TOWARDS COMMONSENSE REASONING WITH HIGHER-ORDER SELECTIONAL PREFERENCE OVER EVENTUALITIES

by

HONGMING ZHANG

A Thesis Presented to
The Hong Kong University of Science and Technology
in Partial Fulfillment of the Requirements for
the Degree of Doctor of Philosophy
in Computer Science and Engineering

August 2021, Hong Kong

ACKNOWLEDGMENTS

During the past three years, I think I am one of the luckiest guys in the world. I am lucky to work with a knowledgeable and considerable supervisor; I am lucky to work on the research topics that I am interested in; I am lucky to have the full support of my family and friends. Understanding commonsense is one of the ultimate goals of the whole artificial intelligence community. Since day one of my Ph.D. journey, I know that I could not fully solve this problem during my Ph.D. study, yet I still chose to work on it. Now, at the end of my Ph.D. study, I hope that my efforts could help shed some light on this exciting research problem, and I hope I will have the chance to keep working on it in the future.

The first person I want to thank is my supervisor, Prof. Yangqiu Song. He is always responsible, considerable, and encouraging. Every time I encountered a problem, no matter it is related to the research or not, I can always trust him and get his help. Without his support, I am not sure if I will have the courage to choose such a challenging topic and keep working on it. Thanks.

The second person I want to thank is Prof. Dan Roth. During my visit to UPenn, he does not only teach me how to do better research but also teach me how to enjoy research. Even though the whole visit is during the big pandemic, and we do not have too many chances to meet face-to-face, I still learn a lot from him during those virtual discussions. I will miss the days in Philly.

I also want to thank my mentors during the past three years (sorted by name): Prof. Muhao Chen, Prof. Yoav Goldberg, Dr. Daniel Khashabi, and Prof. Yan Song. Their knowledge and expertise help me grow fast as a mature researcher.

Of course, I could not forget all the help I received from my collaborators (sorted by name): Mr. Jiaxin Bai, Mr. Hantian Ding, Mr. Yanai Elazar, Ms. Yintong Huo, Mr. Zizheng Lin, Mr. Xin Liu, Ms. Qing Lyu, Ms. Nedjma Ousidhoum, Mr. Haojie Pan, Dr. Elior Sulem, Mr. Haoyu Wang, Mr. Changlong Yu, Mr. Xintong Yu, and Mr. Xinran Zhao. I really enjoyed and learned a lot from working with them.

In the end, I want to thank my family, especially my girlfriend, Ms. Junru Song. We have been in a long-distance relationship for nine years, and she always encourages me to do whatever I want. Choosing to be a Ph.D. is not an easy choice. Finishing it is an even harder one. But with her endless love and support, I can say that I enjoyed the whole journey. Thanks.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

TOWARDS COMMONSENSE REASONING WITH HIGHER-ORDER SELECTIONAL PREFERENCE OVER EVENTUALITIES

by

HONGMING ZHANG

Department of Computer Science and Engineering

The Hong Kong University of Science and Technology

ABSTRACT

Commonsense reasoning has long been a challenging yet vital artificial intelligence problem. In the past few decades, many efforts have been devoted to investigating how to represent, acquire, and apply commonsense knowledge to understand human language. Recently, with the help of large-scale pre-trained language models, the community has made significant progress on many commonsense reasoning benchmarks. However, due to the non-explainability of deep models, it is still unclear whether current models are solving these commonsense reasoning tasks with the correct reason or not. In this thesis, we first conduct experiments to show that even though current deep models can achieve high performance on the original WSC task, they cannot efficiently distinguish the right reasons.

An important reason behind this is that we are lack of a good commonsense representation methodology and a principled commonsense inference methodology. To fill this gap, in this thesis, we propose a new commonsense representation methodology, higher-order selectional preference over eventualities, to model the complex commonsense in our daily life. Following this principle, we developed a scalable eventuality-centric commonsense knowledge extraction pipeline. As a result, we created ASER, which is the largest eventuality-centric knowledge graph in the world.

