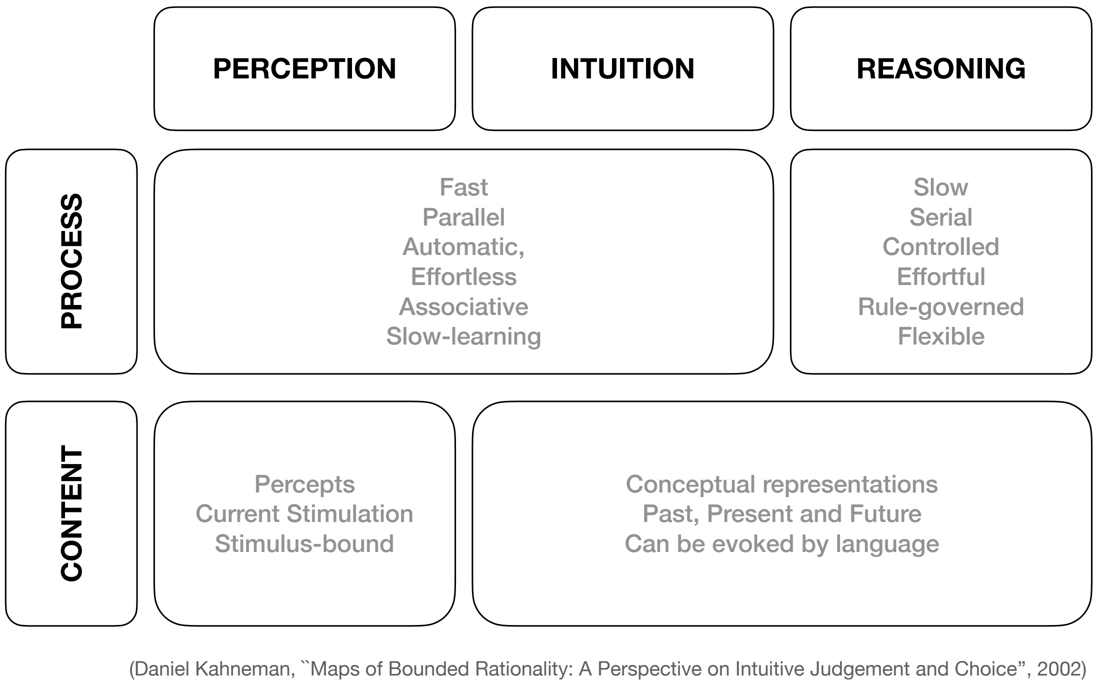


Figure 1.1. Kahneman's three cognitive functions

The operations of *perception* assist us in observing the world and in generating *impressions*, a type of knowledge representation. Example operations are reading articles (text), watching television (image), and listening to podcasts (sound). The operations of *intuition* involve intuitive inferences that are fast and effortless, based on impressions; these happen every moment of a human's life. Example operations are understanding another person's emotions, deducing what happens before or after certain events, and reasoning about motivations and intentions. *Reasoning*, in turn, involves deliberate judgments, which are slow and effortful, and only happen when humans intentionally make them. Example operations are writing PhD theses, solving puzzles, and programming. A related concept used to differentiate intuitive inferences and deliberate reasoning is *accessibility*: the ease with which particular mental contents come to mind [4]. Intuitive thoughts come to mind rapidly and spontaneously, whereas deliberate reasoning is characterized by poorer accessibility.

Under this widely accepted human cognitive system, some researchers suspect there is an information system behind this that maintains knowledge representations. Humans learn knowledge representations through perception, and use them in intuition and reasoning to help make judgments. Knowledge representations have been a core concept of artificial intelligence (AI), but what is a knowledge representation exactly? Davis et al. [5] present a thorough analysis of the concept. A knowledge representation plays different roles in different situations, but essentially it is a surrogate of world knowledge in our mind that enables intelligent reasoning. It is not a pure data structure, as it carries semantics.

The core part of knowledge representations surrogates *commonsense knowledge*—a fundamental level of knowledge that is likely to be shared by all human beings, concerning everyday situations and events. Commonsense knowledge is something that a six-year-old understands naturally without special training. An example of such a description is "You must be awake to eat." We observe that a great deal of intuitive reasoning belongs to commonsense reasoning; deliberate reasoning also requires common sense as background knowledge. Hence, common sense is essential for AI to understand human needs. Representing commonsense knowledge was thus a highly active AI topic in the early '80s. Due to the weak computational power and unresourceful data of the time, the results were inconclusive. The recent revival of commonsense research can be attributed to advanced neural-based models, which construct commonsense representations by perceiving various mediums, such as photos, videos, and text. In this thesis, we focus on leveraging NLP techniques, as we believe languages, as an abundant source of semantic information, are key to advancing commonsense reasoning.

1.2 Commonsense Knowledge Representations in NLP

In NLP, there is a rich history of literature on commonsense knowledge representation for language understanding. In the early '70s, AI researchers began to explore ways to represent commonsense knowledge. Some built knowledge bases for information retrieval, and some extracted knowledge from data. We summarize these into three broad categories: ontology systems, semantic parsing, and script learning.

The purpose of *ontology systems* is to build a comprehensive ontology for all possible concepts existing in human daily life. The Cyc project [6] is a famous, although controversial, example. The goal of the project is to compile millions of language-like descriptions that can be used to compose human common sense. "Mothers are older than their children", for example, is common sense, because even a six-year-old can naturally understand it without special training. Unexpectedly, the Cyc and many other similar ontology-based systems turned out to be building "expert" systems that could relate user queries to relevant descriptions, but could not learn the knowledge that most six-year-olds can understand. Another problem with such expert systems is that creating an ontology that can answer any specific questions, even for small domains, is unreasonably labor-intensive and expensive, raising questions about its worth.

Another way to represent commonsense knowledge is through *semantic parsing*, which converts a natural language span into a machine-understandable representation. FrameNet [7] is an early attempt, approaching knowledge representations using semantic units called *frames*. FrameNet contains 1200 frames, each of which describes a scenario with its core semantic elements. For example, the "commerce sell" frame has *buyers*, *goods*, and *sellers* as its core frame elements. FrameNet has inspired researchers to automate frame parsing on texts with semantic role labeling (SRL), which has attracted much attention in recent years. Another successful attempt of semantic parsing is abstract meaning representation (AMR) [8], a graph-based representation for sentences, which abstracts away grammatical details such that different sentences with the same sense map to the same AMR. Although these methods shed light on extracting and learning commonsense knowledge representations, they still depend on expensive annotated datasets. Researchers are still working on ways to reduce annotation efforts and improve parsing quality.

A *script*, a prototypical event sequence, is a structured knowledge representation. The core concept of *script learning* is that if events repeatedly happen together in everyday life, the events' relationships become human common sense, such as "hungry and eat", "tired and sleep". Early approaches measured co-occurrence probabilities of symbolic event representations. Recent works use neural network models to learn dense vectors for events and use vector similarity to represent their relatedness. Script learning attempts to capture three important characteristics of commonsense reasoning:

1. As with script learning, commonsense reasoning is stochastic in nature. It relies on information that has already been observed to infer what might happen next.

2. Any additional context can effect inference.

3. Imperfect reasoning exists, since no learned knowledge representation works in every situation.

Given these stochastic characteristics, script learning can use weak supervision such as trained classifiers or rule-based systems to extract noisy events and their relationships, as long as the correctness is better than random. Therefore, script learning is a cost-effective and scalable way to learn commonsense representations.

1.3 Learning Framework

For readers without a background in representation learning, we introduce the general learning framework used in this thesis. We understand the framework based on the three human cognitive systems presented in Figure 1.1. The framework contains two steps as a pipeline. The first step pretrains initial knowledge representations using a huge amount of data, which observes the world by mimicking the perception system. The learned representations, based on their accessibility, are used in the intuition and reasoning systems as general knowledge. The second step fine-tunes the representations for a downstream task, or learns a new reasoning engine on top of the representations. This step emulates the deliberate thinking process of the reasoning system. This framework is widely adopted for representation learning in NLP. From early static word embeddings such as word2vec [9] to recent Transformer-based language models such as BERT [10], all share this framework. We apply the framework in the four works presented later.

1.4 Research Questions, Goals, and Contributions

Understanding narrative text requires reasoning about characters' states, actions, and goals. Some information is not explicitly mentioned because it is common sense. Whereas this is straightforward for humans, machine readers often struggle as correct analysis relies on making long-range commonsense inferences over the narrative text.

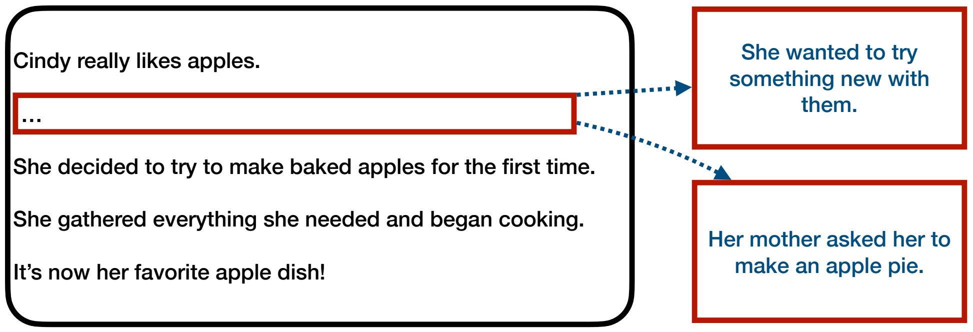

Cindy really likes apples.
...
She decided to try to make baked apples for the first time.
She gathered everything she needed and began cooking.
It's now her favorite apple dish!
She wanted to try something new with them.
Her mother asked her to make an apple pie.

Figure 1.2. Narrative example from StoryCommonsense [11]

Consider the short story in Figure 1.2. The character Cindy expresses a motivation in the event "*She wanted to try something new with them*". As humans, we understand that her motivation can be out of curiosity, and her emotions can be joyful and anticipating. At the end of the story, the event "*It's now her favorite apple dish!*" implies that she has accomplished her desire with positive emotions such as happiness, even though these are not explicit in the text. However, if the story has another motivating event, say "*Her mother asked her to make an apple pie*," the character emotions, reactions and outcomes could be very different. There are also nuanced relations among the events, motivations, and emotions, which humans identify seemingly effortlessly.

The goal of this thesis is to advance script learning methods to model commonsense knowledge from text. Three research questions are investigated in order to improve existing methods under the weakly supervised setting:

1. How to better account for event internal structures?
2. What event relationships are useful and feasible, beyond co-occurrence?
3. How to dynamically consider different situations?

To answer these questions, we build neural event encoders that encode a symbolic event representation into a dense vector representation, which embeds commonsense knowledge for event relations. We seek to build a universal event encoder using weakly supervised methods on huge corpora so that the learned event representations can be general enough to be applied in diverse downstream tasks.

Our "3 + 1" contributions, which include three technical improvements and one work that combines them for commonsense applications, are summarized as follows:

- We identify the importance of entity-based features for representing commonsense knowledge with script learning.
- We create one of the first, if not the first, script learning models to address the multi-relational nature between events.
- We release contextualized event representations trained on large-scale newswire data.
- We develop a graphical model that combines entity-based features, typed event relationships, and grounded context, and show that it is a perfect fit for modeling psychological common sense.

The model performance is empirically verified in both intrinsic and extrinsic evaluations.

We present our contributions in four individual works, building richer event representations step by step. In the first work, unlike previous work which considers only coarse-grained event information, we encode fine-grained and entity-based properties into event representations. We hypothesize that encoding those properties embeds the characteristics of event participants into event representations. To demonstrate, we examine two properties that go beyond lexical information—entity sentiment and animacy—to investigate their impact on event inference. The evaluation results show that sentiment trajectories, especially for animate entities, yield significant improvements over representations without them, attesting our hypothesis.

In the second work, we examine the possibility of modeling multiple relation types between events. We observe that previous works only consider one simple relation—co-occurrence—and measure relatedness by similarity. Other types of event relationships are essentially ignored. We hypothesize that modeling multiple relation types between events helps generalize event representations, which capture information about multiple story lines. We extract multiple types of event relations based on event co-occurrence and discourse transitions. We use two types of event co-occurrence—event pairs with or without shared entities—and nine cherry-picked discourse relations, based on the Penn Discourse Tree Bank (PDTB) [12]. Empirical results show that our models are able to control the direction of inferences, disentangle multiple story lines, and enhance the quality of event representations.

In the third work, we propose a method to build contextualized event representations, dynamically differentiating event representations under distinct contexts. The core concept is that events under different situations should be represented differently. We define the narrative graph (NG), a graph structure in which nodes represent events and edges reflect typed relationships. The graph structure emulates the situation in which the events are involved. We leverage a pretrained language model to encode local event nodes and contextualize them with a neural graph model. Experimental results show that contextualizing event representations on a multi-relational event graph greatly outperforms other strong baselines.

In the fourth work, we extend the NG to an entity-based narrative graph (ENG), in which each node now sticks to an entity mention and an event predicate in pairs. The model is intended to shape the node representation to capture interactions between implicit entity states when they are participating in an event, similar to the way humans infer the mental states of other humans. We believe this is key to solving many commonsense reasoning tasks that require understanding mental interactions. We also study multiple pretraining tasks and their impact on downstream tasks in the context of psychological common sense. Empirical results show that ENG successfully captures psychological interactions and improves performance on three multi-label state classification tasks (Maslow, Reiss, and Plutchik) and one desire fulfillment task. This final work combines all the advances of script learning introduced in this thesis.

In sum, our works advance script learning by adding three elements into event representations: (1) entity-based event properties, (2) multiple types of event relations, and (3) dynamic event contextualization. Intrinsic and extrinsic evaluations both attest the effectiveness of these improvements. This thesis constitutes a thorough study of generalized event representations for commonsense reasoning.

1.5 Road Map

The rest of this thesis is organized as follows. Chapter 2 motivates the script learning methods presented in this thesis, and provides formal definitions of the problems at hand. Chapter 3 reviews the literature on related tasks and script learning models to facilitate later discussion. Chapter 4 presents our first contribution, also published in [13], which builds feature-enriched event representations and investigates the impact of learning with two cherry-picked feature sets. Chapter 5 covers our second contribution, also published in [14], which to the best of our knowledge is the first work to consider multiple types of event relationships for script learning. We show that it is possible to extract discourse relations between events under a weakly supervised setting. Chapter 6 presents a way to extract a novel narrative graph and a model to learn event representations using the graph, also published in [15]. Chapter 7 describes an enhancement for the graph model presented in Chapter 6, adding the flexibility to involve entity-based states. This chapter also studies different pretraining tasks for the graph model and their impact on downstream tasks. Chapter 8 summarizes this thesis.

2. PROBLEM DEFINITION

"If I had an hour to solve a problem and my life depended on the solution, I would spend the first 55 minutes determining the proper question to ask, for once I know the proper question, I could solve the problem in less than five minutes."

—Albert Einstein (1879–1955)

We seek to build a universal event encoder to capture commonsense knowledge with neural models, which can benefit downstream tasks. In this thesis, we approach this event encoder by reasoning the most likely next event, i.e., script learning, under a general framework consisting of pretraining and downstream training. In this chapter, we formally define the framework for model construction and evaluations and discuss the motivation behind it in the context of previous work, pinning down the processes required for training, inference, measures, and metrics.

2.1 Why Script Learning?

How do people determine if a particular behavior is appropriate for a particular situation? For example, when you walk into a coffee shop, how do you know to order coffee from the barista rather than ask for a seafood feast? You find that certain behavior is appropriate under certain circumstances, because you have seen it so often in your daily life, which is manifest as human common sense. Script learning is derived from this intuition. We teach machines to learn from scripts, which are sequences of events, emulating human daily-life observations, assuming that the machines capture patterns between events and accumulate common sense.

In the literature, researchers have also attempted to automate the commonsense retrieval by constructing huge knowledge bases such as Cyc [6], ConceptNet [16], and ATOMIC [17], but this actually worked out to building an information retrieval system that maps human queries to events, or concepts, rather than representing their semantics. There are studies on learning representations from knowledge bases such as ConceptNet Embeddings [18] and COMET [19] which lie somewhere between our methodology—script learning—and knowledge base retrieval. Although they make progress toward general commonsense reasoning, constructing and cleaning the required knowledge bases is expensive, and such knowledge bases are not available in many resourceless languages. In this thesis, we approach commonsense reasoning through script learning, as we believe this is cost-effective and scalable, and has the potential to achieve success through scalability like recent language models. For instance, GPT-3 [20], a popular language model, has 17 billion parameters trained with 500 billion tokens, achieving state-of-the-art performance on various NLU tasks. Although we did not experiment with such a huge amount of data, we seek to build a solid foundation for scalable script learning.

2.2 Formal Description

Let $\mathrm{e} = \langle \mathrm{e}_1, \mathrm{e}_2, ..., \mathrm{e}_n \rangle$ be a finite set of events to be encoded. The event definition depends on the target task. In general, based on how events are ordered into a sequence, there are two categories: entity-based and predicate-based events. Each entity-based event is associated with a specific entity mention and its action (or state), whereas each predicate-based event anchors at a predicate, and related participants, or context, serve as modifiers to the event. In this thesis, we use $\mathrm{e}^{\mathrm{e}} = \langle \mathrm{e}^{\mathrm{e}}_1, \mathrm{e}^{\mathrm{e}}_2, ..., \mathrm{e}^{\mathrm{e}}_n \rangle$ to denote that the current model uses the entity-based definition, and $\mathrm{e}^{\boldsymbol{p}} = \langle \mathrm{e}^{p}_1, \mathrm{e}^{p}_2, ..., \mathrm{e}^{p}_n \rangle$ to indicate that the current model uses the predicate-based definition.

Table 2.1 illustrates the two event definitions with an example. Given the raw text, we identify two human entities: *Jenny* (x_0) and *Kelly* (x_1). For predicate-based events, the system first identifies all verbs in the raw text to create event instances, and then populates the arguments with the participants. This definition treats all events equally without considering protagonists. For entity-based events, we first identify all the mentions of an entity, say x_0, and for each of these identify the verb and other participants. For x_0 and x_1, we respectively observe a sequence of events, corresponding to the entity's action chain. This definition better explains the narrative for a given protagonist.

Let v_{i} be the event representation of e_{i}: $\mathrm{mod}(\mathrm{e}_{\mathrm{i}})$ denotes the set of modifiers for e_{i}, and $v_{\mathrm{i}} = f(\mathrm{e}_{\mathrm{i}}, \mathrm{mod}(\mathrm{e}_{\mathrm{i}}))$. The modifiers can be—but are not limited to—event features, descriptions, and context, depending on the needs of the encoder models. This function

Table 2.1. Event Definition Example

Raw Text	*Jenny went to her favorite restaurant with Kelly, her friend. Jenny ordered a lasagna plate. Kelly admired Jenny so she ordered the same. Jenny liked the food but Kelly did not like it.*
Entities	x_0=Jenny, x_1=Kelly
Predicate-Based	go_to(jenny, restaurant), order(jenny, plate), admire(kelly, jenny), order(she, same), like(jenny, food), not_like(kelly, it)
Entity-Based (x_0)	go_to(x_0, restaurant), order(x_0, plate), admire(kelly, x_0), like(x_0, food)
Entity-Based (x_1)	order(x_1, sandwich), admire(x_0, jenny), not_like(x_1, it)
Predicate-GR (x_0)	(go_to, subj), (order, subj), (admire, dobj), (like, subj)
Predicate-GR (x_1)	(order, subj), (admire, subj), (not_like, subj)
EventComp "Sentence" (x_0)	(go_to, subj), arg:jenny, arg:restaurant, (order, subj), arg:jenny, arg:plate, (admire, dobj), arg:kelly, arg:jenny, (like, subj), arg:jenny, arg:food
EventComp "Sentence" (x_1)	(order, subj), arg:kelly, arg:same, (admire, subj), arg:kelly, arg:jenny, (not_like, subj), arg:kelly, arg:it

denotes that a given event e_i with different features, or properties, or in a different context, can have different representations. The goal of this thesis is thus to learn the event encoder $f(.)$ that provides the best event representation initialization to downstream tasks.

Figure 2.1 visualizes the framework. Training is a pipeline that consists of a pretraining and a downstream-training phase, each with separate objectives. Let $\mathscr{L}_p$ be the pretraining objective function and $\mathscr{L}_d$ be the downstream training objective function. Each objective function contains a scoring function (S_p and S_d respectively) which is a neural network to guide the representations to shape event relationships, e.g., maximize the embedding similarity for co-occurrent event pairs. The network takes in multiple event representations, encoded by the event encoder $f(.)$, and feeds them into a set of feedforward layers called the *head* to score the relationships among events. Note that when one input example has multiple events to be encoded, if we do not mention this explicitly, the event encoder has

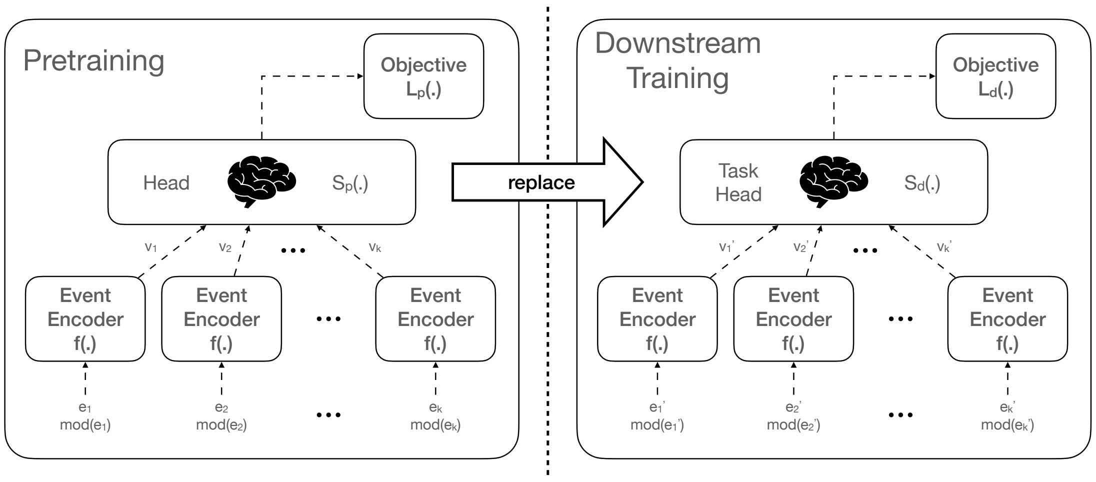


Figure 2.1. Framework visualization

only one unique instance, which means all the events are encoded with the same set of parameters.

For downstream training, we remove the head from pretraining, and retain the pretrained encoder alone with a new task-specific head, which gives the model a better starting point for the downstream task.

2.3 EventComp Model

In this section, we introduce EventComp [2], an important baseline model that was a milestone of script learning; more importantly, the authors proposed *multiple-choice narrative cloze* (MCNC), a standardized benchmark task widely adopted for model comparisons. We will introduce this in Section 2.4.

We explain EventComp in detail using our framework, because on one hand EventComp is an intuitive model that can be used to demonstrate how to use our framework in general for model explanations. On the other hand, we believe this section offers a direct comparison between our models and theirs, facilitating discussions.

2.3.1 EventComp Events

In the context of EventComp, each event is in a form called *Predicate-GR*, an event definition proposed by [1], which is the first work to model scripts using statistical methods. Each Predicate-GR event is associated with an entity mention and syntactically represented as a pair of the verb and grammatical dependency type to the mention, i.e., *(verb, dep)*. In the case of EventComp, the dependency type can be *subject* (subj or arg0), *direct object* (dobj or arg1), and *indirect object* (iobj or arg2); to reduce sparsity, the verb is lemmatized. The two rows at the bottom of Table 2.1 show the usage of the previous example.

These event sequences are then compiled into "sentences" as the model input. This is done by adding argument tokens to the Predicate-GR events (see EventComp Sentence in Table 2.1). As suggested in related work [21], adding argument information, even just the headword, aids in commonsense inference. These argument tokens augment entity-based events with argument information that was captured only by predicate-based events, while retaining the entity-centric nature.

2.3.2 EventComp Training

There are two steps in EventComp pretraining. In the first step, given the created "sentences", each token is treated as a word in a sentence, to which the existing word embedding models are applied for pretraining. The Word2Vec skip-gram [9] is used, a popular model characterized by efficiency and scalability. To set up the objective, the model follows the distributional hypothesis: words that occur in the same contexts tend to have similar meanings [22]:

$$\mathscr{L}_{p,1} = -\frac{1}{N}\sum_{\mathrm{i}=1}^{N}\sum_{\mathrm{j}\in C_\mathrm{i}} \log S_{p,1}(w_\mathrm{j}, w_\mathrm{i}), \tag{2.1}$$

where w_i is the i-th token, and C_i is the context of w_i. $S_p(w_\mathrm{j}, w_\mathrm{i})$ is the network that approximates the conditional probability of w_j given w_i:

$$S_{p,1}(w_\mathrm{j}, w_\mathrm{i}) \propto p(w_\mathrm{j}|w_\mathrm{i}) \tag{2.2}$$

$$\propto \frac{\exp(v_\mathrm{i}\cdot v_\mathrm{j}^T)}{\sum_\mathrm{j}\exp(v_\mathrm{i}\cdot v_\mathrm{j}^T)}, \tag{2.3}$$

where v_i is the representation of w_i. As estimating this function is costly, the authors propose negative sampling and hierarchical softmax sampling strategies. The explanations for these are omitted here as they are not the focus of this thesis. Please refer to the Word2Vec paper [9].

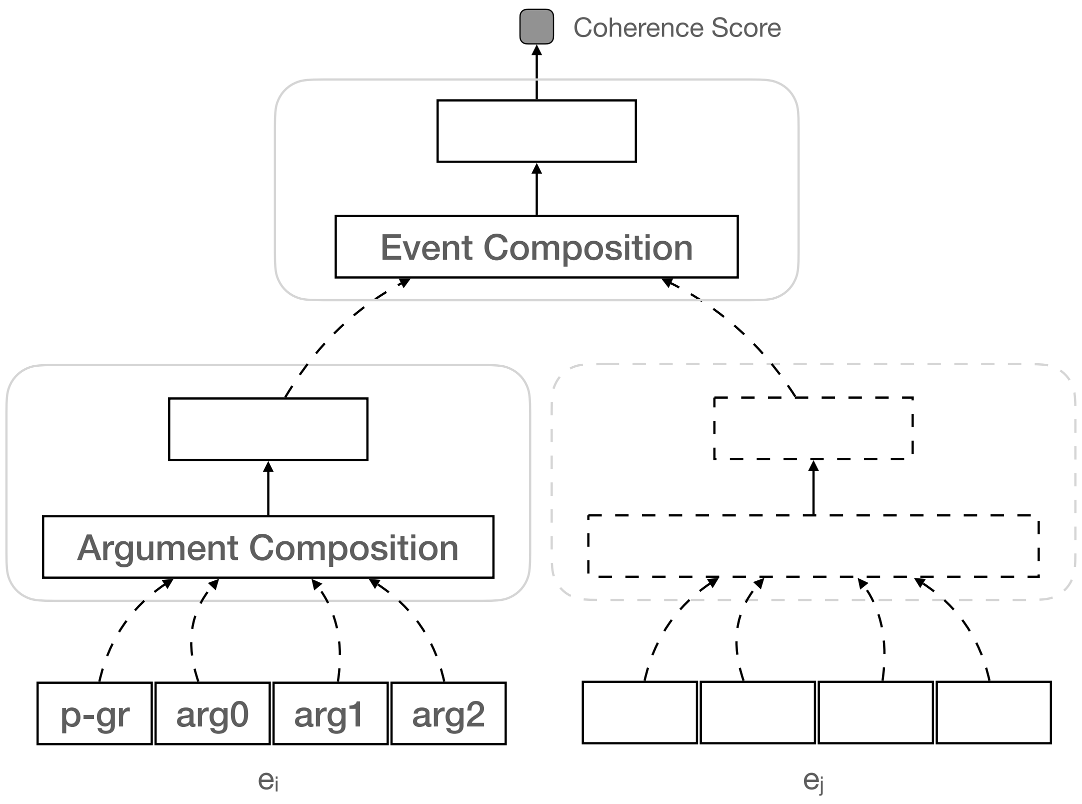


Figure 2.2. EventComp Model Architecture. The argument composition layer on the left shares the parameters with that on the right.

In the second step, once the token representations are trained, the token representations are combined into event representations by concatenating the Predicate-GR representation (p-gr) with the representations of its arguments (arg0, arg1, and arg2), shown in Figure 2.2. Next, this is projected to an argument composition layer and then an event composition layer, each of which contains a feedforward layer. Lastly, the final layer is projected to a one-dimensional coherence score that captures the co-occurrence of the two events. The objective function is a binary cross-entropy variant:

$$\mathscr{L}_{p,2} = -\frac{1}{K}\sum_{\mathrm{i}=1}^{K}\log(y_{\mathrm{i}}S_{p,2}(\mathrm{e}^{\mathrm{e}}_{\mathrm{i},1},\mathrm{e}^{\mathrm{e}}_{\mathrm{i},2}) + (1-y_{\mathrm{i}})(1-S_{p,2}(\mathrm{e}^{\mathrm{e}}_{\mathrm{i},1},\mathrm{e}^{\mathrm{e}}_{\mathrm{i},2}))) + \lambda L_2(\theta), \tag{2.4}$$

where K is the number of sampled event pairs, $y_{\mathrm{i}} \in \{0,1\}$ denotes the negative and positive samples, $\mathrm{e}^{\mathrm{e}}_{\mathrm{i},1}$ and $\mathrm{e}^{\mathrm{e}}_{\mathrm{i},1}$ are the sampled pair, λ is the regulation coefficient, and $L_2(\theta)$ is the L-2 regularization term with respect to all the trainable parameters.

To sample positive event pairs, for event $\mathrm{e}_{\mathrm{i},1}$, the authors randomly sample an event $\mathrm{e}_(\mathrm{i},2)^+$ that co-occurs in an entity-based (coreference) event sequence. For each positive pair, another negative event $\mathrm{e}_(\mathrm{i},2)^-$ is sampled that does not appear in the current sequence.

2.3.3 EventComp Discussion

To date, script modeling has been evaluated by identifying (the link prediction of) events connected via coreference links, specifically by constructing a similarity function between events $S_p(\mathrm{e}_{\mathrm{i},1},\mathrm{e}_{\mathrm{i},2})$. EventComp is one example, which learns the similarity function from scratch. Some early works [1] define the similarity function using pointwise mutual information (PMI), whereas other works directly minimize the embedding similarity [23]. In this thesis, we study whether using similarity-based methods is sufficient to capture event relationships.

Entity-based methods such as EventComp are prevalent in script learning, compared to predicate-based methods. Early work [1] began by following narratives for a specific entity to extract common patterns from each entity's event sequences. Such methods model event relationships based on entity coreferences. However, this may not be the best way to model event relationships. Predicate-based methods are more common for event relationship parsing, such as when identifying temporal or causal relationships between events [24]–[26], which heavily rely on manual labeling. Although such direct supervision does not match the weakly supervised setting of this thesis, it does show that events which appear in neighboring text, whether coreferenced or not, are likely to have strong causal or temporal relationships.

Thus we pose questions 1) *Why not model both types of relationships?* and 2) *Are there other types of event relationships that could be extracted without human effort?*

In addition, assuming that our model already captures event relationships perfectly, are there missing pieces of knowledge that are not observable by the model that harm inference performance? Often it is helpful to present implicit states of events, or participants, to the model. For instance, a happy event is likely to be followed by another happy event.

Strengthening these three aspects to advance script learning is the goal of this thesis.

2.4 Measures and Metrics

Evaluating the knowledge obtained through script learning is challenging. Early work conducted in the '70s defined a script as a sequence of structured events organized in temporal order [27]. Although such early work provided the core concepts of script knowledge, they manually created small sets of scripts, which offer little value in terms of evaluations. The first statistical approach to script learning was introduced by [1], [28]. They also proposed an evaluation task suited for statistical approaches termed *narrative cloze* (NC). Building on the outputs of a coreference resolution system and a dependency parser, narrative event chains were automatically extracted by following mentions of an entity through the narrative text. A NC question is created by removing an event from an event chain; the model is tested on its ability to rank the correct answer over all possible alternatives.

Despite the popularity of the NC test [29]–[32], it raises several difficulties for evaluations. First, there is no standard dataset, complicating comparison between different models. Second, for any given event there are multiple reasonable choices for subsequent events. Evaluating based on a specific choice is somewhat arbitrary. [30] use human evaluation to determine the chosen candidate plausibility. Although this provides good intuition, this is difficult to carry out at scale. Third, the extremely large vocabulary size leads to computational issues. Early work [1], [28], [33] represents events as simple (predicate, dependency) pairs, resulting in a relatively manageable event vocabulary size. Other work [2], [21], [30], [32] explores rich symbolic representations over multi-argument events, which increases the vocabulary size by orders of magnitude, again leading to computational issues. This problem can be addressed by significantly reducing the vocabulary at test time [30]. [2] thus proposed a multiple-choice variation called *multiple-choice narrative cloze* (MCNC), which simplifies the evaluation process and reduces its computational burden.

Indeed, many recently proposed tasks evaluating script knowledge follow this approach and use multiple-choice evaluation. For instance, the Story Cloze Test [34], [35], multiple-choice evaluation over possible story endings, and SemEval-18 Task 11 (http://alt.qcri.org/semeval2018/), multiple-choice evaluation based on commonsense script knowledge inference. Although relevant, these tasks do not directly evaluate the learned model quality, leading to our evaluation task decision.

In this thesis, although we also evaluate our models with a variety of intrinsic and extrinsic tasks, MCNC is the key intrinsic task for our model and for model comparisons. We follow the settings of [2] by extracting a set of event chains $S = \{s_1, s_2, \ldots, s_m\}$ from corpora, where $s_\mathrm{i} = \{\mathrm{e}_1, \mathrm{e}_2, \ldots, \mathrm{e}_k\}$. Note that we fix the chain length to k: we truncate chains longer than k to the first k events and we ignore chains shorter than this. We create an MCNC question q_i from s_i by sampling w negative candidates for the k-th event in each chain such that $q_\mathrm{i} = \{\mathrm{e}_1, \mathrm{e}_2, \ldots, \mathrm{e}_k, n\mathrm{e}_1, n\mathrm{e}_2, \ldots, n\mathrm{e}_w\}, \forall \mathrm{i} = 1 \ldots m$, where $n\mathrm{e}_\mathrm{i}$ is the i-th negative event candidate. The final MCNC question set is $Q = \{q_1, q_2, \ldots, q_m\}$. Each question asks models to identify

$$\underset{\mathrm{e}^* \in \{\mathrm{e}_k, n\mathrm{e}_1, n\mathrm{e}_2, \ldots, n\mathrm{e}_w\}}{\arg\max} P(\mathrm{e}^* | \mathrm{e}_1, \mathrm{e}_2, \ldots, \mathrm{e}_{k-1}). \tag{2.5}$$

Following [2], we set $k = 8, w = 4$ and use accuracy as the evaluation metric. Hence a uniformly random baseline should yield an accuracy of 20%.

3. REVIEW OF RELATED LITERATURE AND STUDIES

"If I have seen further than others, it is by standing upon the shoulders of giants."

—Issac Newton (1643–1727)

In the last chapter, under our problem settings, we defined the target problem and discussed the baseline model EventComp [2], the creators of which present an overview about conducting script learning in a prototypical neural-based manner. We also discussed MCNC, the standard benchmark task which facilitates the discussions in the rest of this thesis.

This chapter will briefly present related models and tasks that are enlightening for advancing script learning. The general history of commonsense reasoning will be addressed, after which previous and contemporary approaches for conducting commonsense reasoning will be presented. Among these, as this work concerns script learning models, we will discuss their script learning model variants and design choices as well as their connection to our work. Then, event-related tasks will be introduced to account for more possible evaluation benchmarks.

3.1 Commonsense Reasoning

Commonsense reasoning is the ability to use a basic level of practical knowledge, e.g., something that a six-year-old can understand, to reason about ordinary, daily-life situations. Early theory about commonsense reasoning was constructed by philosophers, psychologists, and cognitive scientists. Heider [36] examines interpersonal relations with psychological analyses, relating human perceptions, mental states, and reactions. Sellars [37] evokes the distinction between *manifest* and *scientific* images, i.e., commonsense and empirical evidence. In the '80s, commonsense reasoning was further divided into lines of work such as intuitive physics [38] and commonsense psychology [39]. To facilitate commonsense reasoning, AI researchers seek ways to represent commonsense knowledge. In the literature, there are two broad categories of approaches: ontology-based and script-based representations.

3.1.1 Ontology-based Approaches

Ontology-based approaches build a comprehensive ontology for all possible concepts relevant to human daily life. Early work focuses on taxonomic knowledge. OpenCyc [6] is a compilation of millions of language-like descriptions that can be used to compose human common sense, with a focus on implicit knowledge typically not mentioned in human communication, e.g., “All trees are plants.” ConceptNet [16] is a semantic network, representing commonsense knowledge as a semantic network that connects “concepts” with typed and directed edges. These concepts cover broader information than “events”, including human desires and goals; edge types include a closed set of pre-defined relations such as causal, temporal, sub-event, has-properties, etc. ConceptNet has evolved to link data sources for the concepts to be used with modern NLP techniques; e.g., many NLP scientists use ConceptNet as an external knowledge source to solve their target tasks [40]–[42]. Recent ontology-based systems focus on more specific sets of relations. ATOMIC [17] concerns if-else relations, which occupies only 1% of ConceptNet. The work represents events as free-form phrases with structured relationships, such as desires, needs, intents, and reactions. Event2Mind [43] further scopes this down to the relation between events and mental states. Machine learning researchers regard ontologies as knowledge bases for learning knowledge representations such as ConceptNet word embeddings [18], automatic knowledge base completion on ATOMIC [19], and a commonsense inference task based on Event2Mind [43].

Ontology-based approaches have three main drawbacks. First, construction of a knowledge base is a labor-intensive, expensive process. Second, for large-scale corpora, as they rely on knowledge graph completion techniques to make up for missing relations, it is difficult to guarantee the quality. Third, searching for related knowledge in an ontology-based system is challenging. On one hand, even humans often do not know in advance what exact commonsense knowledge is needed to solve a question. On the other hand, existing methods relying on information retrieval techniques such as keyword queries often reveal no useful knowledge, as commonsense knowledge tends to be implicit in text.

3.1.2 Script-based Approaches

As script-based approaches model commonsense knowledge directly from text, they have the potential to compensate for the shortcomings of ontology-based approaches. Early works established theories of *scripts*, from psychology to AI. Tomkins [44], a psychologist, proposes *script theory*, which assumes that human behaviors fall into patterns, like a written script, and describes such patterns as prototypical event sequences. Subsequently, Minsky [45] and Rumelhart [46] manually crafted small sets of scripts to study reasoning. Schank and Abelson [27] extend the idea to the AI field, using scripts as a structured representation to organize commonsense knowledge. Although the interest of AI research slowed down in the '80s, these early efforts laid a strong foundation for script learning. Based on how they organize event relations, we classify script learning models into three categories: *pairwise*, *sequential*, and *graphical*.

Pairwise Models

Pairwise models take event pairs as model inputs. Using our notation, the model scoring function can be written as

$$S(f(\mathrm{e_i}, \mathrm{mod}(\mathrm{e_i})), r, f(\mathrm{e_j}, \mathrm{mod}(\mathrm{e_j}))) \tag{3.1}$$

Events $\mathrm{e_i}$ and $\mathrm{e_j}$ are encoded with the event encoder $f(.)$. Relation type r is not considered in previous studies, which usually consider only one relation type: co-occurrence. To the best of our knowledge, this is the first study to consider multiple relation types for script learning. More details will be presented in Chapter 5.

Chambers and Jurafsky [1] first revived interest in script learning, introducing a statistical approach to estimate the scoring function $S(.)$. They proposed an unsupervised framework by which event sequences are extracted from text. Building on the outputs of a coreference resolution system and a dependency parser, narrative event sequences are automatically extracted by following mentions of an entity through the narrative text. The relatedness between events is computed using pairwise mutual information (PMI) scores.

This unsupervised framework has been widely adopted by the script learning community, with much effort spent on improving scoring. Jan et al. [33] yield improved performance by taking partial event orders into consideration. Rudinger et al. [47] apply a log-bilinear language model [48] to further improve performance. Pichotta and Mooney [21] suggest a multi-argument model, counting predicate co-occurrences with multiple entity arguments. Their empirical results show that including multiple participants helps when representing event semantics.

Many neural-network-based methods have been proposed to ameliorate the drawbacks of traditional statistical approaches. These works suggest replacing the symbolic event representation with a dense vector representation, which corresponds to the event encoder $f(.)$. EventComp [2], which we reviewed in the last chapter, belongs to this category. The authors apply Word2Vec [9] and a compositional neural network to learn a relatedness score for event pairs. Zhao et al. [49] model causality for event pairs with a dual cause–effect energy function. Webber et al. [23] use three-dimensional tensor-based networks to construct event representations, considering role-based and predicate-based compositions.

Sequential Models

Although the prevailing approach involves pairwise models, studies have also been conducted on modeling whole sequences for script learning. The main difference is that the former excels with partial event orders, whereas the latter emphasizes long-range temporal orders. Note that sequential models in general perform better on the MCNC task, as the task in nature is to reason about sequential relationships.

With sequential models, there are two possible ways to model event sequences. First, similar to pairwise models, we encode each event separately using the event encoder and then score the whole sequence, i.e.,

$$S(f(\mathrm{e}_1, \mathrm{mod}(\mathrm{e}_1)), f(\mathrm{e}_2, \mathrm{mod}(\mathrm{e}_2)), ..., f(\mathrm{e}_k, \mathrm{mod}(\mathrm{e}_k)), r), \tag{3.2}$$

Without losing generality, we keep the relation type r, which denotes that the sequence is constructed based on the relation r, although in the literature only a single type of relationship is considered. Pichotta et al. [30] use a sequential model—a long short term memory (LSTM) recurrent neural network (RNN)—to model event sequences. They serialize event sequences similar to the way EventComp does, but augment this with prepositional information. The results suggest that modeling an event sequence as a whole in neural models is beneficial. In another work, Pichotta and Mooney [50] include the textual context of events for predictions, using a sequence-to-sequence LSTM model to make text-level event predictions. The experimental results show that modeling events at the text level, which has textual context, offers superior performance.

The second way is to contextualize the event representation in the event encoder. That is,

$$f(\mathrm{e_i}, \mathrm{mod}(\mathrm{e_i})) = f(\mathrm{e_i}|\mathrm{e_j} \in C_\mathrm{i}), \tag{3.3}$$

where C_i is the context of $\mathrm{e_i}$, i.e., other events in the same sequence with $\mathrm{e_i}$. We can also interpret these contextual events as modifiers of $\mathrm{e_i}$ ($\mathrm{mod}(\mathrm{e_i})$). This contextualization can be accomplished by learning a weight for each event such that the target event representation is composed by its context (per the distributional hypothesis [22]). This is a popular technique in the language modeling literature [10] which we will discuss further in Chapter 6. Scoring is done either in the way denoted in Equation 3.2, or pairwise, by Equation 3.1. Wang et al. [32] adopt this approach and incorporate a dynamic memory component into LSTM to leverage deeper semantic representations.

Graphical Models

The graphical model is a generalization of sequence models. Let $G = (V, E)$, where G is an event graph, V is the node set (the events), and E is the edge set (the relations). Then we have

$$f(\mathrm{e_i}, \mathrm{mod}(\mathrm{e_i})) = f(\mathrm{e_i}|G_\mathrm{i}), \tag{3.4}$$

where G_i is the event graph that contains $\mathrm{e_i}$. The interesting part of this setup is that the target event representation is affected by long-range context, requiring a consistent view for the entire graph.

Although many studies organize scripts as a graph, for learning they still use pairwise or sequential models. The main challenge here is that modeling events and their relations as a graph requires considerable computational power and memory. In some works different methods are leverage to conserve resources. Orr et al. [51] simplify event graphs into tree structures and use a hidden Markov model (HMM) with expectation maximization (EM) for learning. Li et al. [52] approach these difficulties by breaking down the whole event graph into subgraphs. Each subgraph involves events in the same sequence, and a global adjacency matrix is used to measure the transition possibilities for all pairs of events in the subgraph. Note that no studies contextualize the event representation with the whole graph; this is indeed our contribution, which we will present in Chapter 6.

3.2 More Event-Related Tasks

We have introduced MCNC, our core task for evaluating script knowledge; this represents one of many branches of commonsense reasoning. In this section, we mention related commonsense tasks, from traditional Turing-based tasks to modern large-scale datasets for machine learning. All of these have the potential to utilize event representations to improve performance.

3.2.1 Coreference Resolution

The goal of coreference resolution is to cluster entity mentions in a document such that each cluster refers to a specific entity in the real world. The most popular benchmark is the CoNLL 2012 shared task [53], which is based on OntoNotes [54], an annotated corpus. Pichotta [55] empirically applies script learning models for this task.

The Winograd Schema Challenge (Winograd) [56] was proposed in the spirit of the Turing test, evaluating commonsense understanding by resolving coreferences for ambiguous pronouns. As the task's questions were manually created by experts, the dataset is limited in size and complexity. Modern models with NLP techniques such as Transformer-based language models [10] easily achieve high accuracy on this task. Recently, a more challenging and large-scale version called Winogrande [57] was proposed to upgrade the difficulty and size of Winograd. To reduce human bias introduced while creating the task's questions, an adversarial filtering algorithm was applied. Although the task design makes sense, it is still an open question as to whether this is the right benchmark for evaluating general commonsense reasoning.

3.2.2 Textual Entailment

The tasks in this category predict relationships between text spans in various forms. Zeller et al. propose SWAG [58], a textual entailment task for commonsense reasoning under grounded situations. By supplying a short text describing a situation, the task requires that systems reason the next event out of four choices. This setup is very similar to MCNC, except that MCNC requires long-range contextual understanding, whereas SWAG, relying on short descriptions, requires more background knowledge.

Mostafazadeh et al. build ROCStories [59], a crowdsourced large collection of simple stories, where the story length is roughly fixed, i.e., five sentences per story and less than ten words per sentence. Along with ROCStories, the authors also released Story Cloze Test, a story ending prediction task, adapting these stories into a new benchmark by requiring systems to choose between true and false endings to a story.

The Choice of Plausible Alternatives (COPA) task [60] is a similar benchmark for commonsense understanding of events and their relationships. In COPA, a system is presented with a premise and two alternatives that might have a causal relationship with the premise. Although COPA represents events as free-form text with structured relationships, unlike ontology systems, it covers a limited number of relations (cause and effect) and is smaller in scale (contains only 1,000 instances).

3.2.3 Abductive Reasoning

Abductive commonsense reasoning was proposed by Bhagavatula et al. [61] to reason the most plausible explanation for what happened between a given prior–post context pair. Unexpectedly, most questions in this task have stronger ties to the post context, as the answer is likely to explain the conclusion, which suggests that the prior context often does not matter. Such language bias is a large obstacle to creating quality abductive reasoning tasks that remains to be addressed.

3.2.4 Physical Commonsense

Other tasks focus on a specific area of commonsense. Automatically reasoning physical commonsense, such as “Fish swim *under* water (not above water)” or “Humans sleep *on* the bed (not under the bed)”, has seen a resurgence recently. Although its scope spans NLP, computer visions, and robotics, we focus on the text-related tasks. [62] attempt to capture procedural knowledge from cooking recipes. [63] propose a task of predicting implied physical scenes from verb usage. [64] measure knowledge about object sizes, such as “are elephants larger than dogs”.

3.2.5 Psychological Commonsense

Modeling implicit entity states such as mental states has been shown effective in many NLP tasks [65]–[69], whereas resolving the states and their interactions requires commonsense knowledge, which makes this type of task an ideal candidate for script knowledge evaluations. Rashkin et al. [11] created StoryCommonsense, an annotation of ROCStories [59] with human’s mental states, including motivations and emotions. To solve the task, systems must understand event impacts on multiple entities’ minds, resolving the complicated interactions. Sap et al. propose SocialIQA [70] for commonsense reasoning about motivations, emotional reactions, and plausible next events, grounded by a social situation. It is a question-answering task with multiple-choice questions. Again, solving the task requires that models understand human entities’ emotional reactions to events. Rahimtoroghi et al. propose DesireDB [71], which labels desires, such as “I want to ...” or “My goal is ...”, for a blog-like corpus, and sets up tasks to predict if a desire is satisfied or not. If the protagonist’s mental state progresses from “frustrated” to “happy”, this often implies the desire has been satisfied. Modeling such patterns is thus important for solving the task.

3.2.6 Discourse Relations

Parsing discourse relations is another type of task that requires strong commonsense knowledge. The CoNLL 2016 Shared Task [72] sheds light on this aspect. The task goal is twofold: (1) locate discourse connectives and their arguments and (2) classify discourse senses. Among these, classifying implicit discourse senses is strongly connected to commonsense, because unlike explicit discourse senses, where discourse connectives, e.g., *because* and *however*, provide strong cues to help identify the correct sense, implicit discourse senses can be inferred only from the two given argument spans, and as a result rely heavily on modeling common sense.

3.2.7 Event Relationships

Another category of tasks involves capturing different types of relationships between events. Most work of this type uses predicate-based events, and thus basically learns relationships between verb mentions.

In this category a crucial task is temporal relation extraction, in which commonsense knowledge plays an important role. For example, “ordering food” should occur *before* “eating food”, and “waking up” should occur *after* “sleeping”. Annotated corpora such as TimeBank [73], RED [74], TB-Dense [75], and MATRES [76] provide relation supervision for model learning. Based on these annotated corpora, early studies apply hand-crafted features with classifiers to locally determine edge types [77]–[80]. ClearTK [81] and UTTime [82] include more syntactic features. To strengthen the global consistency of those local decisions, later studies add rule-based constraints, such as in NavyTime [83] and CAEVO [84], and structured learning [85]–[87]. Recent works advance this with deep neural models [75], [88], [89].

Several works jointly consider multiple tasks to infer better decisions. Han et al. [90] relate event extraction with their temporal relations to conduct joint learning, as the two tasks are complementary. Ning et al. [91] jointly solve causal and temporal relations between events using ILP-based methods. This is an interesting direction since models could leverage a temporal assumption implicit to causal relations—i.e., “cause” should occur before "effect"—to improve the understanding of both relations. Many corpora are available with these two relation types. Causal-TB [92] is annotated with sparse temporal relations, and is even sparser for causal relations. CaTeRs [93] augments ROCStories [59] with temporal and causal relations between events, but the domain is limited to short stories. Ning et al. [91] re-annotates temporal and causal relations for the EventCausality dataset [94], which was created for causal relations only.

Other relation tasks focus only on causal relations, ignoring temporal relations, but such studies are relatively rare. Hidey and McKeown [95] use causal discourse markers such as "because" to create a dataset from Wikipedia to identify causality in text. Dunietz et al. [96] leverage construction grammar (CxG) to identify text spans with causal relations. Do et al. [94] compute cause-effect association (CEA) scores for event pairs to measure the level of causality. These scores are based on global statistics, distributional similarity methods, and discourse connectives.

3.3 What's Next?

In the next four chapters, we will propose four models to advance script learning. The first two models are pairwise models; the last two are graphical. The first model investigates innovative event properties that facilitate the representation of event semantics. The second model constitutes a new way to construct and learn event representations with multiple relational types. The third model builds a narrative graph for events, based on predicates, to contextualize event representations. The fourth model alters the third model for entity-based events to model psychological states and interactions.

4. FEATURED EVENT EMBEDDING LEARNING

"To live effectively is to live with adequate information."

—Norbert Wiener (1894–1964)

In the previous chapters, we describe backgrounds, motivations, and literature for modeling commonsense with script learning, and briefly introduce the four works we will present in this thesis. This chapter explains our first script learning model—Featured Event Embedding Learning (FEEL)—in details. FEEL enhances script learning models by injecting event representations with fine-grained and entity-based information. In addition to capturing the dependencies between subsequent events, our model can take into account higher level abstractions of the input event which help the model generalize better and account for the context in which the event appears. We evaluated our model over three intrinsic evaluation tasks. The first one is a standard benchmark, Multiple Choice Narrative Cloze (MCNC), and the other two are newly proposed sequential variants to MCNC, which measure models' ability to connect and explain a story line. We showed that our model is competitive with the most recent state-of-the-art. We also showed that our resulting embedding can be used as a strong representation for advanced semantic tasks such as discourse parsing and sentence semantic relatedness.

4.1 Introduction

Many natural language understanding tasks rely on world knowledge. Such knowledge can help support common sense reasoning and provide the context needed for disambiguating text. *Scripts*, introduced by [27], are structured knowledge representations capturing the relationships between prototypical event sequences and their participants. Scripts model our expectations about the relevant causal relationships between events, and as a result can be used to infer how events will unfold in a given scenario. For example, given the event *John shot Jim with a gun*, we can infer that *he was arrested by the police* is more probable than *he fell asleep*. Scripts provide the foundation for automatically making such inferences, supporting semantic tasks such as coreference resolution, discourse parsing and question answering.

As the example above suggests, predicting *"what happens next?"*, also known as the Narrative Cloze (NC) task [1], [2], is the preferred way of evaluating such models. In this work, we propose a generalization of this task highlighting the importance of evaluating inferences over chains of future events. We look into two variants of this task, the first predicts future events, and the second predicts an explanation, connecting the beginning and ending of a longer narrative chain. The following example is a simplified version of these tasks.

Narrative Cloze

Jenny went to a restaurant and ordered a lasagna plate. Jenny liked the food and felt satisfied.

Which of the following events could happen *next*?

a) *She scolded the server.*
b) *She fell asleep.*
c) *She left a big tip.*
d) *She ran out of battery.*

Narrative Explanation

Jenny went to a restaurant and left a big tip.

Which of the following event chains *explain* what happened?

a) *She ordered her food and liked it.*
b) *She hated the food and left angry.*
c) *She walked to a bus station and got on a bus.*

Humans can easily identify that *c)* and *a)* are the correct answers. However, automating this process requires understanding the events, their properties and their implications.

While early works focused on manual construction of script knowledge, the difficulty of scaling these methods to realistic domains has ignited considerable interest in statistical script learning methods [1], [2], [29], [31], [47], [97], [98].

We build on the previous work [1] which used pairs of event predicates and dependency information (corresponding to the *subject/object* dependency links) to represent events and formed event chains based on coreference relationships between these pairs. Their model assumed a discrete event representation, and computed the Pairwise Mutual Information (PMI) between event pairs, in order to support inference tasks. Following the surge of interest in distributed representations of discrete objects [9], most recent approaches represent events using dense continuous vectors, known as event embeddings. For example, [30] and [2] both proposed a neural network model that composes event embeddings with their predicate, dependency, and argument information (subject, object, and prepositional object), either using a feed-forward architecture defined over pairs of events, or using a Recurrent architecture (in this case an LSTM) to capture the dependencies between longer sequences of events.

In this work we contribute to this body of work, and introduce **FEEL**—Featured Event Embedding Learning. Our model is designed to capture fine-grained event properties, that can be exploited to reduce ambiguity when inferring future events. For example, the sentiment polarity of a given event (e.g., *"Jenny liked the food"* implies positive sentiment), can impact the probability of future events (e.g., the probability of negative-sentiment events, such as *a)*, should decrease). The *animacy* of the event's arguments can also provide valuable information, as some actions can only be performed by living entities, and some events change meaning when taking inanimate objects as arguments (e.g., *"this song is sick!"* vs. *"this person is sick!"*). In our example above, option *d)* can be ruled out based on this information.

We focused on these two features, as they provide a useful abstraction, of the specific event (sentiment) or specific argument (animacy). However, there are many other event properties useful for language understanding. Our goal is to provide a general framework for including such information. Specifically, our model makes three contributions.

(1) *Novel Neural Architecture for Event Learning* We suggest to set up learning for event embeddings as multi-task representation learning. The joint objective combines both intra-event learning objectives (e.g., representing prototypical connections between arguments and predicates, such as *policeman* and *arrest*), and an inter-event objective which captures prototypical connections between events.

(2) *Features Enriched Event Embedding* Our architecture provides a highly flexible framework for injecting world knowledge and relevant contextual information needed to accurately represent events. This information is injected into the embedding learning step, resulting in a richer event representation. We specifically looked into higher level abstractions of events—the overall sentiment of the event and animacy information of the event arguments.

(3) *Structured Event Chain Evaluation* We evaluated our model in several settings, using both intrinsic and extrinsic tasks. We followed the evaluation settings created by [2], and showed that using the same resources, our architecture leads to improved performance. Our feature-enriched model resulted in further improvement. Since looking at single-event transitions can fall short of evaluating full scripts, we also defined two additional narrative inference tasks over event sequences. Finally, we evaluated our model by using the generated representation as features for two semantic prediction tasks.

4.2 Related Work

Early works conducted in the 1970s defined a script as a sequence of structured events organized in temporal order [27]. While these early works provided the core concepts of script knowledge, the manual methods employed were difficult to scale to complex domains. Interest in script learning was revived by [1]'s work, which introduced a statistical approach to obtaining script knowledge. They proposed an unsupervised framework to model event sequences. Building on the outputs of a coreference resolution system and a dependency parser, narrative event chains were automatically extracted, by following mentions of an entity through the narrative text. The relationship between events was computed using the PMI score. The model's ability to capture commonsense knowledge was evaluated using the NC task, in which one event is removed from the event chain and the model is evaluated by ranking all candidate events.

Despite the popularity of the NC test [29]–[32], it raises several difficulties. First, there is no standard dataset, making the comparisons between different models much harder. Second, for any given event there are multiple plausible choices for subsequent events. Evaluating based on a specific one is somewhat arbitrary. [30] used human evaluation to determine the chosen candidate plausibility. While providing good intuition, it is difficult to do at scale. Third, the extremely large vocabulary size leads to computational issues. Early works [1], [28], [33] represent events as simple (predicate, dependency) pairs, resulting in a relatively manageable event vocabulary size. More recent works [2], [21], [30], [32] explore rich representation over multi-argument events, which increase the vocabulary size by orders of magnitude, leading to computational issues. Previous work [30] addressed this problem by significantly reducing the vocabulary at test time.