Specifically, it contains 438 million eventualities and 648 million edges among them. Both intrinsic and extrinsic evaluations were conducted to demonstrate the high quality of ASER. We also conduct further experiments to prove the transferability from the selectional preference knowledge in ASER to human-defined commonsense.

On top of ASER, we also explored how to acquire commonsense knowledge from different modalities. Specifically, we first propose a multiplex word embedding model to acquire selectional preference knowledge from text more efficiently. After that, we also introduce how we can leverage the help of analogy and eventuality conceptualization to generalize the knowledge about observed eventualities to unseen ones. Last but not least, we investigate the possibility of acquiring causal knowledge about daily eventualities from the visual signal.

After collecting the knowledge, we also need to apply the structured commonsense for downstream natural language understanding tasks. Thus, in the last part of this thesis, we first use pronoun coreference resolution as the downstream task to investigate how to jointly use the structured knowledge and language representation models for better language understanding. In the end, to explore a commonsense inference model that can be applied to all downstream commonsense reasoning tasks, we propose a novel learning paradigm: commonsense knowledge base commonsense reasoning (CKBQA). Experiments on CKBQA show that even though inference over commonsense knowledge is challenging, models can learn to conduct simple inference after training with a few examples. Besides that, the learned model also demonstrates the generalization ability across tasks, which was not observed in previous commonsense reasoning models.

CHAPTER 1

INTRODUCTION

1.1 The Importance and Challenges of Understanding Commonsense

1.1.1 Definition of Commonsense and its Importance

Understanding natural language requires both the linguistic knowledge of the vocabulary and grammar of a language and the complex semantic knowledge about the world we live in [60]. In his conceptual semantics theory, Ray Jackendoff, a Rumelhart Prize[1] winner, describes semantic meaning as "a finite set of mental primitives and a finite set of principles of mental combination [57]." The primitive units of semantic meanings in human language include *Thing* (or *Object*), *Activity*,[2] *State*, *Event*, *Place*, *Path*, *Property*, *Amount*, etc. Understanding the semantics related to the world requires the understanding of these units and their relations.

To help machines understand those semantic units, linguists and domain experts built knowledge graphs (KGs)[3] to formalize these units and enumerate categories (or senses) and relations of them. Considering that the scale of world knowledge could be enormous, these knowledge graphs have to be large-scale to have decent coverage. Motivated by this, many automatic or semi-automatic information extraction systems have been developed to extract knowledge from large-scale web contents and thus construct large-scale knowledge graphs. Several popular examples are Freebase [10], KnowItAll [37], TextRunner [6], YAGO [157], DBpedia [2], NELL [15], Probase [172], and Google Knowledge Vault [26]. Even though these automatic and semi-automatic methods guarantee the scale of the resulting knowledge graphs, they mostly focus on factual knowledge about name entities or terms (e.g., "Paris" is the capital of "France"). In contrast, a lot of

[1]The David E. Rumelhart Prize is funded for contributions to the theoretical foundations of human cognition.

[2]In his original book, he called it *Action*. But given the other definitions and terminologies we adopted [105, 3], it means *Activity*.

[3]Traditionally, people used the term "knowledge base" to describe the database containing human knowledge. In 2012, Google released its knowledge graph where vertices and edges in a knowledge base are emphasized. For more information about terminologies, please refer to [31].