The NC task was refined by [2] to include a closed set of options for replacing the missing event. The multiple-choice variant is a better fit for evaluating multi-argument events as it tests the quality of the model's commonsense judgments without searching the entire event vocabulary. Indeed, many recently proposed tasks evaluating script knowledge have followed this approach, and use multiple-choice evaluation. For instance, Story Cloze [34], [35], a multiple-choice evaluation over possible story endings, and SemEval-18 Task 11[1], which looks at multiple-choice evaluation based on commonsense script knowledge inferences. While relevant, these tasks do not directly evaluate the learned model quality, leading to our evaluation task decision.

In this work, we re-create [2]'s settings, but also introduce two additional intrinsic evaluation tasks—Multiple-Choice Narrative Sequence (MCNS) and Multiple-Choice Narrative Explanation (MCNE). These tasks were designed to evaluate the model's ability to infer longer events sequences, which better account for narrative structures. This approach joins other recent attempts to include reasoning over narrative structures as part of script knowledge evaluation [98].

Multiple neural-network based methods were proposed to improve the quality of the commonsense event patterns captured by the model. These works suggest replacing the symbolic event representation used by [1] with a dense vector representation. [2] applied Word2Vec [9] and a compositional neural network to learn event embeddings on narrative event chains, where each event token in the chain is either a predicate word or an argument word. [30] proposed a Long Short Term Memory Recurrent Neural Networks (LSTM-RNN), coupled with Beam Search algorithm, which conditions the event representation on longer sequences of previous events.

[1]http://alt.qcri.org/semeval2018/

The fact that argument information is very useful for improving the learned event embedding was implicitly captured in these works. Some other works [99] explicitly identified this fact. Following this intuition, we suggest that representing richer event properties, such as arguments, sentiment, animacy, or even event time and location information, can potentially improve the event representation. These act as event modifiers and should be considered by script learning models. The multi-task approach to learning embedding models has been previously explored when constructing social embeddings [100], where the authors learned embeddings for users co-located in a network graph with their properties. FEEL follows this direction and develops such extensions for general statistical script learning, which can take rich event properties into consideration.

4.3 Model

4.3.1 Model Overview

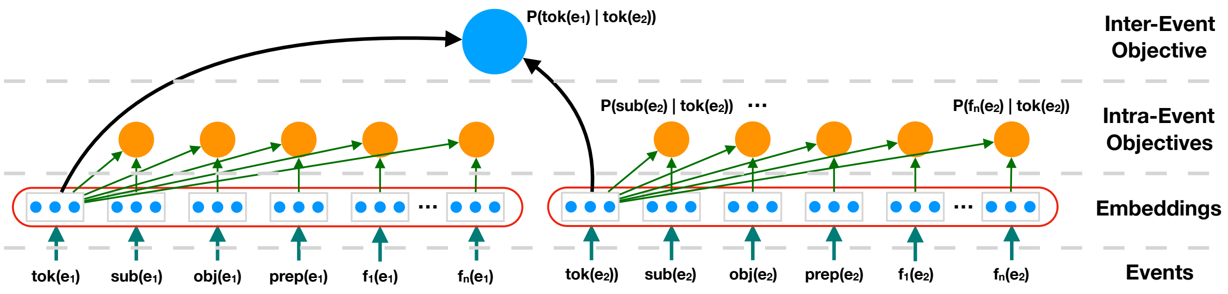


Figure 4.1. FEEL Model Objectives

From a high-level perspective, learning for narrative event models can be broken down into two phases.

First, large amounts of narrative text are preprocessed and event chains are extracted. Early systems [1] used a dependency parser (for connecting verbs and their typed arguments, resulting in a $(predicate, dependency\_type)$ event representation) and a coreference resolution system (for forming chains with the same protagonist). For example, *"Jessie killed a man. She was arrested."* has an event chain $(kill, subj), (arrest, obj)$ for the protagonist *Jessie*. Later systems [2], [30] included the argument words, as well as prepositional phrases. Second, these chains are used for training statistical script models. Initially this is done by computing the PMI between events [1] to capture event co-occurrence statistics. Later systems constructed event embeddings, connecting event tokens with their argument information to form the event representation.

FEEL follows this setup, but also adds an event property extraction step in between, which helps inform the training process. In the following subsections we describe the three phases in FEEL: (1) Narrative Event Chain Extraction, (2) Event Property Extraction, and (3) Model Training.

4.3.2 Narrative Event Chain Extraction

We first preprocess the text using Stanford CoreNLP [101], extracting dependency parses and coreference chains. We follow the coreference chains to form the event chains, by associating each entity mention in the chain with an event e^{e} defined as a tuple *(tok(e^{e}), subj(e^{e}), obj(e^{e}), prep(e^{e}))*, where *tok(e^{e})* $= (predicate, dependency\_type)$ is a token generated by concatenating the predicate and its dependency relation to the protagonist of the event e^{e}; *subj(e^{e}), obj(e^{e}), prep(e^{e})* are the subject word, object word, prepositional object word respectively of the event e^{e} (the superscript is for emphasizing that the event is entity-based, as described in Chapter 2). We abbreviate the event notation e^{e} to e in the rest of this chapter.

All the words are in lower-case and lemmatized, and we represent multi-word noun phrases, using their head word. For the running example given in Introduction, the chain associated with *Jenny* in the sentence *"Jenny went to a restaurant and ordered a lasagna plate"*, will be *((go, subj), jenny, NONE, restaurant), ((order, subj), jenny, plate, NONE).*

Several additional detailed processes are listed below:

- Predicates are not limited to verbs, and include predicative adjectives, which can provide important causal information. For example, *Jame was hungry. He ate a burger.*

- Verbs such as *go, have* and *get*, are too *weak* to express nuanced event semantics. We address this issue by including their particles and clausal complements (xcomp) in the predicate representation. For example, *go to sleep* will be represented as one token *go_to_sleep*.

- We include negations in the predicate, e.g., *didn't enjoy hiking* is represented as *not_enjoy_hike*.
- The possible dependencies of *d(e)* are limited to *subject, object, and indirect object.*
- We filter out high-frequency and low-frequency events empirically by removing the 10 most frequent events and the events that appear less frequently than a threshold t, where $t = 50$ in our case.

4.3.3 Event Property Extraction

FEEL provides a general framework for including event properties into its representation. Our first step is to include the argument information as a type of event properties; however, the true strength of the model is in modeling higher level abstractions of events. In this work we focused on two abstractive properties: sentiment, which captures the overall tone surrounding the event, and argument animacy information, which can help identify nuanced language use, such as idiomatic expressions. The motivation for putting sentiment and animacy in the same model is that both provide an abstraction, of the specific event (in the case of sentiment) or specific argument (animacy). The contribution of the different properties depends on the specific task. We designed our framework to incorporate additional properties allowing users to adapt their embedding to their specific task.

To incorporate sentence-level sentiment information, we use Vader sentiment analyzer [102] from NLTK [103]. The raw sentiment scores range from -1 (negative) to 1 (positive). We discretize the scores into sentiment labels–*Negative, Neutral,* and *Possitive*–by setting up two thresholds on -0.5 and 0.5. Animacy information is added by observing the animacy of the argument associated with the event token. There are three possible animacy types: *Animate, Inanimate,* or *Unknown.*

When adding the two event properties, the event 4-tuple is re-written as a 6-tuple *(tok(e), subj(e), obj(e), prep(e), se(e), ani(e))*, where $se(.)$ and $ani(.)$ refer to the sentiment and animacy extractors respectively.

4.3.4 Model Training

As illustrated in Figure 4.1, FEEL uses a hierarchical multi-task model for constructing the event representation, jointly learning for an inter-event (contextual) objective and several intra-event (local) objectives.

On one hand, the inter-event objective, defined over two events e_1 and e_2, captures the dependencies between subsequent events in a given narrative. In our running example, this objective can capture the relationship between the event *((go, subj), jenny, NONE, restaurant, NEUTRAL, ANIMATE)* and the event *((order, subj), jenny, plate, NONE, NEURTRAL, ANIMATE)* for the protagonist *Jenny*. On the other hand, the intra-event objectives are defined over the event properties, namely $tok(\mathrm{e}), sub(\mathrm{e}), obj(\mathrm{e}), prep(\mathrm{e}), se(\mathrm{e})$, and $ani(\mathrm{e})$, for the event e. Each will learn an embedding, represented in the embedding layer of Figure 4.1. This formulation allows the different properties and the event token to share information. Lastly, combining the inter-event and intra-event objectives forms the FEEL global objective function.

For the inter-event objective, we use the Skip-gram model [9], defined as follows:

$$\begin{aligned} p(C(\mathrm{e})|\mathrm{e}) &= \prod_{\mathrm{e}' \in C(\mathrm{e})} p(\mathrm{e}'|\mathrm{e}) \\ &= \prod_{\mathrm{e}' \in C(\mathrm{e})} \frac{exp(v_{\mathrm{e}'} \cdot v_{\mathrm{e}})}{\sum_{\mathrm{e}^* \in E} exp(v_{\mathrm{e}^*} \cdot v_{\mathrm{e}})}, \end{aligned} \tag{4.1}$$

where e is the current event; $C(\mathrm{e})$ is the context event set in a pre-defined window size k; E is the event vocabulary; and v_{e} is the vector representation of the event e. This can be learned by minimizing the margin-based ranking loss:

$$L_C(\mathrm{e}) = \sum_{\mathrm{e}' \in C(\mathrm{e})} \sum_{\mathrm{e}^* \notin C(\mathrm{e})} max(0, \delta - v_{\mathrm{e}} \cdot v_{\mathrm{e}'} + v_{\mathrm{e}} \cdot v_{\mathrm{e}^*}), \tag{4.2}$$

where δ is the margin; $(\mathrm{e}, \mathrm{e}')$ is a positive event pair; e^* is the negative example sampled from the noise distribution, forming the negative pair $(\mathrm{e}, \mathrm{e}^*)$. The Negative Sampling strategy [9] is used here. In our experiment, we use the uniform noise distribution over the event vocabulary, and set the window size $k = 5$ and the negative ratio $r = 10$.

The intra-event objectives model local information for each event independently. Each property is trained with the base event token $tok(\mathrm{e})$, which biases the learned embeddings to become more similar if the property tends to occur together with the token. The loss function is the same as the Equation (4.2), but, instead, takes the e' as the positive event property and the e^* as the sampled negative property.

FEEL jointly learns for all the objectives by taking a weighted summary:

$$\mathscr{L}_p(\mathrm{e}) = \sum_{\mathrm{i} \in \{C,S,O,P,T,A\}} \lambda_\mathrm{i} L_\mathrm{i}(\mathrm{e}) + \lambda_r \|w\|_2, \tag{4.3}$$

where $L_C(\mathrm{e})$ means the inter-event (context) objective of the event e; $L_S(\mathrm{e})$, $L_O(\mathrm{e})$ and $L_P(\mathrm{e})$ refer to the intra-event objectives between the e and its subject, object and prepositional object, respectively; $L_T(\mathrm{e})$ refers to the local objective between the e and the sentence-level sentiment; $L_A(\mathrm{e})$ refers to the local objective between the e and the entity animacy; λ_i is the weight for the objective $L_\mathrm{i}(\mathrm{e})$; λ_r is the weight for the regularization term $\|.\|$ and w refers to all the trainable parameters[2].

4.4 Experiments

We train the event embedding model over the New York Times (NYT) section of the English Gigaword [104]. It contains about 2M documents of newswire texts and about 1.4M words. We replicate the experimental set up described in the previous work [2], splitting the data into training/dev/testing sets accordingly. For FEEL, we use a 300-dimensional space to embed each property. Our full model (which includes the event token, subject, object, prepositional object, sentiment, and animacy) represents each event with the concatenation of all its property embeddings, which is 1800-dimensional.

[2]For simplicity, λ_i and λ_r are fixed to 1 in this work.

The FEEL embeddings are evaluated over three intrinsic tasks: (1) Multiple-Choice Narrative Cloze (MCNC), (2) Multiple-Choice Narrative Sequences (MCNS), and (3) Multiple-Choice Narrative Explanation (MCNE); and two extrinsic tasks: (1) Semantic Relatedness on Sentences Involving Compositional Knowledge (SICK), and (2) Implicit Discourse Sense Classification (IDSC).

4.4.1 Multiple-Choice Narrative Event Cloze (MCNC)

MCNC task is a multiple-choice variant of the NC task, which addresses the issues incurred by the evolution of multi-argument events, as described in Related Work. We follow the evaluation settings proposed in the previous work [2], which randomly sampled four extra choices from the vocabulary (the random guess baseline will have a 20% accuracy).

Table 4.1. The results of multiple-choice narrative cloze test. Accuracy and Mean Reciprocal Rank (MRR) scores are both reported. *PredDep*, *Args*, *S*, *A* respectively mean that the event token, argument, sentiment, and animacy properties are included in the training.

	Accuracy	MRR
Granroth-Wilding et al., 2016	0.4957	-
Wang et al., 2017	**0.5512**	-
PredDep	0.4232	0.6271
PredDep+Args	0.5135	0.6827
PredDep+Args+S	0.5166	0.6844
PredDep+Args+A	**0.5503**	**0.7096**
PredDep+Args+S+A	0.5418	0.7031

Table 4.1 shows the accuracy and Mean Reciprocal Rank (MRR) scores of our model over the test data. The first row lists the best score reported in the previous work [2]. The results obtained by [32] are reported in the second row. This very recent model exploits information about longer events chains using an LSTM. This approach follows a different intuition than ours, and we hypothesize that combining the two methods would result in an even better model. The rest of the table describe the results obtained by the variants of our model. *PredDep* is trained with the context (inter-event) objective only; the *PredDep+Args* model includes the arguments (subject, object, prepositional objects) in the objective; the *PredDep+Args+S* and *PredDep+Args+A* models include the sentence-level sentiment infor-

mation and protagonist's animacy respectively; and the *PredDep+Args+S+A* model contains all the information mentioned.

The results show that *PredDep* performs worse than *Granroth-Wilding et al., 2016*. This is not surprising as it does not model the event argument information. When this information is used (*PredDep+Args*), our model outperforms *Granroth-Wilding et al.*'s model. Moreover, when additional properties are used, our model's performance is improved, most significantly when using the animacy information. *Wang et al.*'s work gives the competitive result to FEEL by integrating event order information, which indicates that there are many more event properties that could be considered by FEEL.

Interestingly, the results show that including too many properties might hurt performance, as illustrated in the last row. The combination of sentiment and animacy information tends to lead to lower performance. After doing error analysis, we found that in MCNC task if the protagonist is inanimate, usually the model will prefer selecting inanimate choices for other arguments as well (generally correct). However, when adding sentiment information, it will prime the model to select animate options, as animate entities are associated with sentiment more often. This tells us that the two properties sometimes lead to conflicts when making the decisions. Therefore, carefully choosing the appropriate properties for the target application is important.

4.4.2 Multiple-Choice Narrative Sequences (MCNS)

MCNC evaluates the model's ability to infer an event from its context. A natural generalization of this task is to consider inferences over longer sequences of events, as these can better account for narrative structure, rather than pair-wise event relationships. We propose a new evaluation task—MCNS, and set it up by following these steps:

1. We sample n questions of length l from the MCNC.

2. We generated x choices for each event, except the first event. Note the difference from the MCNC, as a multiple-choice question is associated with each time stamp.

3. We modeled each event chain as a Markov Chain with l time stamps and $x + 1$ states at each time stamp, where the first time stamp only contains the starting state. We used an inference algorithm Viterbi [105] to identify the highest scoring event chain.

MCNS evaluates the model's ability to make longer commonsense inference, instead of just predicting one event. In this work we used a simple inference algorithm (a sequence model), but we consider incorporating advanced reasoning algorithms as a very promising direction for future work. In order to evaluate this approach, we used this algorithm over all of our script models. We also replaced the inference algorithm with a simple greedy baseline and perfect skyline.

- **Baseline:** No inference. Instead of Viterbi, for each time stamp, greedily pick the best transition and move to the next time stamp.
- **Skyline:** Break down a sequence of predictions into individual decisions, and give the correct previous state for each decision.

We also included a strong baseline text similarity model. The popular word embedding model—GloVe [106]—is used to score the transitions between events by computing the similarity between the averaged vectors of the event words. The contribution of the embedding generated by FEEL is observed by the performance difference when these embeddings are concatenated with the base GloVe event representation.

In this experiment, 1000 length-5 questions with 4 extra choices at each time stamp are sampled. Accuracy is used as the evaluation metric. The three columns on the left of Table 4.2 show the results, which indicate that in all cases FEEL embeddings offer performance gains when added to GloVe, even when only the event token information (*PredDep*) is provided. Both the sentiment and animacy property provide helps in this task. The best model (*GloVe+PredDep+Args+S*) brings the performance up to 0.416 from 0.332 using Viterbi. Similar to the MCNC results, using all event properties jointly (*GloVe+PredDep+Args+S+A*) does not improve performance, because of the decision conflicts stated previously.

Table 4.2. Results of MCNS (left) and MCNE (right) using Viterbi (-V), the skyline, and the baseline on FEEL and GloVe

	MCNS-V	Baseline	Skyline	MCNE-V
GloVe	0.353	0.297	0.356	0.385
GloVe+PredDep	0.359	0.302	0.362	0.389
GloVe+PredDep+Args	0.332	0.366	0.434	0.37
GloVe+PredDep+Args+S	**0.416**	0.385	0.460	**0.448**
GloVe+PredDep+Args+A	0.399	0.396	0.465	0.429
GloVe+PredDep+Args+S+A	0.365	0.383	0.452	0.403

4.4.3 Multiple-Choice Narrative Explanation (MCNE)

We suggest an additional extension to the MCNC task. The MCNE is also designed to evaluate reasoning over longer event sequences, similarly to MCNS. However, instead of just providing the initial event as input, in the MCNE task both the starting and the ending events are provided, and the prediction task is to infer what happened in between. Human commonsense can build an explanation that connects the two points. For example, given a beginning "Jenny went to a restaurant" and an ending "She felt satisfied", can a human figure out what might happen in the restaurant? One might guess that "she liked the food" is more likely than "she waited for an hour". MCNE provides a platform for script models to demonstrate such deeper understanding of world knowledge. The setup is exactly the same as MCNS, except leaving for the prediction at the final time stamp that is given as an input.

We use the same inference models as in MCNS. Note that when calculating the accuracies we did not include the ending state in both MCNS and MCNE, so the same baseline and skyline used in MCNS are applicable for this task. The right-most column of Table 4.2 summarizes the results.

We observe a very similar trend in the results to MCNS's, but with higher overall accuracies, since additional information is provided to the model during inference. The results also show our models' ability to improve commonsense reasoning using inference. Similar to MCNS, both the sentiment and animacy information help inferences, while the all-featured model results in a small performance drop. The main reason for this is the same as before.

Table 4.3. Pearson scores of SICK task on GloVe and FEEL

Pearson	PredDep	PredDep +Args	PredDep +Args +S	PredDep +Args +A	PredDep +Args +S+A
GloVe	0.7102				
FEEL	0.4452	0.6574	0.6791	0.6714	0.6714
GloVe +FEEL	0.7382	0.7572	0.7518	**0.7676**	0.7604

The results from MCNC, MCNS, and MCNE show that each property (sentiment or animacy) individually contributes more than the combination, and depending on the task the contribution of each type of information varies. For example, animacy makes the highest improvement in MCNC, but the sentiment information helps more in MCNS and MCNE for event sequences. This is because the sentiments have variable patterns along the sequences, while the animacy of the protagonist is almost fixed along the chain.

4.4.4 Semantic Relatedness on Sentences Involving Compositional Knowledge (SICK)

We also evaluate our embedding model as a feature representation for the SICK task.

This dataset was used as the popular shared task in SemEval-2014 [107]. It measures the semanitc relatedness of a given sentence pair. The gold relatedness score is averaged across ten human-annotated scores for each sentence pair, ranging from 1.0 to 5.0, where 1.0 means completely unrelated and 5.0 means very related. The training/dev/testing splits are available on the task website.

Our goal in these experiments is to evaluate whether the event embeddings can help capture the structural properties of sentence. To evaluate this property we augment a baseline system, which uses the GloVe word embedding, with our event embedding, and compare the performance over different variants of our embedding model.

Obtained a performance improvement over GloVe is not trivial. GloVe leverages global matrix factorization and local context windows methods to build general-purpose word embeddings, which have been shown to have better performance than the other popular word embedding model—Word2Vec [9]—in word similarity tasks. We use their 300-dimensional version, pre-trained on Gigaword and Wikipedia.

To construct the input sentence representation for both GloVe and FEEL, first, all the available embeddings in the input sentences are extracted and summed (word and event embedding separately). Second, these representations are fed into a neural-network-based regression model [108] for predicting the related scores. The network architecture is designed for text similarity tasks and is shown below:

$$\begin{aligned}
&h_* = v_{s_1} \otimes v_{s_2} \\
&h_\triangle = |v_{s_1} - v_{s_2}| \\
&h = h_* \oplus h_\triangle \\
&p = \text{softmax}(W \cdot h),
\end{aligned}$$

where $v_{s_1} \in R^k$ and $v_{s_2} \in R^k$ are vector representations of the first and second inputs respectively; $\otimes$ means element-wise multiplication; $\oplus$ is the vector concatenation operator; $W \in R^{5\times 2k}$ is the weight matrix to be trained; the 5 softmax outputs corresponds to scores 1-5. The cross-entropy loss function and Adam [109] with mini-batches are used to optimize the model. The final score is calculated by taking the expectation of the softmax outputs.

Table 4.3 shows Pearson Correlation between the gold and predicted scores. The first row denotes the classification quality with using GloVe-only. It turns out that this simple representation is sufficient to provides a reasonably good result (Pearson Correlation of 0.71). Using FEEL alone does not lead to the optimal results ($Predicate + Args + S$ has the best result among the variants, which is 0.68). This is because when constructing FEEL, some input components, like noun modifiers, are not considered. This simplifies model training by modeling only high-level event concepts while incurs losing some details. However, this does help FEEL captures more “structured” event semantics. The third row of the table indicates the result of using both GloVe and FEEL together, by concatenating them to form the input representations. The results are far better than the previous two representations and go up to 0.77.

We also observe that while animacy information was very useful, the sentiment information generally does not provide additional information for this task. This might be due to the word embedding already capturing this information. Also, since many examples in this dataset have neutral sentiment this contribution of this property is likely to be small. This is not always the case, as we can see in our next task.

4.4.5 Implicit Discourse Sense Classification (IDSC)

Our final evaluation task is the CoNLL 2016 Shared Task [72] on discourse parsing. The original goal is twofold: (1) locate discourse connectives and their arguments and (2) classify discourse senses. In this evaluation, we focus on the highly challenging IDSC task. Unlike explicit discourse senses, where discourse connectives, e.g., *because* and *however*, provide strong cues that help identify the correct sense, implicit discourse senses can only be inferred from the two given argument spans, and as a result relies heavily on modeling the semantic relationships.

The dataset used is based on the Penn Discourse Tree Bank (PDTB) [12], where all the arguments, connectives, and senses are annotated. We used the same settings as the CoNLL shared task: the data splits include training, development, test, and blind test sets; four relation types (non-explicit) and fifteen valid sense classes are used. More detailed information can be found in [12].

Similar to the SICK task, GloVe and FEEL are evaluated together. We use the same method for building the sentence (argument span) representations. The two span representations are concatenated to form the input example to a multi-class classifier. It is a two-hidden-layer neural network, where the activation functions are Rectified Linear Unit (ReLU) and the objective function is the cross-entropy loss. Adam [109] with mini-batches is used for optimizing the parameters.

Table 4.4 shows the micro average F1 scores across all the senses. Like SICK task, GloVe performs reasonably well here, as it captures general semantics at the word level.

We can observe the performance improvement obtained by adding event properties introduced by FEEL to GloVe. Specifically, *GloVe+PredDep+Args+S+A* performs the best over

Table 4.4. Micro average F1 scores across all the discourse senses under the setting of CONLL 2016 shared task.

Micro Average F1	Test	Blind Test
GloVe	0.2982	0.2815
GloVe+PredDep	0.2921	0.2886
GloVe+PredDep+Args	0.2983	0.2862
GloVe+PredDep+Args+S	0.2996	0.3102
GloVe+PredDep+Args+A	0.3063	**0.3111**
GloVe+PredDep+Args+S+A	**0.3174**	**0.3111**

both the test and blind test sets. This suggests that the benefit of using specific properties to enrich the event representation is task-specific.

4.5 Conclusion

In this work, we present a feature-enriched script learning model, FEEL. Our multi-task model learns robust event embeddings by jointly conditioning the event representation on neighboring events and the inner properties of the event. Our architecture provides a framework for injecting world knowledge and relevant contextual information needed to accurately represent events. This highly flexible approach can easily be adapted to the specific needs of different end applications. We specifically looked into two event properties—the sentence-level sentiment and the animacy information of the event protagonist.

The trained event embeddings are evaluated on three intrinsic tasks, including two newly proposed tasks highlighting the importance of evaluating narrative inferences. We also evaluate our model over two extrinsic tasks, by using it as the input representation. Our results show that FEEL can indeed utilize event properties and better account of the structured event semantics.

5. MULTI-RELATIONAL SCRIPT LEARNING

"Like a tree that grows stronger with more branches and roots, you need to find more and more ways to be inspired."

—Yiannis Kouros

The last chapter focuses on enhancing event representations by injecting event properties. This chapter has a broader focus on characterizing relationships between events. Previous script learning approaches embed event representations, based on their observed co-occurrence, such that their relationships are indicated by the similarity in the embeddings. While intuitive, these approaches fall short of representing nuanced relation types needed for downstream tasks. In this work, we suggest to view learning event embeddings as a multi-relational problem, which allows us to capture different aspects of event pairs. We model a rich set of event relations, such as *Cause* and *Contrast*, derived from the Penn Discourse Tree Bank (PDTB). We evaluate our model on three types of tasks, the popular Mutliple-Choice Narrative Cloze and its variants, several multi-relational prediction tasks, and a related downstream task—implicit discourse sense classification.

5.1 Introduction

Representing world knowledge that can be used for commonsense reasoning is a long-standing AI goal. *Scripts* [27] are structured knowledge representations capturing the relationships between prototypical event sequences and their participants in a given scenario. For example, given the event "*John shot Jim with a gun*", we can infer that "*he got arrested by police*" is more probable than "*he fell asleep*".

In recent years, the problem of extracting script knowledge from text has attracted significant attention. Early works [1] focused on symbolic event representations and used Pointwise Mutual Information (PMI) between events to capture their relationships. Recent works [2], [13], [30], [32], [52] represent events using dense vectors, based on event co-occurrence, and use vector similarity over their embeddings to measure their relationship.

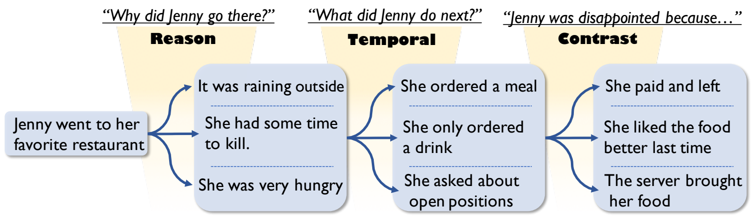

"Why did Jenny go there?"
Reason
"What did Jenny do next?"
Temporal
"Jenny was disappointed because…"
Contrast
Jenny went to her favorite restaurant
It was raining outside
She had some time to kill.
She was very hungry
She ordered a meal
She only ordered a drink
She asked about open positions
She paid and left
She liked the food better last time
The server brought her food

Figure 5.1. Multi-relational commonsense inference requires different relation types, beyond similarity.

Our main observation in this work is that while models for learning script knowledge improved significantly over the last decade, these models can essentially represent only a single event relationship, co-occurrence. That is, events appearing in similar contexts tend to have similar representations. Although this idea works well for a lot of NLP tasks, it is too coarse for modeling commonsense, which should account for fine-grained relationships. To better understand this, consider the example described in Figure 5.1. Given the first event, corresponding to the sentence "*Jenny went to her favorite restaurant.*", called Step 1, any of the following events in Step 2 would be highly related, and thus similar, to the input event. That is, "*It was raining outside*" and "*She was very hungry*" are both possible NEXT events. Using event similarity alone is too coarse to support many relevant inferences. However, if the relation between the events is given, more clues can be applied to support reliable inferences. In Figure 5.1, given Step 2 is a *Reason* to Step 1, analogous to asking the question "*Why did Jenny go there?*", the event "*She was very hungry*" is clearly a more reasonable choice. Therefore, using event similarity alone is too coarse to support many relevant inferences, i.e., capturing the *Reason* for the event, should produce a different set of relevant events, compared to *Temporal* (next) events

To help prioritize between showing diverse types of event relations and providing a framework for this discussion, we focus on a set of discourse relations, introduced by Penn Discourse Tree Bank (PDTB) [12]. Traditional script learning models would fall short of making the inferences here. For example, the last inference step in Figure 5.1 asks for an event that *Contrasts* with the previous step. Based on human commonsense, we can identify that the most probable scenario is "She ordered a meal **but** she liked the food better last time." Modeling the relation type helps us capture different expectations about subsequent events.

We use the fact that these relations are often indicated by discourse markers (e.g., "but", capturing the contrasting relation) to extract supervision for learning these relations.

Our goal in this work is to support such inferences. We introduce a multi-relational event embedding approach, which generalizes the notion of event embedding, by allowing it to capture multiple fine-grained relationships. Our approach builds on recent translation-based embeddings [110], [111], originally introduced in the context of knowledge graph completion. We adapt these methods to the textual inputs, and suggest a compositional neural network used for capturing the event's internal linguistic structure, while using the translation-based embedding objective to capture different relationships between events. We include 11 relation types, capturing the progression of the narrative: COREF_NEXT, the next event in the coreference chain; NEXT, the next event that occurs subsequently in text; and 9 discourse relations, collectively refer to as DISCOURSE_NEXT.

We evaluate our model in three settings. In the first, we evaluate it on a common benchmark, Multiple-Choice Narrative Cloze (MCNC) task [2], and its sequential variants proposed by [13]. We show that we can outperform previously published work by a large margin. In the second setting, we further examine our model's characteristics on three intrinsic tasks. In the last setting, we conduct a challenging downstream task—implicit discourse sense classifications, examplifying the model's applicability.

5.2 Related Work

Statistical Script Learning was popularized by [1], framing the problem as an unsupervised learning problem, using a PMI-based learning model to approximate a conditional probability of event occurrence. Recent approaches build on representation learning techniques, by learning event embeddings with neural networks. [2] utilized Skip-Gram [9] and an event compositional neural network to adjust event representations. [30], [50] applied a LSTM Recurrent Neural Network (RNN), coupled with Beam Search, to model event sequences and their representations. [23] used three-dimensional tensor-based networks to construct the event representations. [13] trained the event embedding with additional features in a hierarchical architecture. [52] constructed an event graph and utilized its network information to make script event predictions. In this work we combine GRU [112], for encoding fine-grained argument information, with a compositional network to generate event representations. GRU was shown to be a competitive alternative to LSTM while requiring less parameters [113], [114].

Modeling multi-relational data was originally explored for Knowledge Graph Completion, typically focusing on a family of translation-based embedding models which view relations as translations in the vector space. For example, *TransE* [110], captures the relation between h, t, r (embedding of arg0, arg1, and relation), by minimizing the distance between $h + r$ and t. *TransH* [115] and *TransR* [111] projects the entities into relation-specific spaces. Recent models address issues, such as maintaining structures [116], [117] and capturing richer interactions [118]. In this work, we adapt *TransE* and *TransR* for narrative script learning, which is an innovative generalization of relation embedding for commonsense inference.

Several recent works looked at modeling specific relationships between events and extracting commonsense knowledge. [49] explored modeling cause-effect relations between events; [17] focused on If-Then relations and showed that their joint multi-task model outperforms the models trained in isolation, based on human evaluations. [119] utilized discourse markers to extract relations between semantic frames and modeled them with prevalent language models. Event2Mind [43] created a dataset capturing the relationship between an event description and its participants' intent and emotional reaction. This idea is related to our work, as the intent and reaction can correspond to *Reason* and *Result* discourse relations in our case. Our goal in this work is to present a relational generalization over such relationships using a shared embedding space.

The Narrative Cloze (NC) task [1] was introduced to evaluate statistical script models by removing an event from a chain, and observing the ranking of the correct answer over the entire event vocabulary, given the rest of the chain. However when complex event structures were considered, e.g., multi-argument events [21], the large vocabulary size introduced both computational issues and ambiguity into the evaluation. As a result, [2] proposed a multiple-choice variation, called MCNC. It simplifies the evaluation process and reduces its computational burden. A similar choice of the multiple-choice adaptation could also be found in recent works, such as Story Cloze [34] and SWAG [58]. In this work, we evaluate our models on MCNC, and two recent variants [13] turning MCNC into a sequential inference task. We also introduce relation-specific evaluation capturing the ability of our model to account for nuanced relations beyond co-occurrence.

5.3 Model

We propose a learning framework, which accounts for the internal predicate-argument structure of events, tuning it to respect different relation types.

5.3.1 Overview

Our framework has two preprocessing phases: Event Extraction and Relational Triplet Extraction. In Event Extraction, we aim to identify events from free-form text. The process builds on a dependency parser and coreference resolution. Once events are extracted, we address their relations, specifically three types: (1) events with coreferent entities, (2) events located near each other, and, more importantly, (3) events connected with discourse relations.

The model uses entity-based event definition, as described in Chapter 2. We abbreviate the event notation e^{e} to e in this chapter. The output of the preprocessing phases is a set of relation triplets $(\mathrm{e}_h, \mathrm{e}_t, r)$, where e_h and e_t are head and tail events, and r is their relation type. We then feed them to a neural network for learning event and relation embeddings. The network objective is an energy function $g(\mathrm{e}_t, \mathrm{e}_h, r)$, which can be used to approximate the conditional probabilities $p(\mathrm{e}_t|\mathrm{e}_h, r)$ or $p(r|\mathrm{e}_t, \mathrm{e}_h)$. This objective captures commonsense knowledge expressed in event relations and embeds it in a vector space, which can be utilized in downstream tasks. Two model variants are proposed in this work. The first model, *EventTransE*, assumes that all the relations are in the same embedding space and jointly learns representations for events and relations. It works well in some cases, though it might not be expressive enough in others. The second model, *EventTransR*, addresses this issue by introducing relation-specific parameters, which project events into relation-specific spaces when measuring their relatedness.

5.3.2 Event Extraction

We construct a preprocessing pipeline to extract events and relations over a large text collection. Each event e consists of three components: predicate ($pred$(e)), subject (subj(e)), and object (obj(e)). Due to computational considerations we restrict the event representation to two arguments. We use a special empty argument representation, *NONE*, for events that have fewer arguments. To obtain the event representation from text, we first run a dependency parser and coreference resolution [1] to acquire the needed information.

Events are extracted by connecting entity mentions on the coreference chain with their corresponding predicate and additional argument, based on the dependency tree. E.g., given, "*Jenny went into her favorite restaurant,*" we extract *(go_into, jenny, her favorite restaurant).*

Unlike the previous works [2], [13], [30], which only consider headwords of entity mentions, we use complete mention spans. In our running example, we consider the object as "*her favorite restaurant*", rather than just "*restaurant*". This allows the models to capture the nuanced information relevant for many commonsense inferences, such as the "favorite" here.

Other details of the event preprocessing follow [13]'s work and are summarized below:

- Predicates are lemmatized and in lower-case.
- Predicates are not only verbs but also predicative adjectives. For instance, "Jenny was hungry. She ordered a big meal." The adjective "hungry" is an event predicate denoting the entity's state.
- Negations should be applied to predicates, e.g., "She didn't eat dinner," results in a new predicate: "not_eat."
- Particles and clausal complements (xcomp) are included in verb predicates, since verbs, such as "go" and "have" are not strong enough to give meaningful information. For instance, in "He went shopping last night," the predicate is "go_shop," rather than "go."

[1]Stanford CoreNLP [101].

Table 5.1. DISCOURSE_NEXT relations from PDTB and the number of relations extracted for training.

DISCOURSE_NEXT Complete	Abbrev.	#relations.
Comparison.Contrast	**Contrast**	7334K
Contingency.Cause.Reason	**Reason**	2818K
Contingency.Cause.Result	**Result**	228K
Contingency.Condition	**Cond.**	3745K
Expansion.Restatement	**Restat.**	16K
Expansion.Conjunction	**Conj.**	98K
Expansion.Instantiation	**Instan.**	249K
Temporal.Synchrony	**Sync.**	63K
Temporal.Asynchronous	**Async.**	379K

- Low-frequency predicates and words in the entity mentions are considered as Out-Of-Vocabulry (OOV) during training. As the vocabulary size is related to memory limitation and rare words are highly likely to introduce noise, only the most active n_{pred} predicates and $n_{argword}$ argument words are considered.

- For the same reason given above, the maximum entity mention lengths, l_{subj} and l_{obj}, are set.

5.3.3 Relational Triplet Extraction

Relations are expressed as triplets $(\mathrm{e}_h, \mathrm{e}_t, r)$, where r is the relation type, and e_h and e_t are events that have an internal structure of $(pred(\mathrm{e}), subj(\mathrm{e}), obj(\mathrm{e}))$. 11 types of relations are considered in this work for demonstrations: COREF_NEXT, NEXT, and 9 discourse relations, which collectively refer to as DISCOURSE_NEXT.

COREF_NEXT captures sequential relationships between events on the same coreference chain. The NEXT relation is defined between events pairs that co-occurr in a fixed-sized ($w_{context}$) context window. It aims to capture related events that do not share arguments. For example, in "*The forest was on fire. Trees burned.*", the two events do not share arguments, but they often co-occur, and thus are related. Previous works about script learning [2], [13], [30], [32], [52] use either COREF_NEXT or NEXT independently, which failed to leverage the shared information.

For DISCOURSE_NEXT, 9 discourse relations, taken from PDTB, are denoted in Table 5.1. These relations correspond to commonsense judgments. For example, we can do causal inference with the *Reason* and *Result*; or we can identify the juxtaposition between events by utilizing *Contrast*.

The discourse relations can be represented with a relation type and a pair of argument spans [72]. For example, "*Jenny went to a restaurant, because she was hungry*" has a relation *Reason* and the spans are the two clauses (omitting the connective "because"). Since training event embedding requires significantly more data than annotated in the PDTB corpus, we approximate this by building a rule-based annotator. We first identify explicit discourse connectives, such as "because," and assume that the surrounding clauses are their argument spans. To determine the relation type, we map the connectives to their most probable type based on the PDTB data. To mitigate the noise, we only take connectives that are highly indicative of their type (85% of connective occurrences are of that type). Note that in our setup a given pair of events might have up to three relations annotated: a discourse relation, NEXT, and COREF_NEXT. While our weakly supervised relation extraction is noisy, we demonstrate empirically its ability to capture these relations.

We create negative examples by corrupting the positive triplets, randomly replacing e_h, e_t, or r with an event or relation. For each positive triplet we sample one negative triplet. The events are sampled from event vocabulary, collected from the training set, and the relations are from the 11 types we support. We have experimented with different negative sampling strategies, such as corrupting the tail event only or sampling with different event distributions. None of them perform better.

5.3.4 Compositional Event Representation

Figure 5.2 shows the architecture of our models. Each event e has a raw representation $(pred(\mathrm{e}), subj(\mathrm{e}), obj(\mathrm{e}))$. The predicate $pred(\mathrm{e})$ is given in an embedding lookup table, a matrix with size $|P| \times d_a$, where P denotes predicate vocabulary. $subj(\mathrm{e})$ and $obj(\mathrm{e})$ are encoded with two separate Bi-GRUs [112]. We call them subject encoder and object encoder, as shown in the figure. The outputs of the encoders are d_a-dimensional respectively. Each GRU is defined as follows:

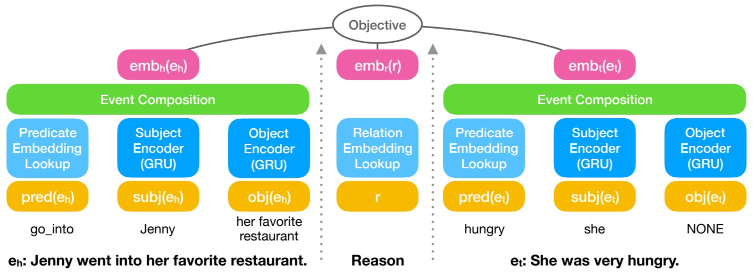


Figure 5.2. Multi-Relational Script Learning Architecture: the left and right networks encode the event embeddings for e_h and e_t; the middle part encodes the relation r. The training objective on top jointly learn these embeddings.

$$
\begin{aligned}
z_t &= \sigma(W^{(z)}x_t + U^{(z)}h_{t-1}) \\
r_t &= \sigma(W^{(r)}x_t + U^{(r)}h_{t-1}) \\
\bar{h}_t &= tanh(Wx_t + r_t \odot Uh_{t-1}) \\
\vec{h}_t &= z_t \odot \vec{h}_{t-1} + (1 - z_t) \odot \bar{h}_t,
\end{aligned}
$$

where x_t is the input token at timestamp t; $W^{(z)}, U^{(z)}, W^{(r)}, U^{(r)}$, W, U are parameters to be trained; $\vec{h}_t \in R^{\frac{d_r}{2}}$ is the hidden memory at timestamp t; z_t and r_t are update and reset gates for controlling purposes. The final argument representation is the concatenation of GRU hidden representations trained in two directions, i.e., $h_t = [\vec{h}_t, \overleftarrow{h}_t]$.

The encoded representations for each event component are then fed into a Event Composition network. The network is fully-connected and has one hidden layer, defined as follows:

$$
\begin{aligned}
h_1 &= relu(W_1 x_\mathrm{e} + b_1) \\
v &= W_2 h_1 + b_2,
\end{aligned}
$$

where x_e is the concatenation of the encoded predicate, subject, and object; $W_1 \in R^{d_h \times 3d_a}, b_1 \in R^{d_h}, W_2 \in R^{d_r \times d_h}, b_2 \in R^{d_r}$ are model parameters. The output $v \in R^{d_r}$ is the event embedding.

For the relations, we embed them using another embedding lookup table. The table size is $n_{rel} \times d_r$, where n_{rel} is the number of relation types. In our case, $n_{rel} = 11$.

5.3.5 Model: EventTransE

EventTransE is an event embedding model inspired by *TransE* [110]. The idea is to embed nodes and their relations in the same vector space so that the distance between nodes reflects their relations. This is called translating operations in the original paper. Based on this idea, we explore a new possibility of learning event embeddings that can make inferences conditioned on different relations. We connect the *TransE* objective to the previous compositional network outputs, which can be formulated as follows:

$$\begin{aligned} g_{transe}(t) &= g_{transe}((\mathrm{e}_h, \mathrm{e}_t, r)) \\ &= \|\mathrm{e}_h + r - \mathrm{e}_t\|_p^p, \end{aligned} \tag{5.1}$$

where $\mathrm{e}_h, \mathrm{e}_t, r \in R^{d_r}$ are the embeddings from the Event Composition network. Note that Equation 5.3.5 is a dissimilarity measure. Lower scores mean that the given two events are strongly related.

5.3.6 Model: EventTransR

A known issue of *EvenTransE* is its limited ability to deal with reflexive, 1-to-N, N-to-1, or N-to-N relations [115]. Consider a simple example illustrating the problem: given Equation (5.3.5), it is possible to learn a zero relation vector r and two arbitrary but identical event representations e_h and e_t, which minimize the loss. *EventTransR* is proposed to address these issues by separating the event and relation spaces as *TransR* [111]. It introduces relation-specific parameters to model the interactions between the spaces. *EventTransR* is defined as follows:

$$\begin{aligned} g_{transr}(t) &= g_{transr}((\mathrm{e}_h, \mathrm{e}_t, r)) \\ &= \|\mathrm{e}_h M_r + r - \mathrm{e}_t M_r\|_p^p, \end{aligned} \tag{5.2}$$

where $r \in R^{d_r}$, $\mathrm{e}_h, \mathrm{e}_t \in R^{d_\mathrm{e}}$ are the input embeddings, and $M_r \in R^{d_\mathrm{e} \times d_r}$ is the relation-specific parameters introduced.

5.3.7 Training Objective

The objective is the Margin-Based Ranking Loss:

$$\mathscr{L}_p(t) = \sum_{t \in T} \sum_{t^* \in T^*} max(0, \delta + g(t) - g(t^*)), \tag{5.3}$$

where T is the set of positive relational triplets; T^* is the set of corrupted relational triplets; δ is the margin, and $g \in \{g_{transe}, g_{transr}\}$. At test time, we can leverage the dissimilarity measures to either predict the tail event given the head event and relation, or predict the relation given the head and tail events:

$$\hat{\mathrm{e}}_t = \arg\min_{\mathrm{e}^* \in E} g(\mathrm{e}_h, \mathrm{e}^*, r);$$
$$\hat{r} = \arg\min_{r^* \in R} g(\mathrm{e}_h, \mathrm{e}_t, r^*).$$

E and R are the event and relation vocabulary.

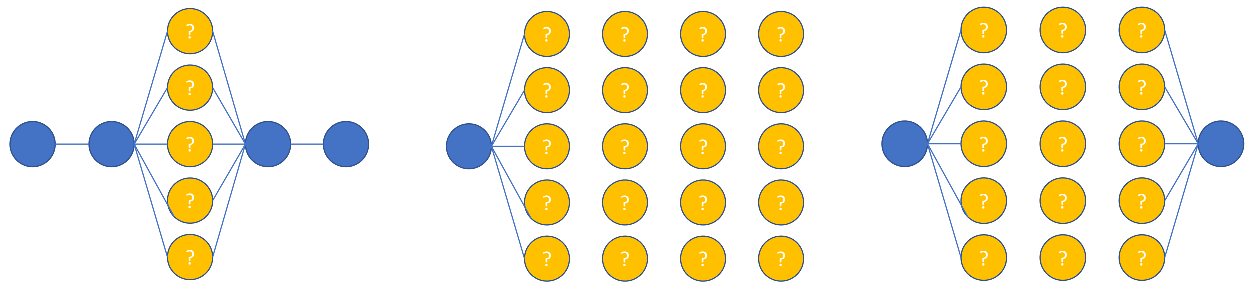
5.4 Experiments

We divided our experimental evaluation into three parts. The first focuses on comparing our models with previous work on several common script learning evaluation tasks. The second evaluates our model's ability to capture different relation types between events. In the third, we apply our models to a related downstream task, implicit discourse sense classification, and achieve competitive results by combining our event embeddings with ELMo [120], a contextualized word embedding model. We also provide an additional qualitative analysis, showing inferences made by our model.

For training, we use the New York Times (NYT) section of the English Gigaword [104]. It contains 2M newswire articles and splits into train/dev/test sets, replicating the setup given by [2]. 500M triplets are extracted from the training set. We tested different sets of hyperparameters, and came up with the following setting: the number of active predicates and argument words, n_{pred} and $n_{argword}$, are both set to 25000; the maximum argument lengths, l_{subj} and l_{obj}, are set to 15; the event contextual window size $w_{context}$ for extracting NEXT relation is 5; the event composition hidden layer has the dimension $d_h = 1000$; Rectified Linear Unit (ReLU) [121] is used as the activation function; embedding dimensions $d_a = 500, d_e = 500$, and $d_r = 500$; the margin δ is empirically set to 1; the optimizer is Adagrad [122] with initial learning rate 0.01; the word embeddings for entity mention encoders are initialized as the word embeddings pre-trained in Skip-Thoughts [113]. All the experimental results are averaged over 5 runs. The source code and pre-trained models are publicly available [2].

5.4.1 Multiple Choice Narrative Cloze Tasks

We begin by evaluating our model on three event representation tasks: Multiple-Choice Narrative Cloze (MCNC), Multiple-Choice Narrative Sequence (MCNS), and Multiple-Choice Narrative Explanation (MCNE). MCNC, proposed by [2], measures script learning models' ability to predict a missing event, given its context, in a multiple-choice setting. This evaluation task is not perfect, as noise would be introduced by automatic extraction tools, but not so common as to invalidate the results, and thus this evaluation is widely accepted. [13] generalized this single-step task, and suggested two sequence inference versions—MCNS and MCNE. Figure 5.3 explains the three tasks. Given an event chain, MCNC chooses one step as a multiple-choice question and generates four negative choices for that step. MCNS turns it into a sequence prediction problem by creating multiple-choice questions for each step, except the start event. MCNE provides an additional clue, which is the end event. The inference model has to connect the start and end by explaining things happened in between.



(a) MCNC (b) MCNS (c) MCNE

Figure 5.3. Comparing single-step prediction (MCNC) and multiple-step inference (MCNS and MCNE).

[2]https://github.com/doug919/multi_relational_script_learning

Following the setup in [2], we evaluate on top of coreferenced event chains, where a protagonist participates each event. The minimum length of the event chains is 9, as short chains are likely to be caused by parsing errors. Our models naturally score the candidates with our training objective $g \in \{g_{trane}, g_{transr}\}$ using Coref_Next relation, while other baselines use cosine similarity.

Multiple-Choice Narrative Cloze

We compared two versions of our models, using the entire argument span, or just its headword, with several recently published results.

We compare our models with the following baselines on the MCNC:

- **Random** uniformly selects a candidate.
- **PPMI** [1] uses co-occurrence information and calculates Positive PMI for event pairs.
- **BiGram** [33] calculates bi-gram conditional probabilities $P(\text{e2}|\text{e1})$ based on event term frequencies.
- **Word2Vec** [9] refers to the pre-trained word embeddings from Word2Vec SkipGram. The summation of word embeddings of predicates and argument mentions are used to represent events.
- **EvSkipGram** [2] uses SkipGram to learn representations from "sentences" formed by predicates and argument headwords.
- **EventComp** [2] uses a neural network to learn a compositional function for EvSkipGram and outputs a coherence score for event pairs.
- **SGNN** [52] is a graph-based model specifically designed for MCNC. It considers each event chain as a sub-graph, and feed it into their GRU-based recurrent networks, which outputs relatedness scores for the candidates.
- **FEEL** [13] is an event embedding model that does multi-task learning for inter-event relations and intra-event features.
- **PairLSTM** [32] is an event embedding model that considers event order information and uses a LSTM network's hidden states for event representations.

Table 5.2. Accuracy scores (%) of MCNC. **-headword** stands for using headwords only in argument mentions. The star sign (*) denotes that the results are based on the newly sampled evaluation set.

Methods	**Accuracy**
Random*	20.00
PPMI [1]	30.52
BiGram [33]	29.67
Word2Vec* [9]	37.39
EvSkipGram* [2]	46.28
EventComp [2]	49.57
FEEL* [13]	51.62
SGNN [52]	52.45
PairLSTM [32]	55.12
EventTransE-headword*	60.50
EventTransR-headword*	59.38
EventTransE*	**63.67**
EventTransR*	62.86

Since we need the complete argument spans for events, which is not available in [2]'s pre-processing procedure, we re-implement the event extraction step by carefully following their procedure. We mark the results based on the newly sampled evaluation set with a star sign (*). We released the newly sampled evaluation set for future comparisons. Table 5.2 shows the results.

Our models outperform the best baseline model for more than 7% absolute accuracy score. We attribute the improvement to three factors: (1) our models encode complete argument mentions rather than just headwords, *EventTranseE-headword* and *EventTransR-headword*, which are our models' variants that use only headwords for arguments, show that about 3% of the improvement is from this; (2) our models have shared event representations over multiple relations, which regularize the representations in diverse aspects, while other baselines do not make use of relations other than COREF_NEXT. (3) our models' training objective directly measures relation-specific dissimilarity between events, while most others are based on simple cosine similarity.

Table 5.3. Acc (%) of MCNS (NS) and MCNE (NE) tasks. **{NS,NE}-V** use Viterbi for inference. **Base-Inf** is a local greedy model using the previous prediction only, and **Skyline-Inf** is given gold contextual events when calculating transition probabilities.

Methods	NS-V	Base-Inf	Sky-Inf	NE-V
GloVe	29.38	27.60	38.50	31.29
FEEL	41.60	38.50	46.00	44.80
EventTransE	**59.48**	**51.22**	**64.47**	**60.94**
EventTransR	58.66	50.73	63.65	60.00

Multiple-Choice Narrative Sequence

The MCNC looks at a single transition between events; however, it does not capture the flow of the entire narrative. [13] proposed MCNS, which instead of sampling candidate options for one event, it samples options for all the events on the chain, except the first event which is used as the starting point for predictions (Figure 5.3b). Based on the dissimilarity scores calculated by our models, we can compute transition probabilities for each step. Then we can find the most likely sequence using Viterbi inference algorithm [105]. We follow the evaluation setting used in [13] and compare three decision models: (1) *Viterbi*, which finds the most probable sequence of predictions; (2) *Baseline-Inf*, which greedily picks the best transition at each step based on the previous prediction; (3) *Skyline-Inf*, which breaks down a sequence of decisions into local decisions, each using the gold states of all the contextual events.

Table 5.3 shows the results. Our models outperform FEEL [13], who introduced the task. The same set of reasons given in the section MCNC explain the improvement. We also note that *EventTransE* is especially strong in making predictions for COREF_NEXT.

Multiple-Choice Narrative Explanation

MCNE is another extension to MCNC. Essentially, in addition to the first event, the final event is also given (Figure 5.3c). Intuitively, the goal of this evaluation task is to capture explanations, consisting of event sequences, that connect the start and end points. The same inference algorithms as MCNS are adopted. The right three columns of Table 5.3 gives the result (Note that the *Baseline-Inf* and *Skyline-Inf* are shared with MCNS). The result shows a similar trend as MCNS, but with higher scores, due to the additional information brought by the last event. Note that when calculating the accuracy, we only consider the event blanks in the middle (ignoring the last prediction made in MCNS) for both MCNS and MCNE. This ensures a fair comparison.

5.4.2 Intrinsic Discourse Relations Evaluation

We suggest three intrinsic tasks, depicted in Figure 5.4, evaluating how multi-relational information is captured. Given a triplet $(\text{e}1, \text{e}2, r)$: (1) predict the next event e2, (2) predict the relation r, and (3) predict its correctness (triplet classification).

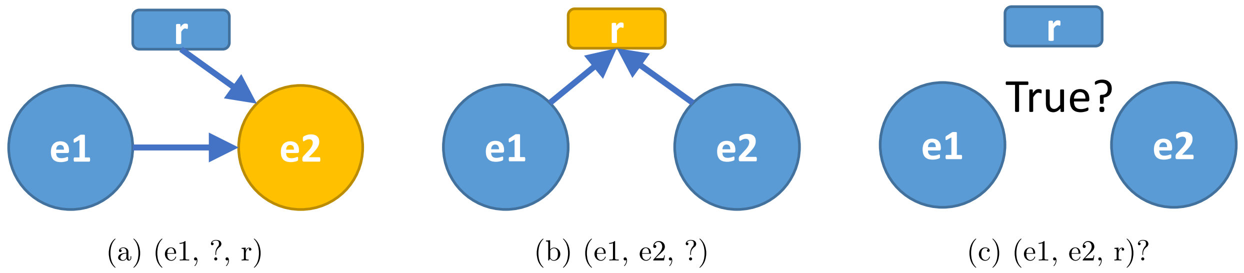


Figure 5.4. Three intrinsic tasks for evaluating our models: (5.4a) predicts the next event given an event and a relation; (5.4b) predicts the relation given a pair of events; (5.4c) binary classification for triplets.

Predict the Next Event

Similar in spirit to the setup described in Fig. 5.1, we ask whether knowing the relation, connecting the head to the tail event, would change the expectation about the tail event.

Given a set of triplets that have discourse relations, for each triplet, we corrupt e_t and sample four extra negative choices to form a multiple-choice question. We compare our model variants with a strong baseline model—ELMo [120]. ELMo is a context-aware word embedding model that has shown strong performance in language understanding tasks. To get the contextualized word embeddings, we have to provide the context, usually the sentence where the target words appear. To retrieve the context, for each event e, we re-construct its "sentence" by concatenating its *subj*(e), *pred*(e), and *obj*(e). The averaged word embedding

Table 5.4. Accuracy scores (%) of the next event prediction, given an event and a relation. **ELMo** is a contextualized word embedding model. **-Random** and **-NEXT** are our model variants that replace the given relation with a random and NEXT relation respectively.

Methods	**Accuracy (%)**
Random	20.00
ELMo [120]	42.97
EventTransE-Random	52.78
EventTransR-Random	26.11
EventTransE-NEXT	51.80
EventTransR-NEXT	51.71
EventTransE	54.83
EventTransR	**55.08**

of the context is used to represent the event. ELMo predicts the next event based on cosine similarity, disregarding the relation. We also make two variants to show our models' awareness to relation types. One replaces the correct discourse relation with a random relation; the other replaces it with a NEXT relation.

Table 5.4 shows the results. We can see that all our model variants outperform the ELMo baseline, as our models are aware of the relation between events. Similarity-based models that can capture frequently co-occurred events fail to consider the nuanced relations. *EventTransR* performs the best as it has relation-specific parameters emphasizing the relational nuances. Interestingly, using **-NEXT** relation only is also very indicative for predicting the next event, which explains why previous works failed to address the nuanced relations. The results for **-Random** relations indicates that *EventTranR* is very sensitive to incorrect relations. This is due to the separation between the relation and event embedding spaces, useful for relation-sensitive tasks. Also, that **EventTranE-Random** model works better than **EventTranE-NEXT** suggests that our models with discourse relations do capture their fine-grained differences. Note that even *EventTranR* with scrambled relations outperform ELMo with a large margin. We hypothesize that ELMo emphesizes similarity rather than nuanced discourse relations between sentences.

Table 5.5. Predicting relation type given two events, a 9-class classification task. **F1** is micro averaged.

Methods	**Accuracy**	**F1**	**MRR**	**Recall@4**
Random	11.11	-	-	-
EventTransE	49.93	50.00	70.05	**83.05**
EventTransR	**50.84**	**51.00**	**70.62**	81.65

Predict the Relation

We predict the correct relation out of the 9 discourse relations (Table 5.1), given two events. Table 5.5 shows the result. With additional relation-specific parameters introduced, *EventTransR* performs better than *EventTransE*. Note that the ability to rank the correct relation is also important as there might be more than one possible next events. According to the MRR and Recall@4, both models are competitive.

Triplet Classification

Table 5.6. Triplet Classification for discourse relations.