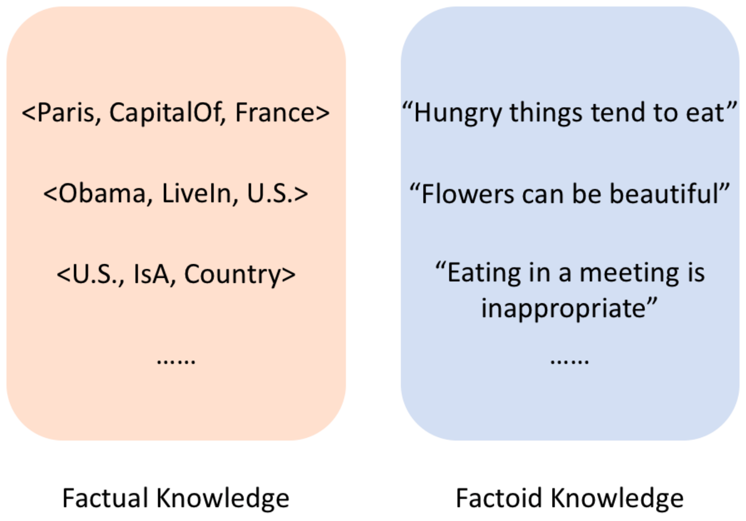

<Paris, CapitalOf, France>
<Obama, LiveIn, U.S.>
<U.S., IsA, Country>
……
"Hungry things tend to eat"
"Flowers can be beautiful"
"Eating in a meeting is inappropriate"
……
Factual Knowledge
Factoid Knowledge

Figure 1.1: Example of factual and factoid knowledge. Factual knowledge is fact and is typically about named entities. As a comparison, factoid knowledge is often just the preference about daily entities or events. Both Factual knowledge and factoid knowledge are crucial for commonsense reasoning.

factoid knowledge [43, 42] about daily entities or events (e.g., "hungry things tend to eat") are omitted.

Examples of factual knowledge and factoid knowledge are shown in Figure 1.1, from which we can see that factual knowledge is fact and is typically about named entities, while factoid knowledge is often just the preference about daily entities or events. Compared with factual knowledge, as a crucial part of the commonsense, factoid knowledge typically is rarely explicitly stated in the text because they are too trivial for human beings, yet they bring significant challenges for machines to conduct commonsense reasoning and to understand human language. For example, when a user says "I am hungry" to a chatbot at 1:00 pm, the chatbot should understand that the user may want to have lunch rather than breakfast and recommend some nearby restaurants. This requires the chatbot to understand the intricate commonsense knowledge about user's states and potential consequent activities (i.e., being hungry can motivate the user to eat) and the implications of location and time (i.e., compared with breakfast, lunch is more likely to appear at 1:00 pm. Thus the chatbot should recommend some real food rather than just a cup of coffee).

In this thesis, to help machines better conduct commonsense reasoning, we focus on the factoid commonsense, which is often omitted by previous efforts. Specifically, we formally define the commonsense we care about in this thesis as follows.

Definition 1.1.1 *Factoid commonsense is preference knowledge about daily* ***entity****,* ***state****, and* ***event****.*

Even though factoid commonsense is often a kind of preference rather than inevitably facts, it plays a critical role in understanding the physical world and human language.

1.1.2 Limitation of Existing Commonsense Reasoning Methodologies

Commonsense reasoning has long been a challenging problem in the artificial intelligence field. Throughout the years, many efforts have been devoted to helping machines understand commonsense with different methodologies.

For example, ConceptNet [84] tries to represent commonsense knowledge with triplets over free-format concepts and leverage crowdsourcing to collect high-quality commonsense triplets. However, three limitations of this methodology limit its usage on downstream tasks. First, such an approach can only cover selected commonsense relations such as "What is capable of doing something" or "What is used for doing something," while missing more complex commonsense such as "we often make a call before we leave." Due to the complexity of commonsense, it is infeasible to manually define all commonsense relations and then collect knowledge for each of them. Besides that, as all the included relations are defined by experts, it is hard to employ an information extraction system to collect such commonsense knowledge automatically. As a result, we can only leverage human annotation, which could be expensive and lead to the scale problem. Even though the latest version of ConceptNet (i.e., ConceptNet 5.0 [153]) contains over 600 thousand commonsense triplets, there is still a big gap between the current progress and covering all the commonsense. Lastly, as aforementioned, commonsense is more likely to be a kind of preference rather than fixed knowledge. However, the fixed triplets can only represent whether a combination is correct or not rather than how likely it will be correct given certain circumstances.