	ELMo		**EventTransE**		**EventTransR**	
	Acc.	**F1**	**Acc.**	**F1**	**Acc.**	**F1**
Reason	60.82	59.02	70.04	69.65	69.35	67.46
Contrast	63.04	58.69	73.09	73.22	74.07	74.90
Cond.	61.92	60.48	71.16	71.22	71.46	72.25
Conj.	65.56	66.01	66.86	68.32	65.42	65.58
Result	60.68	60.84	73.26	73.41	72.69	72.98
Async.	61.25	60.59	71.25	69.61	73.22	**74.27**
Sync.	64.71	62.96	70.20	69.60	72.44	72.49
Instan.	62.64	58.84	**74.80**	**74.39**	73.30	71.83
Restat.	62.86	61.41	74.68	73.35	**75.37**	73.70
Average	62.61	60.98	71.70	71.42	**71.92**	**71.72**

This task is inspired by Triplet Classifications in Knowledge Graph Completion [115], [123]. It predicts whether a given triplet $(\mathrm{e}_h, \mathrm{e}_t, r)$ is valid or not. We sample positive triplets from our dev and test splits and negative triplets by corrupting e_t. We use the dev split to develop a set of relation-specific thresholds λ_r. The score is calculated using $g \in$

$\{g_{trane}, g_{transr}\}$. If the score is lower than λ_r, the triplet is classified as positive; otherwise, it is negative. We sample 500 positive and negative triplets for each relation. The ELMo baseline is similar to previous experiments. We also develop a set of relation-specific thresholds based on ELMo's similarity scores to make predictions. Table 5.6 summarizes the results and shows that the similarity-based model, ELMo, cannot represent the nuanced relations information as good as our model. Interestingly, both our models excelled at predicting the *Expansion* relations (Instant. and Restat.). *EventTransR* get high scores on *Temporal* relations (Async.) which implies its applicability on tasks like event order inference [25]. In general, for tasks requiring nuanced relations, *EventTransR* works better; if we only need to know the NEXT or COREF_NEXT events, *EventTransE* is better. In addition, *EventTransE* has less trainable parameters, converging way faster.

5.4.3 Implicit Discourse Sense Classifications

The final evaluation task is a subtask in CoNLL 2016 Shared Task [72] on implicit discourse sense classification. We follow the same setting as the shared task, with 15 sense classes. More details can be found in [72].

Three baselines, the best and median system of each subtask, are provided. In addition, we also trained a strong baseline based on ELMo. We first create word embeddings for words in the argument spans using ELMo and put an attention layer on top of the words. The attention layer weights the words and create the argument representation. We feed the representations of the two arguments to a neural classifier, where two fully-connected hidden layers with dimensions 256 and 128 are applied. ReLU [121] are used as activation functions and AdaGrad [122] is used for optimizing the parameters. We combine *EventTransE* with the ELMo baseline by having another attention layer on top of the event embeddings and concatenating all the argument representations in the network.

Table 5.7 shows the results. The ELMo baseline is highly competitive, comparable to the winners of the task (ecnucs and ttr). **ELMo+EventTransE**, which is our combined model, consistently contributes to performance, demonstrating the benefit of our model to downstream tasks.

Table 5.7. micro F1 scores (%) for Implicit Discourse Sense Classifications. Evaluated against the best and median systems in CoNLL'16, and ELMo contextualized word embedding with attention layers, which can be improved by incorporating our *EventTransE*.

Methods	**Dev**	**Test**	**Blind**
PurdueNLP [124]	38.05	34.45	29.10
ecnucs [125]	46.42	40.91	34.18
ttr [126]	40.32	36.13	37.67
ELMo	45.60	37.65	36.72
ELMo+EventTransE	46.81	39.05	38.35

5.4.4 Qualitative Analysis

Table 5.8. Small worlds for qualitative analysis: (a) murderer scenario; (b) stock market scenario. **NOA** stands for "No Argument."

World	**Possible Entities**	**Possible Predicates**	**#Candidate Events**
(a)	jim, john, a girl the officer, he, a nurse, a man, a pedestrian, the gun, NOA	safe, hit, stop, hate, gone, happy, angry, smile, change, survive, sad	1100
(b)	shares, money, CEO, the price, CTO, police, employees, a dog, a girl, NOA	strike, hire, sell, buy, happy, sad, angry, smile, make, survive, adjust, increase, good, decrease	1400

To comprehend our models better, we perform a qualitative analysis, which instantiates the exact inferences our models make. In this analysis, our models make inferences in grounded scenarios, where we have clearer expectations about possible events and outcomes. To do so, we create two confined "worlds," where each world only have limited numbers of entities and predicates, and hence a limited number of candidate events. This limitation is enforced as it helps examine quality of the inferences.

Table 5.9. Qualitative analysis for the world (a), which has 1100 number of possible event candidates. The first row, starting scenarios, gives the starting events and the 4 rows below show the inferences based on 4 given discourse relations. The resulting events that match our common sense are bolded and **NOA** stands for "No Argument."

	Triplets (a)	Interpretations (a)
Starting Scenarios	(shot, jim, john), (die, john, NOA), (arrest, police, him)	Jim shot John. Police arrested him. Jim died.
Inference Contrast	**(survive, john, jim)**, **(survive, john, NOA)**	**John survived Jim.** **John survived.**
Inference Result	**(sad, jim, NOA)**, (sad, john, jim)	**Jim was sad.** John sad Jim.
Inference Reason	(angry, NOA, john), **(angry, jim, NOA)**	___ angry John. **Jim was angry.**
Inference Async.	(survive, NOA, john), (survive, jim, john)	___ survived John. Jim survived John.

Table 5.10. Qualitative analysis for the world (b), which has 1400 number of possible event candidates. The notations are identical to Table 5.9's.

	Triplets (b)	Interpretations (b)
Starting Scenarios	(invest, company, fund), (have_soar, stock, NOA)	Company invested fund. The stock has soared.
Inference Contrast	(increase, the price, ceo), **(strike, the price, shares)**	The price increased CEO. **The price stroke shares.**
Inference Result	(increase, the price, ceo), **(make, ceo, shares)**, **(make, ceo, money)**	The price increases CEO. **CEO made shares.** **CEO made money.**
Inference Reason	**(increase, shares, NOA)**, (buy, employees, ceo)	**Shares increased.** Employees bought CEO.
Inference Async.	**(make, shares, ceo)**, **(make, money, ceo)**	**CEO made shares.** **CEO made money.**

Table 5.8 shows the entities and predicates that are selected for the two worlds. The topic of the world (a) is about a murderer and the topic of the world (b) is about stock markets. Both are common topics in newswire articles, which we use for training our models. Note that since each event triplet has two entity components (subject and object) and one predicate,

the number of candidate events is calculated as $n_{pred} \cdot n_{ent}^2$, where n_{pred} is number of predicates and n_{ent} is the number of entities. In these two worlds, we have 1100 and 1400 candidate events.

To conduct the inference, our model ranks the candidate events according to their relevance to the starting scenario, which contains a sequence of events. We use *EventTransR* to embed and rank all the events. For each candidate, we jointly consider its relevance to each starting event through a dissimilarity function based on *EventTransR*, which is defined as follows:

$$s(\mathrm{e}_c) = \sum_{\mathrm{e}_s \in S} f_{transr}(\mathrm{e}_s, \mathrm{e}_c, r), \ \ \forall \mathrm{e}_c \in C,$$

where S is all the events in the starting scenario, C is the set of possible candidate events, and r is the interested discourse relation. Based on the dissimilarity function, candidates with lower dissimilarity will be ranked higher. In addition, we consider four discourse relations—*Contrast*, *Reason*, *Result*, and *Asynchronous*—in this analysis, as they are commonly used in commonsense inferences.

Table 5.9 and 5.10 summarize the analysis. We only list the top 2-3 predicted events. In the world (a) (Table 5.9), *EventTransR* can accurately predict events for three, out of four, relations. In particular, we can contrast the fact that "John died" with "John survived," which has not been addressed in previous works. For *Asynchronous*, on which *EventTransR* fails, the signal for temporal relations is noisier, as many possible outcomes are reasonable. In the world (b) (Table 5.10), our model succeeds for all the four relations. Also, our model is able to tell the difference between *Result* and *Reason*, as indicated by the prediction that "the stock has soared" *leads to* "CEO made money," and "*Because* shares increased, the stock soared." They show that we are able to control the inferences over different discourse perspectives, which is useful for tasks like story generations.

This analysis helps provide more intuitions about the knowledge learned by our models. Note that this is a challenging task even when grounded with a small set of candidate events, as was reported by previous works that looked at event-ranking based evaluations [30], [43].

5.5 Summary

We consider the problem of learning relation-aware event embeddings for commonsense inference, which can account for different relations between events, beyond simple event similarity. We include several event relations, identifying, for example, the causes for them. We show that weak supervision, provided by a rule-based annotator is enough for training our models.

We evaluated and compared two models, *EventTransE* and *EventTransR*, on several narrative cloze and relation-specific tasks, and showed the learned embedding can capture relation-specific information as well as improve performance for a downstream task.

This work lays the foundation for reasoning over narratives and explaining how sentences combine to form them. In the future we would like to expand this direction, and find ways to connect event and relation representation, learning and inference in a unified framework.

6. CONTEXTUALIZING EVENT REPRESENTATIONS WITH A NARRATIVE GRAPH

"All truth passes through three stages. First it is ridiculed, then it is violently opposed, then it is accepted as self-evident."

—Arthur Schopenhauer (1788–1860)

This chapter introduces the third technique we create for advancing script learning. Representing, and reasoning over, *long* narratives requires models that can deal with complex event structures connected through multiple relationship types. This work suggests to represent this type of information as a narrative graph and learn contextualized event representations over it using a graph neural network model. We train our model to capture event relations, derived from the Penn Discourse Tree Bank, on a huge corpus, and show that our multi-relational contextualized event representation can improve performance when learning script knowledge without direct supervision and provide a better representation for the implicit discourse sense classification task.

6.1 Introduction

Representing world knowledge, and reasoning over it, to help improve language understanding is one of the longest standing AI goals. Structured knowledge representations such as *scripts* [27] capture temporal relations between events to describe human-level representations of common scenarios. For example, the *Restaurant Script* captures the fact that food is first ordered and only then paid for. Initial works relied on manual script construction, a labor-intensive task that is hard to scale to the number of possible scenarios. More recent works focus on extracting this knowledge directly from text, using symbolic event representations [1] or more recently, exploiting representation learning advances and representing events using dense vectors, learned from data [2], [13], [30], [32], [52]. While these works differ in the way the internal structure of the event is represented, broadly speaking, the resulting models resemble word-embedding approaches [9], representing event co-occurrence in a low-dimensional vector space, and as a result use vector similarity over their embedding to measure their relationship.

In this paper, we follow the observation that many natural language understanding tasks require a more expressive representation that can capture the context in which events appear [127] and consider multiple relations between events [14], and going beyond simple event similarity to represent relations. To help explain the intuition behind it, consider the following example, consisting of a short story and a Multiple-Choice Narrative Cloze (MCNC) question [2], the standard evaluation for such models.

Example 1: *Jenny gets coffee*

Jenny woke up very early and had some time to kill. She went outside and noticed that it was raining, so she went inside her favorite coffee-shop. She greeted the waiter ...

What happened next?

(a) she bought a new car.

(b) she ordered a steamy latte.

(c) she ordered a large breakfast

(d) she asked about open positions.

Events typically correspond to predicate-argument structures, and the narrative cloze task is modeled as ranking event pairs based on their similarity, using consecutive events as positive examples. Based on this approach, identifying that (a) is not a reasonable option is straightforward, however, the task of separating between (b), (c) and (d) is much harder, and requires models that can reason about the broader context in which an event occurs, capturing the *cause* of entering the coffee-shop (i.e., killing time) and the activity most associated with it (i.e., ordering coffee).

To meet this challenge we suggest a multi-relational contextualized representation of events, generalizing ideas from contextualized word representations [10], [120] to multi-relational narrative representation. Similar to contextualized word representations, we suggest learning an event representations which captures the narrative it is a part of. For example, the event *"she went inside the coffee-shop"* would be represented differently given different context, such as different weather conditions (*"it was sunny and warm"*), differ-

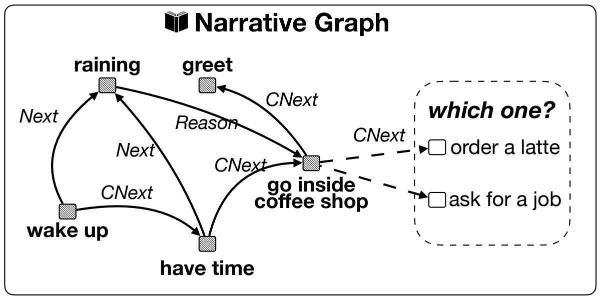


Figure 6.1. Narrative Graph extracted for Example 1. Some edges are omitted for clarity.

ent time of the day (*"it was almost noon when she woke up"*) or if the protagonist needed employment. In each one of these cases, the relationship between the contextualized event representation and the answer candidates would be different. **Unlike** contextualized word embedding models, our challenging settings require dealing with complex internal event structure (associations between the predicate and the entities, and their semantic roles), long narrative text, often beyond the length that can be effectively represented using these models, as well as representing complex relationships between events, beyond co-occurrence. To identify the association between the question and the correct answer, (b), the contextualized event representation should capture the reason for entering the coffee-shop, in this case indicated by the discourse connective *"so"*.

We propose using Narrative Graph (NG) to represent the text, consisting of nodes, corresponding to events, and edges representing observed relations between events. These relations capture the sequential order of event occurrence, represented using the **Next** relationship, events sharing a coreferenced entity are connected via the **CNext** relationship. In addition, we represent discourse relations corresponding to six relations defined in the Penn Discourse Tree Bank (PDTB) [12], which include **Before, After, Sync., Contrast, Reason** and **Result**. We rely on the discourse connectives associated with each relations to add these relations to the NG. Figure 6.1 provides an example of a partial narrative graph corresponding to the example above. We define the contextualized event embedding over this graph, by using a Relational Graph Convolution Network (R-GCN) [128], a relational variant of the Graph Convolution Network architecture (GCN) [129], which creates contextualized node representations by unfolding the graph structure recursively into a tree structure and learns a composition function, similar to a tree-based Recursive NN. This architecture allows us to take into account the narrative structure and the different discourse relations connecting events when embedding the event node.

We associate the event text, along with its local context, with each node, and use BERT [10] to encode its initial representation, contextualized locally. During training, the error is back-propagated over the graph to train the narrative relationships' composition parameters, and then to BERT, to train the NG-contextualized representation of the individual event.

We define an unsupervised learning process, learning to recover removed edges from a given narrative graph and capture incorrect associations between event nodes and edges. This process allows the model to learn the association between the missing information and the observed context in the narrative graph. We use the New York Times section of English Gigaword [104] for training the model. We evaluate the model on MCNC and its relational variants, as well as the popular, and challenging, implicit discourse classification task [72].

6.2 Related Work

Statistical script learning is an unsupervised learning problem addressing the probabilities of event co-occurrence. [1] started the early work, using Pairwise Mutual Information (PMI) -based models to calculate the conditional probability distribution. In recent years, neural-based learning frameworks emerged, leading to a wave of model evolution. [2] combine Skip-Gram [9], a word embedding model, with neural networks for learning event representations. [50] built a Long Short Term Memory (LSTM) network to learn the next relation along coreferent event chains, modeling relationships over event sequences. [23] constructed a high-dimensional tensor-based neural network, inspired by Computer Vision models, to learn event representations. [13] and [32] showed that adding event features, such as entity animacy, sentiments, or event order information, help commonsense inference. [52] started using graph structures, beyond pairwise or sequential models, to capture event context. [14] made a multi-relational model capturing different relation types between events with translating-based objective functions, which is the closest work to this paper. In this paper, the NG model uses two-level (word and event) contextualizations, built on top of a pre-trained language model–BERT, coupled with multi-relational graph structures, to learn event representations.

In the literature, the definitions of events can be categorized in two ways: entity-centric or predicate-centric. Early works [1] operated on the entity-centric events, following coreference chains of a specific entity to model sequences of events. Predicate-GR [2] was a widely adopted event definition here, consisting of a pair of dependency type, such as subject or object, and predicate token, such as verbs. Recent works [13], [14], [21] moved to the predicate-centric events (also called multi-argument events). Each event was anchored at a predicate and considered related entity mentions and modifiers as context to the event. Other works focusing on event extraction [130] or relation extraction [90] also adopted this definition, as it tends to capture more comprehensive view of events' semantics, aggregating information from multiple entities. In this paper, we also choose this definition, since our goal is to utilize events' context to model event relationships.

Graph neural models are often applied in Knowledge Base Completion. Early works used random-walk-based methods, aiming to build scalable neural models, such as DeepWalk [131], node2vec [132], LINE [133] and GraphSAGE [134]. Graph Convolution Networks (GCN) introduced by [129] provide an efficient way of aggregating features from neighboring nodes. Several GCN variants followed, Relational Graph Convolution Networks (R-GCN) [128] added relation type information to address the multi-relational knowledge bases. Graph Attention Networks (GAT) [135] manipulated attention layers for aggregating neighboring messages. Gated mechanisms [136], [137] were proposed for mitigating the impact of data noise. GCN were also used for NLP applications, to represent structure [136] and social information [138]. In this paper, we adopt R-GCN for NG, as modeling different types of relationships is crucial for event commonsense inference, as attested by [14].

Discourse relations are crucial aspects for completing language understanding. Early works focused on identifying explicit and implicit discourse relations under supervised settings [72], [139]–[141], while recent works mined discourse connectives to refine sentence representations unsupervisedly [142]–[144]. Our work learns discourse relations between events by leveraging the fact that some explicit connectives and their categories are relatively easy to identify. We build a simplified discourse annotator that can be used to extract discourse relations between events without suffering from high noise.

6.3 Model

6.3.1 Overview

We propose a learning framework for constructing event embeddings, contextualized by a relational event graph. The proposed approach can be used for many discourse and narrative analysis tasks, that go beyond the sentence level.

The framework consists of two levels of hierarchical contextualizations. The first, defined at the word level, uses contextualized word embeddings, such as BERT [10], which was applied successfully to various Natural Language Understanding (NLU) tasks. The second level, which is the main novelty of this paper, contextualizes *event*. Similar to words, events in different scenarios can have different meanings, e.g., a smile can mean positive or negative signs. As contextualized word embeddings tend to focus on local information, failing to capture high-level conceptual transitions, such as discourse relations, we suggest a new data structure to represent the input, called Narrative Graph (NG), which represents a document using its events and their relationships.

6.3.2 Preprocessing

Event Extraction

The model uses predicate-based event definition, as described in Chapter 2. We abbreviate the event notation e^p to e in this chapter. We define events as verb predicates that have at least one dependency link to entity mentions. The dependency links include subject (nsubj), direct object (dobj), indirect object (iobj), prepositional words or noun modifiers (nmod)[1]. Along with the verb predicates, we take the sentence they appear in as their local (word-level) context. To further differentiate the representations of the events appearing in the same sentence, we take into account their predicate position as inputs. Each event appears in a NG as a node, and edges between nodes represent event relationships.

[1]Stanford CoreNLP [101] pipeline is used for extracting dependency trees and coreference resolutions.

Relation Extraction

Table 6.1. Statistics of event relations extracted from *New Youk Times* section of *English Gigawords* [104]. 1.42M documents are used after excluding documents that are too long or too short.

Relation Types between Events		
Complete Name	Abbrev.	#relations.
Next	**Next**	274M
Coreferent Next	**CNext**	66M
Temporal.Async.Precedence	**Before**	1.63M
Temporal.Async.Succession	**After**	1.52M
Temporal.Synchrony	**Sync.**	0.55M
Comparison.Contrast	**Contrast**	0.91M
Contingency.Cause.Reason	**Reason**	0.22M
Contingency.Cause.Result	**Result**	2.41M

The relation is defined as a triplet (e_h, r, e_t), where e_h and e_t are head and tail events, and r is the relation type. We extract eight types of relations, including two narrative relations, **CNext** and **Next**, and six discourse relations. All relations are directional. Table 6.1 summarizes statistics of the relations we extracted from the corpus English Gigaword [104]. We explain each relation type as follows:

(1) The **CNext** relation stands for Coreferent Next relation, inspired by [1], capturing narrative relationships between events with shared entities on coreference chains[1]. Based on the procedure proposed by [14], we first identify all possible events and connect pairs of the events with a **CNext** relation if they have entity mentions appearing in the same coreference chain. For example, "Jim *shot* John. John *died*." *shot* and *died* have the **CNext** relation *(shot, **CNext**, died)* because the entity John is the participant to both events in a sequential order.

(2) The **Next** relation is defined between events appearing in the neighboring sentences. It aims to capture the event relationship where two events are relevant but do not have shared participants. For example, "The weather turned bad. The rain started falling." has the relation *(turned, **Next**, falling)*. These two events have no shared participant but are clearly related.

(3) The six discourse relations (the last six rows in Table 6.1) are selected from PDTB for capturing transitions between events. For example, "Jenny fell asleep, because she was tired." has a relation **Reason** and the argument spans (*ARG1* and *ARG2*) are the two clauses. Instead of having relations over arguments spans, we adapt the relation definition to the event level, where e_h comes from *ARG1* and e_t comes from *ARG2*. Note that when getting sentence context for event predicates, we mask the discourse connective, such as "because", from the model, because we want the model to learn relationships between events, rather than a simple decision function of key words. More detailed relation definitions can be found in the PDTB annotation manual [12].

Since the relations annotated in PDTB are not enough for generalizing event embeddings, we construct a rule-based discourse annotator. We first compile a list of discourse connectives by looking at the annotated relations in PDTB. To reduce the noise, only highly indicative connectives are considered. For example, "however" indicates **Contrast** relation and "in the meanwhile" denotes **Sync.** relation. We then search for the discourse connectives (CONN) in documents, and use three patterns to locate the argument spans:

1. {ARG1}. {CONN} {ARG2}.
2. {ARG1}, {CONN} {ARG2}.
3. {CONN} {ARG2}, {ARG1}.

where the first pattern has a discourse relation across two sentences while the other two have it in one sentence with multiple clauses. Since each argument span could have multiple events, we use all possible pairs. While the extracted relations are noisy, we demonstrate that they help in learning event representations in experiments.

Narrative Graph

The extracted events and relations from a document form a NG. The NG is an event-level abstraction of the document, as depicted in Figure 6.1, describing typed relational transitions between events. In this paper, the NG is modeled with a graph neural network. We have to limit the graph size, as there are physical memory limitation when training the network.

The size is controlled by two hyperparameters: s_{min} and s_{max}, standing for the minimum and maximum numbers of nodes.

6.3.3 Neural Architecture

We define two contextualized embedding functions:

$$\begin{aligned} \mathrm{e} &= f_{word}(p, loc(p), ctx(p)), \\ v &= f_{event}(\mathrm{e}, g(\mathrm{e})), \end{aligned} \tag{6.1}$$

where p is the target event predicate; $loc(p)$ is the token offset of the predicate in the sentence; $ctx(.)$ is the local context function; $f_{word}(.)$ encodes p and get its contextualized word embedding e, representing the event with the local context; $g(.)$ is the event context function, retrieving all events and relations in the document, i.e., $g(\mathrm{e}) = \{\mathrm{e}^*, r | \mathrm{e}^* \in doc(\mathrm{e}), r \in doc(\mathrm{e})\}$; lastly, $f_{event}(.)$ encodes the event, along with its NG, and outputs the contextualized event embedding v.

In this paper, we use BERT [10] for f_{word}, Relational Graph Convolution Network (R-GCN) [128] for f_{event}, and NG for the event context function $g(.)$. The following subsections will explain more in details. Note that this architecture setting is for demonstrating purposes. Our framework retains the flexibility of adopting other embedding and context functions.

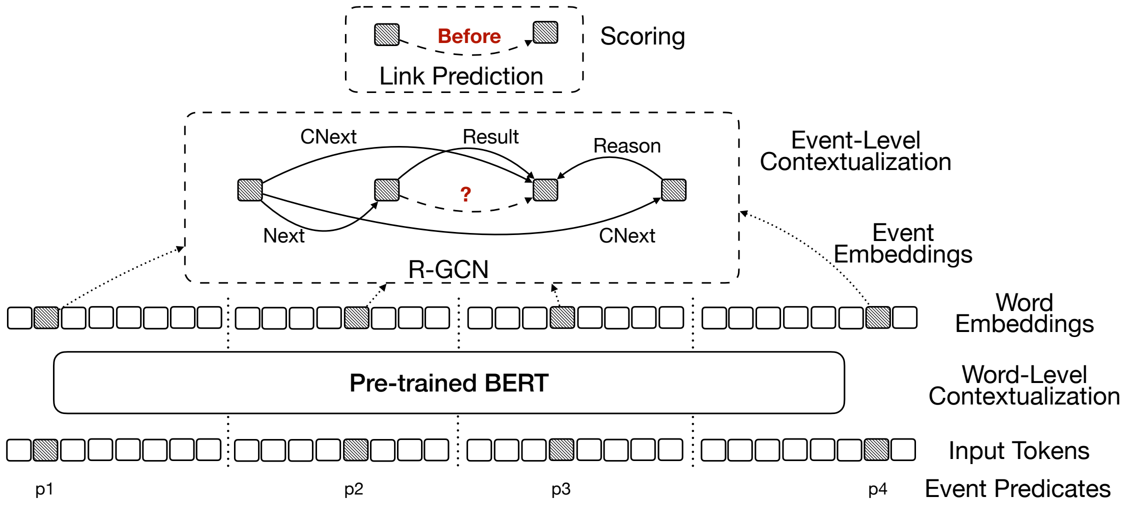


Figure 6.2. Neural architecture for the Narrative Graph model.

Word-Level Contextualization

Figure 6.2 visualizes the NG model. The input tokens are the event predicate along with its sentence context. We use BERT as the local (word-level) encoder. It has three embedding tables to represent the input, which are token embedddings, position embeddings, and token type embeddings. The token type embeddings were originally used for distinguishing input sentences for BERT's next-sentence pre-training task. Recent work [90] has shown that an effective way to fine-tuned BERT for events is to encode special tokens, such as event predicates, with the token type (token_type_id). We adopt this idea and use the token type inputs to mark event predicates, i.e., $token\_type\_id = 1$ for predicate tokens and $token\_type\_id = 0$ otherwise. This method emphasizes predicates when encoding events and generates slightly different contextualized representations for different emphases, even in the same sentence. For the rest of this paper, unless mention explicitly, we encode events with BERT in this way. In our training procedure, we initialize our model with pre-trained BERT and fine-tune it, and represent each event with its predicate word embeddings output from BERT.

Event-Level Contextualization

Graph Convolution Networks (GCN) [129] were designed to process graph structures by propagating messages between local neighboring nodes through graph convolution. R-GCN [128] adds relational considerations so that it can operate on multi-relational graphs[2]. The network is defined as follows:

$$h_{\mathrm{i}}^{l+1} = ReLU\left(\sum_{r\in R}\sum_{u\in N_r(v_{\mathrm{i}})}\frac{1}{c_{\mathrm{i},r}}W_r^l h_u^l\right), \tag{6.2}$$

where h_{i}^{l+1} is the hidden representation for the node v_{i} at layer $l+1$; $N_r(v_{\mathrm{i}})$ is the set of neighboring nodes under the r relation; $c_{\mathrm{i},r}$ is the normalizatoin factor; W_r^l is the relation-specific parameters for layer l; and R is the set of relation types (in our case, the eight types denoted in Table 6.1).

[2]We also have experimented with gated mechanism [136] for R-GCN to mitigate the noise from parsing errors. However, the performance is slightly worse.

The R-GCN is connected to BERT on top, taking only predicate word embedding to represent each event node. The node representations are contextualized by local neighbors according to NG. The number of R-GCN layers l_{rgcn} is a hyperparameter to control the order of neighbors to be considered.

Negative Sampling

As the NG model is contextualized over NG, we have to create negative graphs by removing some edges and predict them. To do so, we first determine a set of hyperparameters: the number of truncated graphs n_{neg_g} created for each NG, the proportion of edges to be removed r_{neg_e} for each truncated graph, and the number of negative edges n_{neg_e} to be sampled for each removed edge. Once they are determined, we sample the edges to be removed by their relation type, based on a smoothed distribution, where we sample **Next**, **CNext**, and each discourse relation with probabilities 0.5, 0.2 and 0.05 respectively. The reason why we smooth the distribution is to avoid undersampling the rare relation types. For each sampled edge, we truncate its e_h, r, and e_t uniformly.

Objective

There are two common objectives researchers have been using for optimizing graphical networks: node classifications and link predictions [128]. We select the latter one, as our goal is to capture structural transitions between events. However, it is possible to train for both objectives jointly within our framework, and we leave it for future work.

We score a target link (triplet) with a modified version of DistMult [145], an effective scoring function designed for knowledge base completion. The function is defined as follows:

$$D(h, r, t) = v_h^T W_r v_t, \tag{6.3}$$

where v_h and v_t are the representations for head and tail events of the triplet, and $W_r \in R^{d \times d}$ are relation-specific parameters. The original DistMult restricts W_r to a diagonal matrix to account for the huge amount of relation types existing in knowledge bases. We relax this as we need to address more fine-grained differences between relations, such as directionality[3].

The final loss function is the Cross-Entropy Loss with weighted classes:

$$\mathscr{L}_p = -\frac{1}{|\mathscr{T}|} \sum_{(h,r,t,y)\in\mathscr{T}} y \log(\sigma(w_r \cdot D(h,r,t))) \\ +(1-y)\log(1-\sigma(w_r \cdot D(h,r,t))), \tag{6.4}$$

where $\mathscr{T}$ is the set of sampled triplets with labels; $\sigma(.)$ is the logistic sigmoid function; w_r is the class weight depending on relation type distributions; and y is the binary label.

6.4 Evaluations

Our evaluation consists of two parts. The first part conducts intrinsic evaluation, evaluating the basic characteristics of the NG model. In the second part extrinsic evaluation is performed, by using the NG event embedding for a downstream task–Implicit Discourse Relation Sense Classification [72], from CoNLL 2016. The source code and models used in this paper are publicly available[4].

6.4.1 Data and Experiment Settings

For pretraining and intrinsic evaluations, we use the NYT section of English Gigaword [104], which contains about 2M newswire documents. We filter out extremely short and long documents by limiting the number of graph nodes between 20 and 350 ($s_{min} = 20$ and $s_{max} = 350$). This leaves us 1.42M documents, and about 345M relations are extracted (see Table 6.1). The data splits follow [2]'s setting, dividing the documents into train/validation/test sets. Other hyperparameters are listed as follows: the number of R-GCN layers $l_{rgcn} = 2$, the number of truncated graphs $n_{neg\_g} = 4$, the ratio of edges to be removed $r_{neg\_e} = 0.05$, the number of negative edges per removed edge $n_{neg\_e} = 20$, the hidden layer

[3]We have also tried other scoring functions, such as TransE families [110], but DistMult outperforms them.
[4]https://github.com/doug919/narrative_graph_emnlp2020

size $d = 128$, the class weights in the loss function are inversely proportional to the class distribution given in Table 6.1.

For training the model, we use AdamW optimizer [146] with initial learning rate 0.0002. No warm-up steps are used. The BERT encoder is initialized with BERT-Tiny [147], a distilled compact version of BERT to accommodate the large graph structure, and fine-tuned during training. We experiment with dropout rates {0, 0.1, 0.2, 0.4} and use the model that achieves the best result in the validation set. The number of model parameters is 4812168. We search the hyperparameter for about 30 trials using a month, and use F1-macro score over Triplet Classification task (Table 6.5) for selecting the model. The expected validation performance is 58.89% F1-macro score. The final model is trained on four NVIDIA 1080Ti GPUs for 5 days.

For extrinsic evaluation, the data is from the CoNLL 2016 shared task, using their data splits [72].

6.4.2 Intrinsic Evaluation

The intrinsic evaluation consists of four tasks. The first task is Multiple-Choice Narrative Cloze (MCNC), proposed by [2], which measures the models' ability to recover a missing event given its coreferent event chain. The second evaluates the models' ability to identify the tail event, given the head event and relation, i.e., $(\mathrm{e}_h, r, ?)$. The third evaluates the models' ability to detect the correct relations between two given events, i.e., $(\mathrm{e}_h, ?, \mathrm{e}_t)$. The fourth evaluation is a binary triplet classication, inspired by knowledge base completion, where a test triplet is given and the binary classifier identifies it is true or false.

Baselines

Six baseline models are considered.

1. **Random**: makes random predictions.
2. **EventComp-BERT**: is an implementation of EventComp [2] but replace the event encoder with BERT. It uses a feed-forward neural network to compose a coherence

score for event pairs based on coreference chains. It is a single-relational model that only considers **CNext**.

3. **EventLSTM-BERT**: is an attention-based LSTM model that captures event coreference chains. It is also a single-relational model (**CNext**). We follow [32]'s architecture and settings but use BERT for encoding events and remove the dynamic memory component.
4. **EventTransE-BERT**: is an implementation of EventTransE [14], but replace the event encoder with BERT. It is a strong uncontextualized event embedding model, outperforming various models on the MCNC task. It trains on multi-relational data and a translating-based loss (TransE) is used for scoring event triplets.
5. **Event-BERT-sim**: uses the pre-trained BERT model without fine-tuning and scores event pairs with cosine similarity, which simply measures the embedding similarity between events. The relation type is not taken into account. This baseline gives the idea about how much performance gain can be acquired from word-level contextualization.
6. **Event-BERT-ft**: is fine-tuned (ft) using the same objective and data as the NG. However, the event-level contextualization, i.e., R-GCN layer, is skipped, so it is a pairwise event models powered by BERT. It is a multi-relational model and the loss function is identical to NG.

Multiple-Choice Narrative Cloze

We begin with the popular benchmark–MCNC, which predicts the next event, given its preceding events. It was originally proposed by [1] as a ranking problem, which ranks all possible events given an event chain. However, the ranking metric over a huge set of event vocabularies is not easy to interpret for model comparisons. [2] thus adapted it to a multiple-choice setup, rendering a clear performance metric. [14] further generalized it to the multi-relational setting. In this task, we follow [2]'s set-up. Each question has an input sequence of 8 events that are connected with **CNext**, and the target event has 4 negative and 1 positive choices. Since the question set released by previous works does not contain document information required by our NG model, we re-sample the question set with document information. 10000 test instances are sampled from the test split.

Table 6.2. Accuracy scores (%) for MCNC. The task asks models to predict the target event, out of 5 choices, given a sequence of events with **CNext** relation. The model type S means single-relational models and M means multi-relational models.

Methods	**Type**	**Validation**	**Test**
Random	-	20.00	20.00
Event-BERT-sim	S	40.18	41.24
EventComp-BERT	S	54.12	53.86
EventLSTM-BERT	S	62.78	62.62
Event-BERT-ft	M	47.22	47.20
EventTransE-BERT	M	57.92	58.35
NG	M	**65.86**	**63.59**

Table 6.2 lists the result. The first row shows the random baseline for 5 choices. The following three rows are single-relational models that only consider event co-occurrence with **CNext** relation. Event-BERT-sim uses event similarity without fine-tuning, which gives the basic performance. EventComp-BERT fits to the event pairs with **CNext** relation and perform better. The sequential model EventLSTM-BERT preforms very well, since this task set-up is perfect for sequential models like LSTM. However, EventLSTM-BERT does not have the ability to digest multi-relational data. The rest three models are multi-relational models. NG outperforms EventLSTM-BERT significantly, since it encodes the narrative graph structure and other relation types. If we compare NG with Event-BERT-ft (NG without R-GCN), we can see that the graph structure improves the result with a large margin (18.64% absolute accuracy improvement in the test set), making NG the best performer over all the single- and multi-relational models.

Predict Coreferent Next Event

In this task, we predict the tail event of a **CNext** relation. Unlike MCNC, where a sequence of coreferent events are given, we only take one event as the input and predict the other. We also adopt the multiple-choice setting, and to strengthens the evaluation,

Table 6.3. Accuracy scores (%) over 10-choice MC-questions for CNext relation. Each question has the form $(e_h, r, ?)$, where the head event e_h and relation r are given and the model predicts the correct tail event.

Methods	Accuracy
Random	10.00
Event-BERT-sim	32.43
EventComp-BERT	55.16
Event-BERT-ft	50.10
EventTransE-BERT	58.16
NG	**60.94**

Table 6.4. Predicting the discourse sense, out of 6 candidates, between two given events, i.e., $(e_h, ?, e_t)$.

Methods	Acc.	F1	MRR	Recall@3
Random	16.67	-	-	-
EventTransE-B	44.65	29.33	64.59	81.05
Event-BERT-ft	59.24	55.42	75.27	91.26
NG	**80.27**	**79.68**	**88.05**	**95.74**

we increase the number of candidates to 10 to make the task more challenging. 5000 test instances are randomly sampled from the test split.

Table 6.3 shows the task result. We can see that even under this more challenging setting, NG can still outperforms all the models. Event-BERT-ft can be interpreted as "NG without R-GCN". We can see that without the event-level contextualization, the performance drops significantly (-10.84% absolute accuracy). The result denotes that as the high-level structure over events are contextualized in the embeddings, the NG model can make better predictions for events in various scenarios. The EventTransE-BERT is a strong competitor here, as it also benefits from multi-relational modeling, but, again, without the event-level contextualization, it performs worse than NG. This again attests the importance of encoding the narrative graph structure. Note that the EventLSTM-BERT cannot be applied here, as it requires a fixed length input.

Table 6.5. Binary classification for a given triplet $(\mathrm{e}_h, r, \mathrm{e}_t)$. The scores are macro-averaged over the **minority** class. The validation performance is 56.91%.

Methods	**Precision**	**Recall**	**F1**
Event-BERT-sim	3.45	74.04	6.59
EventTransE-BERT	30.17	53.61	38.62
Event-BERT-ft	49.19	36.79	42.09
NG	**68.20**	**66.21**	**67.19**

Predict Discourse Sense

In this task, the models predict the discourse sense for a given pair of events. It is a multi-class classification problem over 6 discourse senses used in this paper.

Table 6.4 shows the result with four different metrics, including accuracy, F1-macro score, Mean Reciprocal Rank (MRR), and Recall@3. The later two metrics evaluate models' ranking ability. We compare three multi-relational models in this task. The NG outperforms EventTransE-BERT, which means that the DistMulti objective for the other two models is more sensitive to relation types than TransE. The NG also outperforms Event-BERT-ft. Both models achieve high Recall@3, which means that over 90% of correct relations are ranked on the top half. The main difference between the two models is the event-level contextualization (R-GCN component), which brings in 21.03% absolute accuracy improvement. The NG model can both rank and select the answer with the most confidence.

Triplet Classification

Table 6.5 shows the result for triplet classifications, where a triplet is given and the task is to predict it is true or false. There are 229k positive and 4584k negative triplets, sampled with the smoothed class distribution described in the Negative Sampling section. Event-BERT-sim does not consider relation types, so it only measures event similarity based on BERT embeddings. The low precision and high recall shows that most events are similar if we do not take the relation type into account. Again, the big performance gain of NG over

Table 6.6. Triplet classification breakdown for *NG*. The scores are macro-averaged over the **minority** class.

	NG			
	Precision	**Recall**	**F1**	**#pos. / #neg.**
Next	59.53	89.86	71.62	149k / 2343k
CNext	46.80	40.38	43.36	59k / 1027k
Before	62.45	69.34	65.72	4.9k / 223k
After	82.18	58.31	68.22	4.5k / 217k
Sync.	73.35	59.08	65.45	1.9k / 179k
Contrast	67.97	76.15	71.83	6.2k / 244k
Reason	73.24	76.87	75.01	2.9k / 192k
Result	80.08	59.72	68.42	0.6k / 159k
macro-avg	68.20	66.21	67.19	-

Event-BERT-ft is due to the NG-contextualized event embeddings, which offers high-level summary of documents.

Table 6.6 shows triplet classification results by type. Given the low positive-to-negative examples ratio, we report the F1 score over the minority class, and the macro-average over all these scores. We note the difficulty of predicting **CNext**, although it has the second highest number of examples. We attribute this to the noise generated by the coref resolution solver, as other relations have clearer signals for learning, and the fact that **CNext** is the only relation that connects two events that could be far away from each other in text. We leave this issue for future work.

6.4.3 Extrinsic Evaluation

The last evaluation is over a downstream task, Implicit Discourse Sense Classification, a subtask from CoNLL 2016 shared task [72]. The task is a multi-class classification task with 15 discourse classes, including explicit and implicit relations. The explicit relations mean that the discourse connective, such as “because”, exists in the text, providing clues for the sense prediction, while the implicit one does not have it. We only evaluate on the subtask for implicit relations as it is both challenging and useful for language understanding.

Table 6.7. F1-micro scores for Implicit Discourse Sense Classifications. EvTransE is abbreviated for EventTransE. The start signs mean that its event representation is concatenated with *ELMo* word embeddings. The validation performance for NG is 46% F1 score.

Methods	**Test**	**Blind**
PurdueNLP [124]	34.45	29.10
ecnucs [125]	40.91	34.18
ttr [126]	36.13	37.67
ELMo [120]	37.65	36.72
EvTransE* [14]	39.05	38.35
NG*	**42.84**	**43.91**

Several baseline models are chosen from the leader board of the shared task, including the best and median systems (the first three rows). Following [14]'s experiment setting, ELMo [120] is used as the basic features for the supervised classification. The input features feed to a self-attention layer and then two fully-connected hidden layers, with dimensions 256 and 128, are added on top for classifications. The EventTransE baseline concatenates its event representation with ELMo and feeds to the same network architecture. The NG model applies its event embeddings in the same way as EventTransE, and achieves the best performance.

6.5 Conclusions

We propose the Narrative Graph embedding model to learn contextualized event representations for disambiguating discourse relations. We use weak supervision, provided by the predictions of off-the-shelf NLP tools and a rule-based discourse annotator, to learn event representations capturing world knowledge useful for downstream tasks. Our model considers multiple discourse relations types, such as "contrast" or "cause". We evaluate our model on three intrinsic tasks, including triplet classification and event/relation predictions, as well as an extrinsic task–discourse relation classification. Our results show that the model can outperform competitive systems. In the future we intend to apply our model to discourse analysis tasks which require modeling long-range dependencies.

7. MODELING HUMAN MENTAL STATES WITH AN ENTITY-BASED NARRATIVE GRAPH

"An expert is a person who has made all the mistakes that can be made in a very narrow field."

—Niels Bohr (1885–1962)

This chapter presents our fourth work that combines our proposed three techniques described in the previous three chapters to construct an Entity-based Narrative Graph (ENG). ENG considers the mental states of the characters in a story, modeling entities, along with the events, to learn contextualized event representations in a grounded situation. We experiment with different weakly-supervised objectives for task-adaptive pretraining, in-domain training and symbolic inference to capture dependencies between different decisions in the output space. We evaluate our model on two narrative understanding tasks: predicting character mental states and desire fulfillment. We also conduct a qualitative analysis to examine how graph contextualization changes the event representations.

7.1 Introduction

Understanding narrative text requires modeling the motivations, goals and internal states of the characters described in it. These elements can help explain intentional behavior and capture causal connections between the characters' actions and their goals. While this is straightforward for humans, machine readers often struggle as a correct analysis relies on making long range common-sense inferences over the narrative text. Providing the appropriate narrative representation for making such inferences is therefore a key component. In this paper we suggest a novel narrative representation and evaluate it on two narrative understanding tasks, analyzing the characters' mental states and motivation [11], [148], and desire fulfillment [71], [149].

We follow the observation that narrative understanding requires an expressive representation capturing the context in which events appear and the interactions between characters' states. To clarify, consider the short story in Fig. 7.1. The desire expression appears early in the story and provides the context explaining the protagonist's actions. Evaluating the

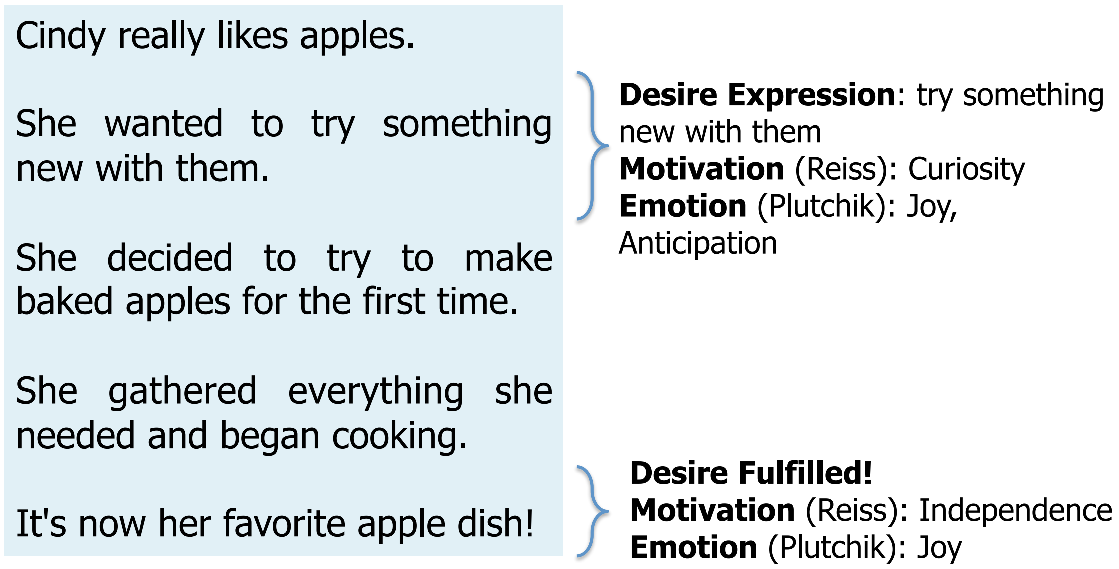

Cindy really likes apples.
She wanted to try something new with them.
She decided to try to make baked apples for the first time.
She gathered everything she needed and began cooking.
It's now her favorite apple dish!
Desire Expression: try something new with them
Motivation (Reiss): Curiosity
Emotion (Plutchik): Joy, Anticipation
Desire Fulfilled!
Motivation (Reiss): Independence
Emotion (Plutchik): Joy

Figure 7.1. Narrative Example

fulfilment status of this expression, which tends to appear towards the end of the story, requires models that can reason over the desire expression (*"trying something new"*), its target (*"apples"*) and the outcome of the protagonist's actions (*"it's now her favorite apple dish!"*). Capturing the interaction between the *motivation* underlying the desire expression (in Fig. 7.1, CURIOSITY) and the *emotions* (in Fig. 7.1, ANTICIPATION) likely to be invoked by the motivation can help ensure the consistency of this analysis and improve its quality.

To meet this challenge we suggest a graph-contextualized representation for entity states. Similar to contextualized word representations [10], [120], we suggest learning an entity-based representation which captures the narrative it is a part of. For example, in *"She decided to try to make baked apples for the first time"* the mental state of "she" would be represented differently given a different context, such as a different motivation for the action (*"Her mother asked her to make an apple dish for a dinner party"*). In this case, the contextualized representation would capture the different emotion associated with it (e.g., FEAR of disappointing her mother). Unlike contextualized word embedding models, our challenging settings require dealing with complex internal event structure (associations between the predicate and the entities, and their semantic roles), long narrative text, often beyond the length that can be effectively represented using these models. Furthermore, we exploit the event structure, and incorporate constraints ensuring consistency between the mental state attributes of the characters.

We begin by generating an Entity-based Narrative Graph (ENG) representation of the text. Unlike other graph-based narrative representations [150]–[152] which require intensive human annotation, we emphasize simplicity and shift the focus from symbolic graph representations of nuanced information to their learned embedding. In our representation nodes correspond to events and edges represent observed relations between events. These relations capture the sequential order of event occurrence, represented using the **Next** relationship. Events sharing a coreferenced entity are connected via the **CNext** relationship. We also represent discourse relations corresponding to six relations defined in the Penn Discourse Tree Bank (PDTB) [12], which include **Before, After, Sync., Contrast, Reason** and **Result**.

We define the contextualized event embedding over this graph, by using a Relational Graph Convolution Network (R-GCN) [128], a relational variant of the Graph Convolution Network architecture (GCN) [129], which creates contextualized node representations by unfolding the graph structure recursively into a tree structure and learning a composition function. This architecture allows us to take into account the narrative structure and the different discourse relations connecting events when embedding the event node.

We first define a self-supervised pre-training process for embedding the narrative graph, by learning to recover removed edges and capture incorrect associations between event nodes and edges. We apply our the learned graph representation to two challenging narrative analysis tasks, predicting characters' psychological states [11] and desire fulfilment [71] and show that our model can outperform competitive transformer-based representations of the narrative text. Our code and trained models will be publicly available in the camera-ready version.

7.2 Related Work

Tracking entities and modeling their properties has proven successful in a wide range of tasks, including language modeling [153], question answering [154] and text generation [155].

In an effort to model complex story dynamics in text, [11] released a dataset for tracking emotional reactions of characters in stories. In their dataset, each character mention is annotated with three types of mental state descriptors: Maslow's "hierarchy of needs" [156], Reiss' "basic motives" [157], that provide a more informative range of motivations, and Plutchik's "wheel of emotions" [158], comprised of eight basic emotional dimensions (e.g. joy, sadness, etc). In their paper, they showed that neural models with explicit or latent entity representations achieve promising results on this task. [40] approached this task by extracting multi-hop relational paths from ConceptNet, while [159] leveraged semantics of the emotional states by embedding their textual description and modeling the co-relation between different entity states. [71] introduced a dataset for the task of desire fulfillment. They identified desire expressions in first-person narratives and annotated their fulfillment status. They showed that models that capture the flow of the narrative perform well on this task.

Representing the narrative flow of stories using graph structures and multi-relational embeddings has been studied in the context of script learning [14], [15], [160]. In these cases, the nodes represent predicate-centric events, and entity mentions are added as context to the events. In this paper, we use an entity-centric narrative graph, where nodes are defined by entity mentions and their textual context. We encode the textual information in the nodes using pre-trained language models [10], [161], and the graph structure with a relational graph neural network [128]. To learn the representation, we incorporate a task-adaptive pre-training phase. [162] showed that further specializing large pre-trained language models to domains and tasks within those domains is effective.

7.3 Entity-based Narrative Graph

7.3.1 Framework Overview

Many NLU applications require understanding entity states in order to make sophisticated inferences [11], [17], [19]. In this work, we propose a learning framework that includes task-adaptive pretraining (TAPT) and downstream task training to train an entity-based narrative graph (ENG), a graph neural model designed to capture implicit states and interactions between entities. We extend the narrative graph proposed by [15], which models event relationships, and adapt it for entity mentions. Although ENG has the flexibility to be applied in various entity-based tasks, we demonstrate and explain it through a target downstream task, StoryCommonsense [11].

Our framework consists of four main components: Node Encoder, Graph Encoder, Learning Objectives, and Symbolic Inference, outlined in Figure 7.2. The node encoder is a function used to extract local information about the target entity mention corresponding to the uncontextualized node representation. The graph encoder uses a graph neural network to contextualize the node representations within a document, generating entity-context-aware representations. The learning objectives use this representation for several learning tasks, such as node classification, link prediction, and document classification. Finally, we include a symbolic inference procedure to capture dependencies between output decisions.

We introduce a training pipeline, containing pretraining and downstream training, following recent evidence suggesting that task-adaptive pretraining is potentially useful for many NLU tasks [162]. We experiment with three pretraining setups, including the common whole-word-masking pretraining [161], and two newly proposed unsupervised pretraining objectives based on ENG. We then evaluate two downstream tasks: StoryCommonsense and DesireDB [71]. StoryCommonsense aims at predicting three sets of mental states based on psychological theories [156]–[158], while DesireDB's goal is to identify whether a target desire is satisfied or not. Solving these tasks requires understanding entities' mental states and their interactions.

7.3.2 Node Encoder

Each node in our graph captures the local context of a specific entity mention (or character mention). Following [159], we format the input information to feed into a pretrained language model. For a given character c and sentence s, the inputs to the node encoder consist of three components $(s, ctx(c), L)$, where s is the sentence in which c appears, $ctx(c)$ is the context of c (all the sentences that the character appears in), and L is a label sentence. The label sentence is an artificial sentence of the form "[entity name] is [label 1], [label 2], ..., [label k]." The k labels correspond to the targets in the downstream task. For example, in

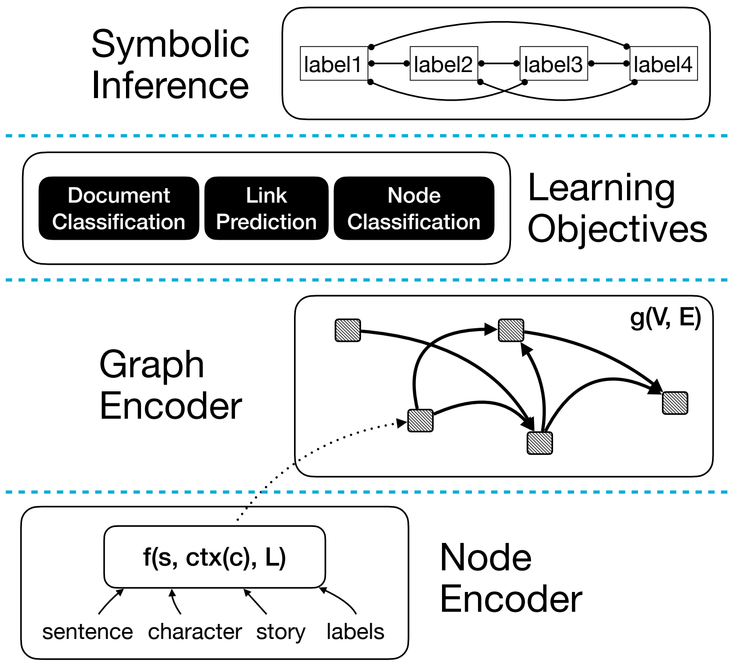


Figure 7.2. Overview of the ENG framework.

StoryCommonsense, the Plutchik state prediction task has eight labels characterizing human emotions, such as *joy*, *trust*, and *anger*. [159] shows that self-attention is an effective way to let the model take label semantics into account, and improve performance[1].

Our best model uses RoBERTa [161], a highly-optimized version of BERT [10], to encode nodes. We convert the node input $(s, ctx(c), L)$ to RoBERTa's two-sentence input format by treating s as the first sentence, and the concatenation of $ctx(c)$ and L as the second sentence. After forward propagation, we take the pooled sentence representation (i.e., $<s$ $>$for RoBERTa, *CLS* for BERT), as the node representation v. This is formulated as $v = f_{roberta}(s, ctx(c), L)$.

7.3.3 Graph Encoder

The ENG is defined as $ENG = (V, E)$, where V is the set of encoded nodes in a document and E is the set of edges capturing relationships between nodes. Each edge e $\in E$ is a triplet $(v1, r, v2)$, where $v1, v2 \in V$ and r is an edge type $(r \in R)$. Following [15], we use eight

[1]Our preliminary experiments also confirm this.

relation types ($|R| = 8$) that have been shown to be useful for modeling narratives. Next denotes if two nodes appear in neighboring sentences. CNext expresses the next occurrence of a specific entity following its co-reference chain. Six discourse relation types, used in [15] and defined in Penn Discourse Tree Bank (PDTB) [12], are also used in this work, including Before, After, Sync., Contrast, Reason, Result. Their corresponding definition in PDTB and can be found in Table 7.1. Following [15], we use the Stanford CoreNLP pipeline[2] [101] to obtain co-reference links and dependency trees. We use them as heuristics to extract the above relations and identify entities for TAPT[3]. Details of this procedure can be found in [15]. Note that although we share the same relation definitions, our nodes are defined over entities, instead of predicates.

For encoding the graph, we use a Relational Graph Convolution Network (R-GCN) [128], which is designed for Knowledge Base Completion. This architecture is capable of modeling typed edges and is resilient to noise. R-GCN is defined as:

$$h_{\mathrm{i}}^{l+1} = ReLU\left(\sum_{r\in R}\sum_{u\in U_r(v_{\mathrm{i}})} \frac{1}{z_{\mathrm{i},r}} W_r^l h_u^l\right), \tag{7.1}$$

where h_{i}^l is the hidden representation for the i-th node at layer l and $h_{\mathrm{i}}^0 = v_{\mathrm{i}}$ (output of the node encoder); $U_r(v_{\mathrm{i}})$ represents v_{i}'s neighboring nodes connected by the relation type r; $z_{\mathrm{i},r}$ is for normalization; and W_r^l represents trainable parameters.

Our implementation of R-GCN propagates messages between entity nodes, emulating the interactions between their psychological states, and thus enriching node representations with context. Note that our framework is flexible, and alternative node and graph encoders could be used.

7.3.4 Output Layers and Learning Objectives

We explore three learning problem types.

[2]Stanford CoreNLP v4.0 with default annotators.

[3]For StoryCommonsense, since the entity names are annotated, we simply use them.

Table 7.1. Alignment between PDTB relations and the abbreviations used in this paper. The third column in the sampling distribution.

Abbrev.	**PDTB**	**Distr.**
NEXT	–	50%
CNEXT	–	20%
BEFORE	Temporal.Async.Precedence	5%
AFTER	Temporal.Async.Succession	5%
SYNC.	Temporal.Synchrony	5%
CONTRAST	Comparison.Contrast	5%
REASON	Contingency.Cause.Reason	5%
RESULT	Contingency.Cause.Result	5%

Node Classification

For node classification, we use the contextualized node embeddings coming from the graph encoder, and plug in a k-layer feed-forward neural network on top ($k = 2$ in our case). The learning objectives could be either multi-class or multi-label. For multi-class classification, we use the weighted cross-entropy loss (CE). For multi-label classification, we use the binary cross-entropy (BCE) loss for each label[4]:

$$CE = -\frac{1}{N}\sum_{\mathrm{i}=1}^{N} \alpha_{\mathrm{i}} y_{\mathrm{i}} \log(S(g(f(x_{\mathrm{i}})))), \tag{7.2}$$

where $S(.)$ is the Softmax function, $f(.)$ is the graph encoder, $g(.)$ is the node encoder, x_{i} is the input including the target node i $((s, ctx(c), L))$ and all other nodes in the same document (or ENG), y_{i} is the label, and α_{i} is the weight.

Link Prediction

This objective tries to recover missing links in a given ENG. We remove a small portion of edges (20% in our case) and learn to predict them. To obtain negative examples, we sample edges by truncating either end of the positive edges, based on the relation type distribution given in Table 7.1, taken from the training set. Following [128], we score each edge sample with DistMult [145]:

[4]We tried weighted an unweighted BCE, and selected the unweighted one for our final model.

$$D(\mathrm{i}, r, \mathrm{j}) = h_\mathrm{i}^T W_r h_\mathrm{j}, \tag{7.3}$$

where W_r is a relation-specific trainable matrix (non-diagonal) and h_i and h_j are node embeddings coming from the graph encoder. A higher score indicates that the edge is more likely to be active. To learn this, we reward positive samples and penalize negative ones, using an adapted CE loss:

$$\begin{aligned} L = -\frac{1}{T} \sum_{(\mathrm{i},r,\mathrm{j},y) \in T} & y \log(\sigma(\epsilon_r D(\mathrm{i}, r, \mathrm{j}))) \\ & + (1 - y) \log(1 - \sigma(\epsilon_r D(\mathrm{i}, r, \mathrm{j}))), \end{aligned} \tag{7.4}$$

T is the sampled edges set, $y = \{0, 1\}$, $\sigma(.)$ is the Sigmoid function, and ϵ_r is the edge type weight, based on the edge sampling rate (Table 7.1).

Document Classification

For such tasks, such as DesireDB, we aggregate the node representations from the entire ENG to form a single representation. To leverage the relative importance of each node, we add a node attention layer. We calculate the attention weight for each node by attending on a target embedding. In DesireDB, we use the sentence embedding for the desire expression.

$$\begin{aligned} a_\mathrm{i} &= ReLU(W_a[h_\mathrm{i}; h_t] + b_a) \\ z_\mathrm{i} &= exp(a_\mathrm{i}) \\ \alpha_\mathrm{i} &= \frac{z_\mathrm{i}}{\sum_k z_k} \quad ; \quad h_d = \sum_\mathrm{i} \alpha_\mathrm{i} h_\mathrm{i} \end{aligned} \tag{7.5}$$

, where h_i is the i-th node representation, h_t is the target embedding (e.g, the desire expression), W_a and b_a are trainable parameters, and h_d is the final document representation. We then feed h_d to a two-hidden-layer classifier to make predictions. We use the loss function specified in Eq. 7.2.

7.3.5 Task-Adaptive Pretraining

Recent studies demonstrate that downstream tasks performance can be improved by applying the self-supervised pretraining task on the text of the target domain [162], we refer to this step as Task-Adaptive Pre-Training (TAPT). We investigate whether different TAPT objectives can provide different insights for the target task. We empirically set our target task as StoryCommonsense, and since StoryCommonsense is based on RocStories [59], we run TAPT on all the RocStories text (not including the validation and testing sets). We use the learning parameters suggested by [162] and explore three different TAPT settings:

Whole-Word Masking: Randomly masks a subset of words and asks the model to recover them from their context [161], [163]. We perform this task over RoBERTa, initialized with *roberta-base*.

ENG LinK Prediction: Weakly-supervised TAPT over the ENG. The setup follows Sec. 7.3.4(Link Prediction) to learn a model that can recover missing edges in the ENG.

ENG Node Sentiment Classification: Performs weakly-supervised sentiment TAPT. We use the Vader sentiment analysis [102] tool to annotate the sentiment polarity for each node in the ENG, based on its sentence. The setup follows Sec. 7.3.4 (Node Classification).

7.3.6 Symbolic Inference

In addition to modeling the narrative structure in the embedding space, we add a symbolic inference procedure to capture structural dependencies in the output space for the StoryCommonsense task. To model these dependencies, we use DRaiL [164], a neural-symbolic framework that allows for defining probabilistic logical rules on top of neural network potentials.

Decisions in DRaiL are modeled using rules, which can be weighted (i.e., soft constraints), or unweighted (i.e., hard constraints). Rules are formatted as horn clauses: A $\Rightarrow$ B, where A is a conjunction of observations and predicted values, and B is the output to be predicted. Weighted rules are associated with a neural architecture, used to learn the rule weights. The collection of rules represents the global decision, and the solution is obtained by performing MAP inference. In DRaiL, parameters are trained using the structured hinge loss.

We used feed-forward networks over the node embeddings obtained by the objectives outlined in Sec. 7.3.4 and 7.3.5, without back-propagating to the full graph. We model the following rules:

Weighted rules

We score each state, as well as *state transitions* to capture the progression in a character's mental state throughout the story.

$$\texttt{Entity}(\texttt{e}_\texttt{i}) \Rightarrow \texttt{State}(\texttt{e}_\texttt{i}, \texttt{l}_\texttt{i})$$

$$\texttt{State}(\texttt{e}_\texttt{i}, \texttt{l}_\texttt{i}) \wedge \texttt{HasNext}(\texttt{e}_\texttt{i}, \texttt{e}_\texttt{j}) \Rightarrow \texttt{State}(\texttt{e}_\texttt{j}, \texttt{l}_\texttt{j})$$

Where e_i and e_j are two different mentions of the same character, and `HasNext` is a relation between consecutive sentences. `State` can be either `Maslow`, `Reiss` or `Plutchik`.

Unweighted rules

There is a dependency between Maslow's "hierarchy of needs' and Reiss "basic motives" [11]. We introduce logical constraints to disallow mismatches in the Maslow and Reiss prediction for a given mention e_i. In addition to this, we model positive and negative sentiment correlations between Plutchik labels. To do this, we group labels into positive (e.g. joy, trust), and negative (e.g. fear, sadness). We refer to this set of rules as *inter-label dependencies.*

$$\texttt{Maslow}(\texttt{e}_\texttt{i}, \texttt{m}_\texttt{i}) \wedge \neg\texttt{Align}(\texttt{m}_\texttt{i}, \texttt{r}_\texttt{i}) \Rightarrow \neg\texttt{Reiss}(\texttt{e}_\texttt{i}, \texttt{r}_\texttt{i})$$

$$\texttt{Reiss}(\texttt{e}_\texttt{i}, \texttt{r}_\texttt{i}) \wedge \neg\texttt{Align}(\texttt{m}_\texttt{i}, \texttt{r}_\texttt{i}) \Rightarrow \neg\texttt{Maslow}(\texttt{e}_\texttt{i}, \texttt{m}_\texttt{i})$$

$$\texttt{Plut}(\texttt{e}_\texttt{i}, \texttt{p}_\texttt{i}) \wedge \texttt{Pos}(\texttt{p}_\texttt{i}) \wedge \neg\texttt{Pos}(\texttt{p}_\texttt{j}) \Rightarrow \neg\texttt{Plut}(\texttt{e}_\texttt{i}, \texttt{p}_\texttt{j})$$

Given that the DesireDB task requires a single prediction for each narrative graph, we do not employ symbolic inference for this task.

7.4 Evaluations

Our evaluation includes two downstream tasks and a qualitative analysis. We report the results for different TAPT schemes and symbolic inference on StoryCommonsense. For

Table 7.2. Results for the StoryCommonsense task, including three multi-label tasks (Maslow, Reiss, and Plutchik), for predicting human's mental states of motivations or emotions.

		Maslow			Reiss			Plutchik		
Group	**Models**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
G1	**Random**	7.45	49.99	12.96	1.76	50.02	3.40	10.35	50.00	17.15
	TF-IDF	29.79	34.56	32.00	20.55	24.81	22.48	22.71	25.24	23.91
	GloVe	27.02	37.00	31.23	16.99	26.08	20.58	19.47	46.65	27.48
	LSTM	30.34	40.12	34.55	21.38	28.70	24.51	25.31	33.44	28.81
	CNN	29.30	44.18	35.23	17.87	37.52	24.21	24.47	38.87	30.04
	REN	26.85	44.78	33.57	16.73	26.55	20.53	25.30	37.30	30.15
	NPN	26.60	39.17	31.69	15.75	20.34	17.75	24.33	40.10	30.29
G2	**SA-ELMo**	34.91	32.16	33.48	21.23	16.53	18.59	47.33	40.86	43.86
	SA-RBERT	43.58	30.03	35.55	24.75	18.00	20.84	46.51	45.45	45.97
	LC-BERT	43.05	41.31	42.16	29.46	28.67	29.06	49.36	52.09	50.69
	LC-RBERT	43.25	47.17	45.13	39.62	29.75	33.98	47.87	53.41	50.49
G3	**ENG**	43.87	51.13	47.22	37.66	36.20	36.92	48.96	56.07	52.27
	ENG+Mask	44.27	53.54	**48.47**	39.29	33.93	36.41	49.64	56.93	**53.03**
	ENG+Link	43.47	52.80	47.68	37.17	37.18	**37.18**	50.62	54.48	52.48
	ENG+Sent	45.29	50.89	47.93	36.69	36.14	36.41	49.48	57.12	**53.03**
G4	**ENG+IL**	40.90	58.03	47.98	31.67	41.19	35.81	49.93	74.95	59.93
	ENG+IL+ST	40.47	58.43	47.82	31.80	40.58	35.66	51.19	72.60	**60.04**

the qualitative analysis, we visualize and compare the contextualized graph embeddings and contextualized word embeddings.

7.4.1 Data and Experiment Settings

For TAPT, we use RocStories, as it has a decent amount of documents (90K after excluding the validation and testing sets) that share the text style of StoryCommonsense. For all tasks, we use the train/dev/test splits used in previous work.

All the RoBERTa models used in this paper are initialized with *roberta-base*, and the BERT models with *bert-base-uncased*. The maximum sequence length for the language models is 160; for large ENGs, we set the maximum number of nodes to 60; all the hidden layer have 128 hidden units; and the number of layers for R-GCN is 2.

For learning parameters in TAPT, we set the batch size to 256 through gradient accumulations; the optimizer is Adam [109] with an initial learning rate of $\texttt{1e}-4$, $\epsilon = \texttt{1e}-6$, $\beta = (0.9, 0.98)$, weight decay 0.01, and warm-up proportion 0.06. We run TAPT for 100 epochs. For the downstream tasks, we conduct a grid search of Adam's initial learning rate from $\{\texttt{2e}-3, \texttt{2e}-4, \texttt{2e}-5, \texttt{2e}-6\}$, 5000 warm-up steps, and stop patience of 10. Model selection is done on the validation set. We report results for the best model. For learning the potentials for symbolic inference with DRaiL [164], we use local normalization with a learning rate of 1e-3, and represent neural potentials using 2-layer Feed-Forward Networks over the ENG node embeddings. All hidden layers consist of 128 units. The parameters are learned using SGD with a patience of 5, tested against the validation set. For more details, refer to [164]. Note that while it would be possible to back-propagate to the whole graph, this is a computationally expensive procedure. We leave this exploration for future work.

7.4.2 Task: StoryCommonsense

StoryCommonsense consists of three subtasks: Maslow, Reiss, and Plutchik, introduced in Sec. 7.2. For each task, for each sentence-character pair in a given story, conduct multi-label classifications for each subtask. Each story was annotated by three annotators and the final labels were determined through a majority vote. For Maslow and Reiss, the vote is count-based, (i.e., if two out of three annotators flag a label, then it is an active label). For Plutchik, the vote is rating-based, where each label has an annotated rating, ranging from $\{0, 5\}$. If the averaged rating is larger or equal to 2, then it is an active label. This is the set-up given in the original paper [11]. Some papers [159] report results using the count-based majority vote, resulting in scores that are not comparable to ours. Therefore, we re-implement two recent strong models proposed for this task–the Label Correlation model (LC [159]) and the Self-Attention model (SA [40]) and evaluate them under the same set of hyper-parameters and model selection strategies as our models.

We briefly explain all the baselines, as well as our model variants shown in Table 7.2. The first group (G1) are the baselines proposed in the task paper. **TF-IDF** uses TF-IDF features, trained on RocStories, to represent the target sentence s and character context $ctx(c)$, and uses a Feed-Forward Net (FFN) classifier; **GloVe** encodes the sentences with the pretrained GloVe embeddings and learns uses a FFN; **CNN** [165] replaces the FFN with a Convolutional Neural Network; **LSTM** is a two-layer bi-directional LSTM; **REN** [154] is a recurrent entity network that learns to encode information for memory cells; and **NPN** [155] is an **REN** variant that includes a neural process network.

The second group (G2) of baselines are based on two recent publications–**LC** and **SA**–that showed strong performance on this task. We re-implement them and run the evaluation under the same setting as our proposed models. They originally use BERT and ELMo, respectively. To provide a fair comparisons, we also train a RoBERTa variant for them (LC-RBERT and SA-RBERT).

The third (G3) and fourth (G4) groups are our model variants. **ENG** is the model without TAPT; **ENG+Mask**, **ENG+Link**, and **ENG+Sent** are the models with Whole-Word-Masking (WM), Link Prediction (LP), and Node Sentiment (NS) TAPT, respectively. In the last group, **ENG(Best) + IL** and **ENG(Best) + IL + ST** are based on our best ENG model with TAPT and adding inter-label dependencies (IL) and state transitions (ST) using symbolic inference, described in Sec. 7.3.6.

Table 7.2 reports all the results. We can see that Group 2 generally performs better than Group 1 on all three subtasks, suggesting that our implementation is reasonable. Even without TAPT, **ENG** outperforms all baselines, rendering $2-3\%$ absolute F1-score improvement. With TAPT, the performance is further strengthened. Moreover, we find that different TAPT tasks offer different levels of improvement for each subtask. The WM helps the most in Maslow and Plutchik, while the LP and NS excel in Reiss and Plutchik, respectively. This means that different TAPTs embed different information needed for solving the subtask. For example, the ability to add potential edges can be key to do motivation reasoning (Reiss), while identifying sentiment polarities (NS) can help in emotion analysis (Plutchik). This observation suggests a direction of connecting different related tasks in a joint pipeline. We leave this for future work.

Lastly, we evaluate the impact of symbolic inference. We perform joint inference over the rules defined in Sec. 7.3.6. On Table 7.2, we can appreciate the advantage of modeling these dependencies for predicting Plutchik labels. However, the same is not true for the other two subtasks, where symbolic inference increases recall at the expense of precision, resulting in

Table 7.3. Results for the DesireDB task: identifying if a desire described in the document is fulfilled or not. Acronyms: Precision (P), Recall (R).

	Fulfilled			**Unfulfilled**			**Average**		
Models	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
ST-BOW	78.00	78.00	78.00	57.00	56.00	57.00	67.50	67.00	67.50
ST-All	78.00	79.00	79.00	58.00	56.00	57.00	68.0	67.50	68.00
ST-Disc	80.00	79.00	80.00	58.00	56.00	57.00	68.00	67.50	68.00
LR-BOW	69.00	65.00	67.00	53.00	57.00	55.00	61.00	61.00	61.00
LR-All	79.00	70.00	74.00	52.00	64.00	58.00	65.50	67.00	66.00
LR-Disc	75.00	84.00	80.00	60.00	45.00	52.00	67.50	64.50	66.00
BERT	81.75	75.90	78.72	57.95	66.23	61.82	69.85	71.06	70.27
BERT+ENG	81.99	83.06	**82.52**	65.33	63.64	**64.47**	73.66	73.35	**73.50**

no F1 improvement. Note that labels for Maslow and Reiss are sparser, accounting for 55% and 42% of the nodes, respectively. In contrast, Plutchik labels are present in 68% of the nodes.

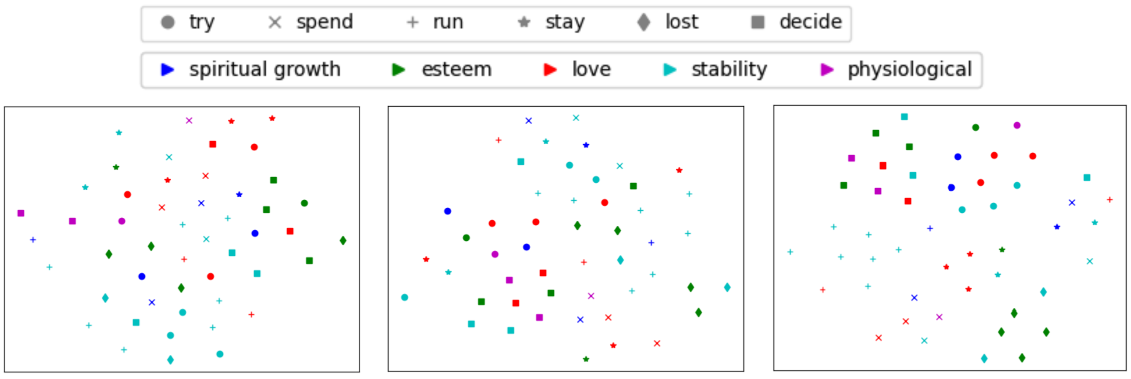
7.4.3 Task: DesireDB

DesireDB [71] is the task of predicting whether a given desire expression is fulfilled or not, given its prior and post context. It requires aggregating information from multiple parts of the document. If a target desire is "I want to be rich", and the character's mental changed from "sad" to "happy" along the text, we can infer that their desire is likely to be fulfilled.

We use the baseline systems described in [71], based on SkipThought (ST) and Logistic Regression (LR), with manually engineered lexical and discourse features. We train a stronger baseline by encoding the prior and post contexts, as well as the desire using BERT. Then, we add an attention layer (Eq. 7.5) for the two contexts over the desire expression. The resulting three representations (the weighted prior and post representations, and the desire representation) are then concatenated. For ENG, we add an attention layer over the nodes to form the ENG document representation. We compare BERT and BERT+ENG document representations by feeding each of them in to a two-layer FFN for classfications, as described in Sec. 7.3.4 (Doc. Classification).

Table 7.3 shows the result. The BERT baseline outperforms other baselines with a large gap, 4.27% absolute increase in the averaged F1-score. Furthermore, BERT+ENG forms a better document summary for the target desire, which further increase another absolute 3.23% on the avg. F1-score, which illustrates that ENG can be used in various settings for modeling entity information.

7.4.4 Qualitative Analysis



(a) Graph CTX (b) Word CTX (Story) (c) Word CTX (Sentence)

Figure 7.3. t-SNE visualization of embeddings based on ENG and RoBERTa. Six cherry-picked verbs are represented by shapes, and five Maslow labels are represented by colors. Acronyms: Contextualizaed (CTX).

We conduct qualitative analysis by measuring and visualizing distances between event nodes corresponding to six verbs and their Maslow labels. We project the node embeddings, based on different encoders, to a 2-D space using t-SNE [166]. We use shapes to represent verbs and colors to represent labels. In Fig. 7.3b and 7.3c, RoBERTa, pretrained on Whole-Word-Masking TAPT, was used. Node are word-contextualized, receiving the whole story (W-CTX-STORY) or the target sentence (W-CTX-SENT) as context. In these two cases, event nodes with the same verb (shape) tend to be closer. In Fig. 7.3a, we use ENG as the encoder to generate graph-contextualized embeddings (ENG-CTX). We observe that nodes with the same label (color) tend to be closer. In all cases, the embedding was trained using only the TAPT tasks, without task specific data. The ENG embedding are better at capturing entities' mental states, rather than verb information, as the graph structure is entity-driven.

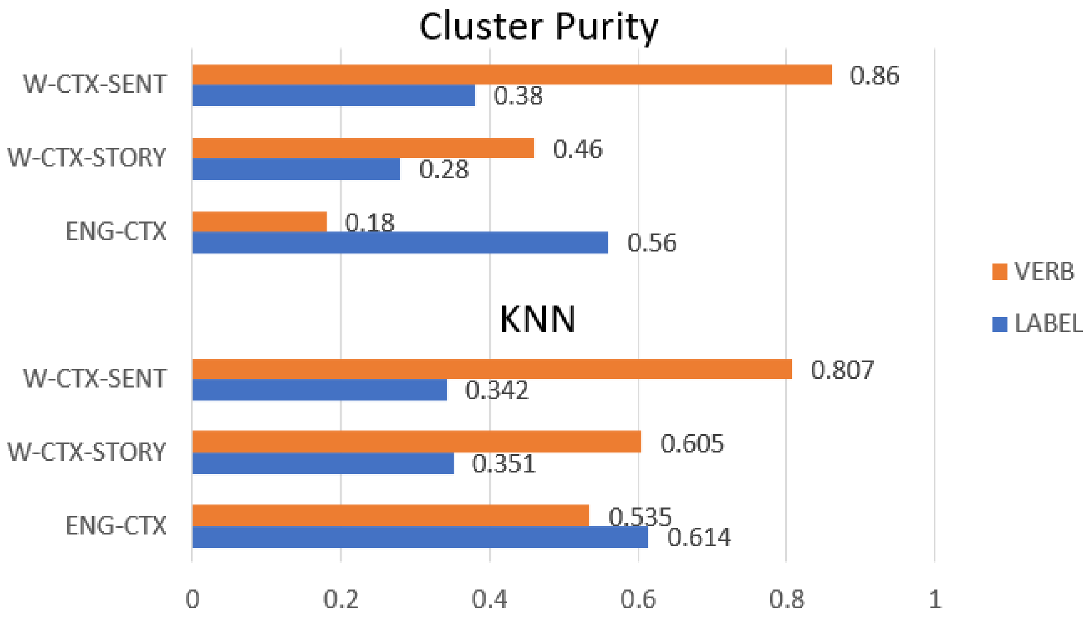


Figure 7.4. Cluster Purity and KNN Classification results for graph- and word-contextualized embeddings.

Figure 7.4 makes this point quantitatively. We use 10-fold cross validation and report averaged results. The proximity between verbs and between labels are measured in two ways: cluster purity and KNN classification. For the cluster purity [167], we cluster the events using K-Means ($K = 5$), and calculate the averaged cluster purity, defined as follows:

$$\frac{1}{N}\sum_{c\in C}\max_{d\in D}|c\cap d|, \tag{7.6}$$

where C is the set of clusters and D is either the set of labels or verbs. For the graph contextualization, we can see that the labels have higher cluster purity than the verbs, while for the word contextualization, the verbs have higher cluster purity. This result aligns with our visualization. The KNN classification uses the learned embedding as a distance function. The KNN classifier performs better when classifying labels using the graph-contextualized embeddings, and the vice-versa when classifying verbs, demonstrating that ENG helps capture entities' states better.

7.5 Conclusions

We propose a ENG model that can capture the implicit states of entities by multi-relational graph contextualization. We study three types of weakly-supervised TAPTs for ENG and their impact to downstream tasks. The evaluation includes two psychological commonsense inference tasks. The results shows that ENG can outperform other strong baselines, and can be benefit from different types of TAPT for different tasks. In future work, we want to connect different TAPT schemes and downstream tasks, and explore constrained representations.

8. CONCLUSIONS

"No matter what situations you experience in life, think about them as you draw conclusions about what the situations means for you."

—Sunday Adelaja

In this thesis, to address the recent surge of research about commonsense reasoning, we propose three novel techniques to improve statistical script learning for representing commonsense knowledge, and present them in four individual works. We briefly summarize them as follows:

Featured Event Embedding Learning (Chapter 4, [13]) is a multi-task script learning model, considering intra-event and inter-event objectives jointly and enriching event representations with entity-based features. We specifically analyze two sets of entity-based features—entities' sentiment and animacy—and identify their impacts on commonsense reasoning. Three intrinsic and two extrinsic evaluations are performed. For the intrinsic evaluations, in addition to the standard narrative cloze task—MCNC, we propose two sequential variants—MCNS and MCNE. MCNS addresses models' ability to make multi-step commonsense inferences, while MCNE conducts abductive reasoning to explain what things are happening in between two given events. For the extrinsic evaluations, we use the learned event representations as features in an implicit discourse sense classification task and a sentence semantic relatedness task. The experimental result denotes that FEEL can utilize fine-grained event properties to better account for event semantics.

Multi-relational Script Learning (Chapter 5, [14]) is the first attempt to account for multiple relation types between events. We consider two types of the NEXT relation, based on the text order and entity coreference chain, and pick nine discourse relations related to commonsense reasoning. We show that with a simple rule-based discourse annotator, we can extract meaningful discourse relations between events to conduct script learning. The intrinsic evaluations compare our two model variants—EventTransE and EventTransR—with other baselines, and show that the event representations can be generalized better when considering multiple relation types.

Narrative Graph (Chapter 6, [15]) introduces a graphical abstraction for events in a document. This is the first script learning model that contextualizes event representations in a grounded situation. The multi-typed contextualization greatly enhances the model's ability to capture long-range dependencies between events. Following our previous works, we compare our model variants with recently published script learning models in both intrinsic and extrinsic evaluations, and demonstrate the power of event contextualization.

Entity-based Narrative Graph (Chapter 7) is an extension to the Narrative Graph. It utilizes the three techniques we developed in the previous three works, incorporating the entity-based states, multi-typed relations, and event contextualization when constructing event representations. This work applies the model in psychological commonsense reasoning. Solving it requires understanding entities' mental states, and entity-to-event and event-to-event interactions. This work also examines different objectives used in the pretraining phase, such as link predictions and node classifications, extending the language model pretraining to more interesting setups in the field of script learning. To further improve the task performance, symbolic inferences are added on the output space to capture label dependencies. In the evaluations, we achieve the best results over our strong baselines, including the three works we mentioned above, in predicting character mental states and desire fulfillment. A qualitative analysis is conducted and shows that this model can adapt event representations toward related labels, while the word embeddings based on language models, such as BERT, cannot do that.

It is exciting to see the recent advances and popularity of using script learning to represent commonsense knowledge; however, from recent research, we observe two main limitations. First, *noise reductions* for event and relation extractions require more explorations. With recent enhancement of deep learning models, accurately learning patterns between events is no longer a big issue. Deep learning models have enough capacity to memorize a huge amount of patterns in languages, e.g., the GPT-3 language model [20] can write articles almost indistinguishable from what humans write. Script knowledge is supposed to be a subset of the universal language model. Recent works rely on NLP toolboxes to identify symbolic information for event extractions, which contains inevitable noises. We need to explore more reliable supervisions to extract event relations from text so that script learning can become a precise knowledge source for AI, or even human intelligence.

The second limitation is about increasing noise tolerance for graphical neural networks (GNN). GNN has been a hot research topic in recent years. The goal is to learn node representations that take into account topology and node properties. Recent works use fast approximations, such as spectral graph convolutions, to efficiently learn the structure through message propagation. However, from our experience we found that the models often suffer heavily from noise. Methods like regularization and factorization only offer limited help. Creating graph models with the ability to handle noise in a huge, complex, and possibly dynamic graph is a key direction for script learning.

We hope this thesis can encourage and motivate more researchers to join the community of statistical script learning. For future work, we suggest including an important concept—*schema*—into script learning. A schema describes a set of roles with their participated events or behaviors. For example, "a buyer *purchases* a seller's products" should appear in a commercial schema. There are early attempts that either consider schemas as clustered events [168]–[171] or identify schemas through path searching on knowledge graphs [172]. However, connecting schema knowledge with scripts remains a huge challenge. Although our Narrative Graph can model events in a grounded situation, it is still necessary to connect the situation to a specific schema such that the related knowledge can be involved for commonsense reasoning. We expect to see more works to take this challenge.

REFERENCES

[1] N. Chambers and D. Jurafsky, “Unsupervised learning of narrative event chains.,” in *ACL*, vol. 94305, 2008, pp. 789–797.

[2] M. Granroth-Wilding and S. Clark, “What happens next? event prediction using a compositional neural network model.,” in *AAAI*, 2016, pp. 2727–2733.

[3] D. Kahneman, “Maps of bounded rationality: A perspective on intuitive judgment and choice,” *Nobel prize lecture*, vol. 8, pp. 351–401, 2002.

[4] E. T. Higgins, “Activation: Accessibility, and salience,” *Social psychology: Handbook of basic principles*, pp. 133–168, 1996.

[5] R. Davis, H. Shrobe, and P. Szolovits, “What is a knowledge representation?” *AI magazine*, vol. 14, no. 1, pp. 17–17, 1993.

[6] D. B. Lenat, “Cyc: A large-scale investment in knowledge infrastructure,” *Communications of the ACM*, vol. 38, no. 11, pp. 33–38, 1995.

[7] C. F. Baker, C. J. Fillmore, and J. B. Lowe, “The berkeley framenet project,” in *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, Association for Computational Linguistics, 1998, pp. 86–90.

[8] L. Banarescu, C. Bonial, S. Cai, M. Georgescu, K. Griffitt, U. Hermjakob, K. Knight, P. Koehn, M. Palmer, and N. Schneider, “Abstract meaning representation for sembanking,” in *Proceedings of the 7th linguistic annotation workshop and interoperability with discourse*, 2013, pp. 178–186.

[9] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” *arXiv preprint arXiv:1301.3781*, 2013.

[10] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.

[11] H. Rashkin, A. Bosselut, M. Sap, K. Knight, and Y. Choi, *Modeling naive psychology of characters in simple commonsense stories*, 2018. arXiv: 1805.06533 [cs.CL].

[12] R. Prasad, E. Miltsakaki, N. Dinesh, A. Lee, A. Joshi, L. Robaldo, and B. L. Webber, “The penn discourse treebank 2.0 annotation manual,” 2007.

[13] I.-T. Lee and D. Goldwasser, “Feel: Featured event embedding learning,” *AAAI*, pp. 4840–4847, 2018.

[14] I.-T. Lee and D. Goldwasser, "Multi-relational script learning for discourse relations," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 4214–4226.

[15] I.-T. Lee, M. L. Pacheco, and D. Goldwasser, "Weakly-supervised modeling of contextualized event embedding for discourse relations," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, 2020, pp. 4962–4972.

[16] H. Liu and P. Singh, "Conceptnet—a practical commonsense reasoning tool-kit," *BT technology journal*, vol. 22, no. 4, pp. 211–226, 2004.

[17] M. Sap, R. LeBras, E. Allaway, C. Bhagavatula, N. Lourie, H. Rashkin, B. Roof, N. A. Smith, and Y. Choi, "Atomic: An atlas of machine commonsense for if-then reasoning," *arXiv preprint arXiv:1811.00146*, 2018.

[18] R. Speer, J. Chin, and C. Havasi, *Conceptnet 5.5: An open multilingual graph of general knowledge*, 2018. arXiv: 1612.03975 [cs.CL].

[19] A. Bosselut, H. Rashkin, M. Sap, C. Malaviya, A. Celikyilmaz, and Y. Choi, *Comet: Commonsense transformers for automatic knowledge graph construction*, 2019. arXiv: 1906.05317 [cs.CL].

[20] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, *Language models are few-shot learners*, 2020. arXiv: 2005.14165 [cs.CL].

[21] K. Pichotta and R. J. Mooney, "Statistical script learning with multi-argument events.," in *EACL*, vol. 14, 2014, pp. 220–229.

[22] Z. S. Harris, "Distributional structure," *Word*, vol. 10, no. 2-3, pp. 146–162, 1954.

[23] N. Weber, N. Balasubramanian, and N. Chambers, "Event representations with tensor-based compositions," *arXiv preprint arXiv:1711.07611*, 2017.

[24] T. Caselli and P. Vossen, "The event storyline corpus: A new benchmark for causal and temporal relation extraction," in *Proceedings of the Events and Stories in the News Workshop*, 2017, pp. 77–86.

[25] Q. Ning, Z. Feng, H. Wu, and D. Roth, "Joint reasoning for temporal and causal relations," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, 2018, pp. 2278–2288.

[26] L. Gao, P. K. Choubey, and R. Huang, “Modeling document-level causal structures for event causal relation identification,” 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)*, 2019, pp. 1808–1817.

[27] R. C. Schank and R. P. Abelson, “Scripts, plans, goals and understanding: An inquiry into human knowledge structures.,” 1977.

[28] N. Chambers and D. Jurafsky, “Unsupervised learning of narrative schemas and their participants,” in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, Association for Computational Linguistics, 2009, pp. 602–610.

[29] K. Pichotta and R. J. Mooney, “Statistical script learning with recurrent neural networks,” *EMNLP 2016*, p. 11, 2016.

[30] K. Pichotta and R. J. Mooney, “Learning statistical scripts with lstm recurrent neural networks.,” in *AAAI*, 2016, pp. 2800–2806.

[31] H. Peng and D. Roth, “Two discourse driven language models for semantics.,” in *ACL*, 2016.

[32] Z. Wang, Y. Zhang, and C.-Y. Chang, “Integrating order information and event relation for script event prediction,” in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 57–67.

[33] B. Jans, S. Bethard, I. Vulić, and M. F. Moens, “Skip n-grams and ranking functions for predicting script events,” in *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, Association for Computational Linguistics, 2012, pp. 336–344.

[34] N. Mostafazadeh, M. Roth, A. Louis, N. Chambers, and J. F. Allen, “Lsdsem 2017 shared task: The story cloze test,” *LSDSem 2017*, p. 46, 2017.

[35] S. Chaturvedi, H. Peng, and D. Roth, “Story comprehension for predicting what happens next,” in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 1604–1615.

[36] F. Heider, *The psychology of interpersonal relations*. Psychology Press, 1982.

[37] W. Sellars, *Science, perception, and reality*, ser. International library of philosophy and scientific method. Humanities Press, 1963. [Online]. Available: https://books.google.com/books?id=m1pFy1ba7rkC.

[38] M. McCloskey, A. Washburn, and L. Felch, "Intuitive physics: The straight-down belief and its origin.," *Journal of Experimental Psychology: Learning, Memory, and Cognition*, vol. 9, no. 4, p. 636, 1983.

[39] C. M. Judd, J. Westfall, and D. A. Kenny, "Treating stimuli as a random factor in social psychology: A new and comprehensive solution to a pervasive but largely ignored problem.," *Journal of personality and social psychology*, vol. 103, no. 1, p. 54, 2012.

[40] D. Paul and A. Frank, "Ranking and selecting multi-hop knowledge paths to better predict human needs," 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)*, Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 3671–3681. DOI: 10.18653/v1/N19-1368. [Online]. Available: https://www.aclweb.org/anthology/N19-1368.

[41] X. Wang, P. Kapanipathi, R. Musa, M. Yu, K. Talamadupula, I. Abdelaziz, M. Chang, A. Fokoue, B. Makni, N. Mattei, *et al.*, "Improving natural language inference using external knowledge in the science questions domain," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 7208–7215.

[42] B. Bi, C. Wu, M. Yan, W. Wang, J. Xia, and C. Li, "Incorporating external knowledge into machine reading for generative question answering," *arXiv preprint arXiv:1909.02745*, 2019.

[43] H. Rashkin, M. Sap, E. Allaway, N. A. Smith, and Y. Choi, "Event2mind: Commonsense inference on events, intents, and reactions," *arXiv preprint arXiv:1805.06939*, 2018.

[44] S. S. Tomkins, "Script theory: Differential magnification of affects.," in *Nebraska symposium on motivation*, University of Nebraska Press, 1978.

[45] M. Minsky, "A framework for representing knowledge," 1974.

[46] D. E. Rumelhart, "Notes on a schema for stories," in *Representation and understanding*, Elsevier, 1975, pp. 211–236.

[47] R. Rudinger, P. Rastogi, F. Ferraro, and B. Van Durme, "Script induction as language modeling.," in *EMNLP*, 2015, pp. 1681–1686.

[48] A. Mnih and G. Hinton, "Three new graphical models for statistical language modelling," in *Proceedings of the 24th international conference on Machine learning*, 2007, pp. 641–648.

[49] S. Zhao, Q. Wang, S. Massung, B. Qin, T. Liu, B. Wang, and C. Zhai, "Constructing and embedding abstract event causality networks from text snippets," in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, ACM, 2017, pp. 335–344.

[50] K. Pichotta and R. J. Mooney, "Using sentence-level lstm language models for script inference," *arXiv preprint arXiv:1604.02993*, 2016.

[51] J. W. Orr, P. Tadepalli, J. R. Doppa, X. Fern, and T. G. Dietterich, "Learning scripts as hidden markov models," *arXiv preprint arXiv:1809.03680*, 2018.

[52] Z. Li, X. Ding, and T. Liu, "Constructing narrative event evolutionary graph for script event prediction," *arXiv preprint arXiv:1805.05081*, 2018.

[53] S. Pradhan, A. Moschitti, N. Xue, O. Uryupina, and Y. Zhang, "Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes," in *Joint Conference on EMNLP and CoNLL-Shared Task*, 2012, pp. 1–40.

[54] E. Hovy, M. Marcus, M. Palmer, L. Ramshaw, and R. Weischedel, "Ontonotes: The 90% solution," in *Proceedings of the human language technology conference of the NAACL, Companion Volume: Short Papers*, 2006, pp. 57–60.

[55] K. Pichotta *et al.*, "Advances in statistical script learning," Ph.D. dissertation, 2017.

[56] H. Levesque, E. Davis, and L. Morgenstern, "The winograd schema challenge," in *Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*, Citeseer, 2012.

[57] K. Sakaguchi, R. L. Bras, C. Bhagavatula, and Y. Choi, *Winogrande: An adversarial winograd schema challenge at scale*, 2019. arXiv: `1907.10641 [cs.CL]`.

[58] R. Zellers, Y. Bisk, R. Schwartz, and Y. Choi, "Swag: A large-scale adversarial dataset for grounded commonsense inference," *arXiv preprint arXiv:1808.05326*, 2018.

[59] N. Mostafazadeh, N. Chambers, X. He, D. Parikh, D. Batra, L. Vanderwende, P. Kohli, and J. Allen, "A corpus and evaluation framework for deeper understanding of commonsense stories," *arXiv preprint arXiv:1604.01696*, 2016.

[60] M. Roemmele, C. A. Bejan, and A. S. Gordon, "Choice of plausible alternatives: An evaluation of commonsense causal reasoning.," in *AAAI spring symposium: logical formalizations of commonsense reasoning*, 2011, pp. 90–95.

[61] C. Bhagavatula, R. L. Bras, C. Malaviya, K. Sakaguchi, A. Holtzman, H. Rashkin, D. Downey, S. W.-t. Yih, and Y. Choi, *Abductive commonsense reasoning*, 2020. arXiv: `1908.05739 [cs.CL]`.

[62] Y. Bisk, R. Zellers, R. L. Bras, J. Gao, and Y. Choi, *Piqa: Reasoning about physical commonsense in natural language*, 2019. arXiv: 1911.11641 [cs.CL].

[63] M. Forbes and Y. Choi, "Verb physics: Relative physical knowledge of actions and objects," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vancouver, Canada: Association for Computational Linguistics, Jul. 2017, pp. 266–276. DOI: 10.18653/v1/P17-1025. [Online]. Available: https://www.aclweb.org/anthology/P17-1025.

[64] Y. Elazar, A. Mahabal, D. Ramachandran, T. Bedrax-Weiss, and D. Roth, *How large are lions? inducing distributions over quantitative attributes*, 2019. arXiv: 1906.01327 [cs.CL].

[65] Y. Ji, C. Tan, S. Martschat, Y. Choi, and N. A. Smith, "Dynamic entity representations in neural language models," *arXiv preprint arXiv:1708.00781*, 2017.

[66] Z. Yang, P. Blunsom, C. Dyer, and W. Ling, *Reference-aware language models*, 2017. arXiv: 1611.01628 [cs.CL].

[67] M. Henaff, J. Weston, A. Szlam, A. Bordes, and Y. LeCun, *Tracking the world state with recurrent entity networks*, 2017. arXiv: 1612.03969 [cs.CL].

[68] C. Kiddon, L. Zettlemoyer, and Y. Choi, "Globally coherent text generation with neural checklist models," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Austin, Texas: Association for Computational Linguistics, Nov. 2016, pp. 329–339. DOI: 10.18653/v1/D16-1032. [Online]. Available: https://www.aclweb.org/anthology/D16-1032.

[69] A. Bosselut, O. Levy, A. Holtzman, C. Ennis, D. Fox, and Y. Choi, *Simulating action dynamics with neural process networks*, 2018. arXiv: 1711.05313 [cs.CL].

[70] M. Sap, H. Rashkin, D. Chen, R. LeBras, and Y. Choi, *Socialiqa: Commonsense reasoning about social interactions*, 2019. arXiv: 1904.09728 [cs.CL].

[71] E. Rahimtoroghi, J. Wu, R. Wang, P. Anand, and M. Walker, "Modelling protagonist goals and desires in first-person narrative," in *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, Saarbrücken, Germany: Association for Computational Linguistics, Aug. 2017, pp. 360–369. DOI: 10.18653/v1/W17-5543. [Online]. Available: https://www.aclweb.org/anthology/W17-5543.

[72] N. Xue, H. T. Ng, S. Pradhan, A. Rutherford, B. L. Webber, C. Wang, and H. Wang, "Conll 2016 shared task on multilingual shallow discourse parsing.," in *CoNLL Shared Task*, 2016, pp. 1–19.

[73] J. Pustejovsky, P. Hanks, R. Sauri, A. See, R. Gaizauskas, A. Setzer, D. Radev, B. Sundheim, D. Day, L. Ferro, *et al.*, "The timebank corpus," in *Corpus linguistics*, Lancaster, UK., vol. 2003, 2003, p. 40.

[74] T. O'Gorman, K. Wright-Bettner, and M. Palmer, "Richer event description: Integrating event coreference with temporal, causal and bridging annotation," in *Proceedings of the 2nd Workshop on Computing News Storylines (CNS 2016)*, Austin, Texas: Association for Computational Linguistics, Nov. 2016, pp. 47–56. DOI: 10.18653/v1/W16-5706. [Online]. Available: https://www.aclweb.org/anthology/W16-5706.

[75] Y. Meng and A. Rumshisky, "Context-aware neural model for temporal information extraction," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 527–536.

[76] Q. Ning, H. Wu, and D. Roth, "A multi-axis annotation scheme for event temporal relations," in *ACL*, Jul. 2018. [Online]. Available: http://cogcomp.org/papers/NingWuRo18.pdf.

[77] I. Mani, M. Verhagen, B. Wellner, C. Lee, and J. Pustejovsky, "Machine learning of temporal relations," in *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, 2006, pp. 753–760.

[78] N. UzZaman, H. Llorens, L. Derczynski, J. Allen, M. Verhagen, and J. Pustejovsky, "Semeval-2013 task 1: Tempeval-3: Evaluating time expressions, events, and temporal relations," in *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, 2013, pp. 1–9.

[79] N. Chambers, S. Wang, and D. Jurafsky, "Classifying temporal relations between events," in *Proceedings of the 45th annual meeting of the association for computational linguistics companion volume proceedings of the demo and poster sessions*, 2007, pp. 173–176.

[80] M. Verhagen and J. Pustejovsky, "Temporal processing with the tarsqi toolkit," in *COLING 2008: Companion Volume: Demonstrations*, 2008, pp. 189–192.

[81] S. Bethard, "ClearTK-TimeML: A minimalist approach to TempEval 2013," in *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, Atlanta, Georgia, USA: Association for Computational Linguistics, Jun. 2013, pp. 10–14. [Online]. Available: https://www.aclweb.org/anthology/S13-2002.

[82] N. Laokulrat, M. Miwa, Y. Tsuruoka, and T. Chikayama, "Uttime: Temporal relation classification using deep syntactic features," in *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, 2013, pp. 88–92.

[83] N. Chambers, "Navytime: Event and time ordering from raw text," NAVAL ACADEMY ANNAPOLIS MD, Tech. Rep., 2013.

[84] N. Chambers, T. Cassidy, B. McDowell, and S. Bethard, "Dense event ordering with a multi-pass architecture," *Transactions of the Association for Computational Linguistics*, vol. 2, pp. 273–284, 2014.

[85] K. Yoshikawa, S. Riedel, M. Asahara, and Y. Matsumoto, "Jointly identifying temporal relations with markov logic," in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, 2009, pp. 405–413.

[86] Q. Ning, Z. Feng, and D. Roth, "A structured learning approach to temporal relation extraction," *arXiv preprint arXiv:1906.04943*, 2019.

[87] A. Leeuwenberg and M. F. Moens, "Structured learning for temporal relation extraction from clinical records," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, 2017, pp. 1150–1158.

[88] Y. Meng, A. Rumshisky, and A. Romanov, "Temporal information extraction for question answering using syntactic dependencies in an lstm-based architecture," *arXiv preprint arXiv:1703.05851*, 2017.

[89] B. Zhou, Q. Ning, D. Khashabi, and D. Roth, *Temporal common sense acquisition with minimal supervision*, 2020. arXiv: `2005.04304 [cs.CL]`.

[90] R. Han, Q. Ning, and N. Peng, "Joint event and temporal relation extraction with shared representations and structured prediction," *arXiv preprint arXiv:1909.05360*, 2019.

[91] Q. Ning, Z. Feng, H. Wu, and D. Roth, "Joint reasoning for temporal and causal relations," *arXiv preprint arXiv:1906.04941*, 2019.

[92] P. Mirza and S. Tonelli, "An analysis of causality between events and its relation to temporal information," in *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, 2014, pp. 2097–2106.

[93] N. Mostafazadeh, A. Grealish, N. Chambers, J. Allen, and L. Vanderwende, “Caters: Causal and temporal relation scheme for semantic annotation of event structures,” in *Proceedings of the Fourth Workshop on Events*, 2016, pp. 51–61.

[94] Q. Do, Y. S. Chan, and D. Roth, “Minimally supervised event causality identification,” in *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, 2011, pp. 294–303.

[95] C. Hidey and K. McKeown, “Identifying causal relations using parallel Wikipedia articles,” in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 1424–1433. DOI: 10.18653/v1/P16-1135. [Online]. Available: https://www.aclweb.org/anthology/P16-1135.

[96] C. J. Fillmore, “The mechanisms of “construction grammar”,” in *Annual Meeting of the Berkeley Linguistics Society*, vol. 14, 1988, pp. 35–55.

[97] A. Modi and I. Titov, “Inducing neural models of script knowledge.,” in *CoNLL*, vol. 14, 2014, pp. 49–57.

[98] A. Modi, I. Titov, V. Demberg, A. Sayeed, and M. Pinkal, “Modeling semantic expectation: Using script knowledge for referent prediction,” *TACL*, 2017.

[99] S. Ahrendt and V. Demberg, “Improving event prediction by representing script participants.,” in *HLT-NAACL*, 2016, pp. 546–551.

[100] J. Li, A. Ritter, and D. Jurafsky, “Learning multi-faceted representations of individuals from heterogeneous evidence using neural networks,” *arXiv preprint arXiv:1510.05198*, 2015.

[101] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky, “The stanford corenlp natural language processing toolkit.,” in *ACL (System Demonstrations)*, 2014, pp. 55–60.

[102] C. J. Hutto and E. Gilbert, “Vader: A parsimonious rule-based model for sentiment analysis of social media text,” in *Eighth international AAAI conference on weblogs and social media*, 2014.

[103] S. Bird, E. Klein, and E. Loper, *Natural language processing with Python: analyzing text with the natural language toolkit.* ” O’Reilly Media, Inc.”, 2009.

[104] R. Parker, D. Graff, J. Kong, K. Chen, and K. Maeda, “English gigaword fifth edition ldc2011t07. dvd,” *Philadelphia: Linguistic Data Consortium*, 2011.

[105] A. Viterbi, “Error bounds for convolutional codes and an asymptotically optimum decoding algorithm,” *IEEE transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, 1967.

[106] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global vectors for word representation,” in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543. [Online]. Available: http://www.aclweb.org/anthology/D14-1162.

[107] M. Marelli, L. Bentivogli, M. Baroni, R. Bernardi, S. Menini, and R. Zamparelli, “Semeval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment.,” in *SemEval@ COLING*, 2014, pp. 1–8.

[108] K. S. Tai, R. Socher, and C. D. Manning, “Improved semantic representations from tree-structured long short-term memory networks,” *arXiv preprint arXiv:1503.00075*, 2015.

[109] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[110] 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*, 2013, pp. 2787–2795.

[111] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, “Learning entity and relation embeddings for knowledge graph completion.,” in *AAAI*, vol. 15, 2015, pp. 2181–2187.

[112] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling,” *arXiv preprint arXiv:1412.3555*, 2014.

[113] R. Kiros, Y. Zhu, R. R. Salakhutdinov, R. Zemel, R. Urtasun, A. Torralba, and S. Fidler, “Skip-thought vectors,” in *NIPS*, 2015, pp. 3294–3302.

[114] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[115] Z. Wang, J. Zhang, J. Feng, and Z. Chen, “Knowledge graph embedding by translating on hyperplanes.,” in *AAAI*, vol. 14, 2014, pp. 1112–1119.

[116] R. Xie, Z. Liu, and M. Sun, “Representation learning of knowledge graphs with hierarchical types.,” in *IJCAI*, 2016, pp. 2965–2971.

[117] H.-G. Yoon, H.-J. Song, S.-B. Park, and S.-Y. Park, “A translation-based knowledge graph embedding preserving logical property of relations,” in *NAACL*, 2016.

[118] M. Nickel, L. Rosasco, T. A. Poggio, *et al.*, "Holographic embeddings of knowledge graphs.," in *AAAI*, vol. 2, 2016, pp. 3–2.

[119] H. Peng and D. Roth, "Two discourse driven language models for semantics," *arXiv preprint arXiv:1606.05679*, 2016.

[120] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365*, 2018.

[121] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML*, 2010, pp. 807–814.

[122] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *JMLR*, vol. 12, no. Jul, pp. 2121–2159, 2011.

[123] R. Socher, D. Chen, C. D. Manning, and A. Ng, "Reasoning with neural tensor networks for knowledge base completion," in *NIPS*, 2013.

[124] M. L. Pacheco, I.-T. Lee, X. Zhang, A. K. Zehady, P. Daga, D. Jin, A. Parolia, and D. Goldwasser, "Adapting event embedding for implicit discourse relation recognition," *Proceedings of the CoNLL-16 shared task*, pp. 136–142, 2016.

[125] J. Wang and M. Lan, "Two end-to-end shallow discourse parsers for english and chinese in conll-2016 shared task," *Proceedings of the CoNLL-16 shared task*, pp. 33–40, 2016.

[126] A. Rutherford and N. Xue, "Robust non-explicit neural discourse parser in english and chinese," *Proceedings of the CoNLL-16 shared task*, pp. 55–59, 2016.

[127] D. Goldwasser and X. Zhang, "Understanding satirical articles using common-sense," *Transactions of the Association for Computational Linguistics*, vol. 4, pp. 537–549, 2016.

[128] 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*, Springer, 2018, pp. 593–607.

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

[130] C. Walker, S. Strassel, J. Medero, and K. Maeda, "Ace 2005 multilingual training corpus," *Linguistic Data Consortium, Philadelphia*, vol. 57, p. 45, 2006.

[131] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 701–710.

[132] A. Grover and J. Leskovec, "Node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 2016, pp. 855–864.

[133] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *Proceedings of the 24th international conference on world wide web*, 2015, pp. 1067–1077.

[134] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in neural information processing systems*, 2017, pp. 1024–1034.

[135] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[136] D. Marcheggiani and I. Titov, "Encoding sentences with graph convolutional networks for semantic role labeling," *arXiv preprint arXiv:1703.04826*, 2017.

[137] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, JMLR. org, 2017, pp. 933–941.

[138] C. Li and D. Goldwasser, "Encoding social information with graph convolutional networks forpolitical perspective detection in news media," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 2594–2604.

[139] Z.-M. Zhou, Y. Xu, Z.-Y. Niu, M. Lan, J. Su, and C. L. Tan, "Predicting discourse connectives for implicit discourse relation recognition," in *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, Association for Computational Linguistics, 2010, pp. 1507–1514.

[140] J. Park and C. Cardie, "Improving implicit discourse relation recognition through feature set optimization," in *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, Association for Computational Linguistics, 2012, pp. 108–112.

[141] O. Biran and K. McKeown, "Aggregated word pair features for implicit discourse relation disambiguation," 2013.

[142] E. Malmi, D. Pighin, S. Krause, and M. Kozhevnikov, "Automatic prediction of discourse connectives," *CoRR*, vol. abs/1702.00992, 2017. arXiv: 1702.00992. [Online]. Available: http://arxiv.org/abs/1702.00992.

[143] A. Nie, E. Bennett, and N. Goodman, "Dissent: Learning sentence representations from explicit discourse relations," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 4497–4510.

[144] D. Sileo, T. Van-De-Cruys, C. Pradel, and P. Muller, "Mining discourse markers for unsupervised sentence representation learning," *arXiv preprint arXiv:1903.11850*, 2019.

[145] K.-W. Chang, W.-t. Yih, B. Yang, and C. Meek, "Typed tensor decomposition of knowledge bases for relation extraction," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1568–1579.

[146] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[147] I. Turc, M.-W. Chang, K. Lee, and K. Toutanova, "Well-read students learn better: On the importance of pre-training compact models," 2019.

[148] M. Abdul-Mageed and L. Ungar, "Emonet: Fine-grained emotion detection with gated recurrent neural networks," in *Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: Long papers)*, 2017, pp. 718–728.

[149] S. Chaturvedi, D. Goldwasser, and H. Daumé III, "Ask, and shall you receive? understanding desire fulfillment in natural language text," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, 2016, pp. 2697–2703.

[150] W. G. Lehnert, "Plot units and narrative summarization," *Cognitive science*, vol. 5, no. 4, pp. 293–331, 1981.

[151] A. Goyal, E. Riloff, and H. Daumé III, "Automatically producing plot unit representations for narrative text," in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, 2010, pp. 77–86.

[152] D. Elson, "Dramabank: Annotating agency in narrative discourse.," in *LREC*, 2012, pp. 2813–2819.

[153] Y. Ji, C. Tan, S. Martschat, Y. Choi, and N. A. Smith, "Dynamic entity representations in neural language models," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark: Association for Computational Linguistics, Sep. 2017, pp. 1830–1839. DOI: 10.18653/v1/D17-1195. [Online]. Available: https://www.aclweb.org/anthology/D17-1195.

[154] M. Henaff, J. Weston, A. Szlam, A. Bordes, and Y. LeCun, "Tracking the world state with recurrent entity networks," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, OpenReview.net, 2017. [Online]. Available: https://openreview.net/forum?id=rJTKKKqeg.

[155] A. Bosselut, O. Levy, A. Holtzman, C. Ennis, D. Fox, and Y. Choi, "Simulating action dynamics with neural process networks," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, OpenReview.net, 2018. [Online]. Available: https://openreview.net/forum?id=rJYFzMZC-.

[156] A. H. Maslow, "A theory of human motivation," *Psychological Review*, vol. 50, pp. 370–396, 1943. [Online]. Available: http://doi.apa.org/index.cfm?fuseaction=showUIDAbstract&uid=1943-03751-001.

[157] S. Reiss, "Multifaceted nature of intrinsic motivation: The theory of 16 basic desires," *Review of General Psychology*, vol. 8, pp. 179–193, Sep. 2004. DOI: 10.1037/1089-2680.8.3.179.

[158] R. Plutchik, "A general psychoevolutionary theory of emotion," *Theories of emotion*, vol. 1, pp. 3–31, 1980.

[159] R. Gaonkar, H. Kwon, M. Bastan, N. Balasubramanian, and N. Chambers, "Modeling label semantics for predicting emotional reactions," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 4687–4692. DOI: 10.18653/v1/2020.acl-main.426. [Online]. Available: https://www.aclweb.org/anthology/2020.acl-main.426.

[160] Z. Li, X. Ding, and T. Liu, "Constructing narrative event evolutionary graph for script event prediction," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, J. Lang, Ed., ijcai.org, 2018, pp. 4201–4207. DOI: 10.24963/ijcai.2018/584. [Online]. Available: https://doi.org/10.24963/ijcai.2018/584.

[161] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, *Roberta: A robustly optimized bert pretraining approach*, 2019. arXiv: 1907.11692 [cs.CL].

[162] S. Gururangan, A. Marasović, S. Swayamdipta, K. Lo, I. Beltagy, D. Downey, and N. A. Smith, *Don't stop pretraining: Adapt language models to domains and tasks*, 2020. arXiv: 2004.10964 [cs.CL].

[163] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[164] M. L. Pacheco and D. Goldwasser, "Modeling content and context with deep relational learning," in *Transactions of the Association for Computational Linguistics (TACL)*, 2020.

[165] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[166] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[167] C. D. Manning, H. Schütze, and P. Raghavan, *Introduction to information retrieval.* Cambridge university press, 2008.

[168] N. Chambers, "Event schema induction with a probabilistic entity-driven model," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 2013, pp. 1797–1807.

[169] K.-H. Nguyen, X. Tannier, O. Ferret, and R. Besançon, "Generative event schema induction with entity disambiguation," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2015, pp. 188–197.

[170] L. Sha, S. Li, B. Chang, and Z. Sui, *Joint learning templates and slots for event schema induction*, 2016. arXiv: 1603.01333 [cs.CL].

[171] L. Huang, T. Cassidy, X. Feng, H. Ji, C. Voss, J. Han, and A. Sil, "Liberal event extraction and event schema induction," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016, pp. 258–268.

[172] F. Christopoulou, M. Miwa, and S. Ananiadou, *Connecting the dots: Document-level neural relation extraction with edge-oriented graphs*, 2019. arXiv: 1909.00228 [cs.CL].

VITA

I-Ta Lee was born in Taipei city, Taiwan, in 1985. He completed a B.S. (with presidential rewards) in the Computer Science and Engineering Department at Yuan Ze University in Taoyuan, Taiwan, in 2008. He completed an M.S. under the guidance of Dr. Shun-Ren Yang, in the Department of Computer Science at National Tsing Hua University in Hsinchu, Taiwan, in 2010, and is an Honorary Member of the Phi Tau Phi Scholastic Honor Society. He was a Software Engineer and served in the industry from 2010 to 2014 at Moxa Inc. and TrendMicro Inc.. He completed his Ph.D. under the advice of Dr. Dan Goldwasser in the Department of Computer Science at Purdue University in 2020.

I-Ta specializes in handling the problems that require Natural Language Understanding (NLU) and Machine Learning techniques. He hopes to share expertise through innovative works with engineers and information scientists all over the global. I-Ta's research interests mainly concern in modeling human commonsense to refine NLU. He has in-depth experiences about creating commonsense models with Statistical Script Learning, whose core concept is predicting "what happens next." To put it concisely, the models predict the next event given what has happened in the past. Imagine that your Siri can guess what you want without explicit commands. His ultimate goal is to build commonsense for machines in real-world applications so that natural machine-human communications can happen.

PUBLICATIONS

I-Ta Lee, Maria L. Pacheco, and Dan Goldwasser. *“Modeling Psychological States through An Entity-Based Narrative Graph.”* Under Review. 2021

I-Ta Lee, Maria L. Pacheco, and Dan Goldwasser. *”Weakly-Supervised Modeling of Contextualized Event Embedding for Discourse Relations.”* Findings of The 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2020

Lee, I-Ta, and Dan Goldwasser. *”Multi-relational script learning for discourse relations.”* Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL). 2019.

I-Ta Lee and Dan Goldwasser. *“FEEL: Featured Event Embedding Learning.”* The 32th AAAI Conference on Artificial Intelligence (AAAI). 2018