Besides representing commonsense with fixed triplets, another line of effort is representing commonsense with language models. With the help of pre-trained language models, rich contextualized knowledge is encoded into the vector space. Recently, the community has made impressive performance on several commonsense reasoning benchmarks following this methodology. However, as we still need to fine-tune these language models with a large-scale of training data so that they can work and these models are typically not explainable, it is hard for us to tell why these models can solve the questions (i.e., Whether it is because it already has the commonsense rea-

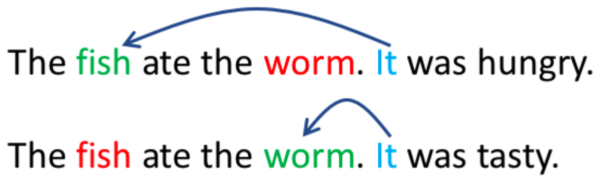

The fish ate the worm. It was hungry.
The fish ate the worm. It was tasty.

Figure 1.2: A pair of questions in WSC.

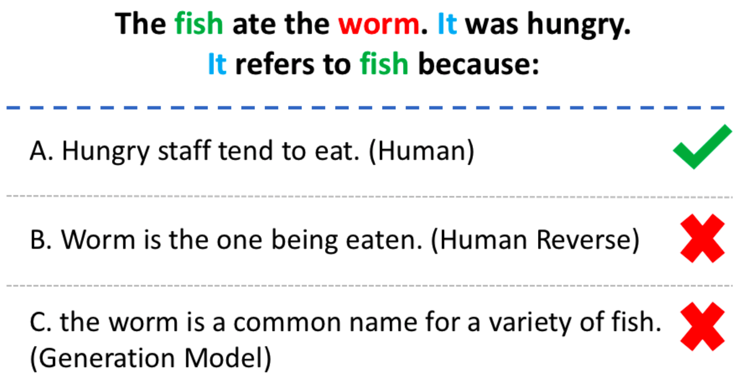

The fish ate the worm. It was hungry.
It refers to fish because:
A. Hungry staff tend to eat. (Human)
B. Worm is the one being eaten. (Human Reverse)
C. the worm is a common name for a variety of fish. (Generation Model)

Figure 1.3: An example from the WinoWhy dataset. Plausible and implausible reasons are indicated with the tick and the crosses, respectively. Different candidate reasons are shown in brackets. "Human Reverse" means the human reason for the reverse question.

soning ability or captures the spurious associations in the dataset). To answer this question, we present an in-depth diagnosis of current progress on commonsense reasoning. Specifically, we use the Winograd Schema Challenge [72] as the base task and ask the models to select the plausible reason for making the prediction.

Among all developed commonsense reasoning tasks, the Winograd Schema Challenge (WSC) [72], which is a hard pronoun coreference resolution task, is one of the most influential ones. All questions in WSC are grouped into pairs such that paired questions have minor differences (mostly one-word difference) but reversed answers. For each question, we denote the other question in the same pair as its reverse question. A pair of the WSC task is shown in Figure 1.2. Based on the design guideline of WSC, all commonly used features (e.g., gender, plurality, and co-occurrence frequency) do not have any effect. Human beings can solve these questions because of their shared commonsense knowledge. For example, ordinary people can know that the pronoun "it" in the first sentence refers to "fish" while the one in the second sentence refers to "worm" because "hungry" is a common property of something eating things while "tasty" is a common property of something being eaten.

Model	Property (337)	Object (856)	Eventuality (928)	Spatial (674)	Quantity (206)	Others (496)	Overall (2865)
Majority Voting	54.30%	56.31%	56.47%	52.67%	52.43%	55.24%	55.67%
BERT (base)	56.97%	56.54%	56.25%	54.01%	51.94%	55.44%	55.92%
BERT (large)	56.38%	57.24%	56.14%	53.41%	51.94%	56.65%	56.13%
RoBERTa (base)	54.30%	56.31%	56.90%	52.67%	52.91%	55.44%	55.78%
RoBERTa (large)	54.30%	56.43%	56.47%	52.67%	52.43%	55.04%	55.67%
GPT-2 (small)	56.68%	54.91%	57.11%	54.45%	**59.71%**	57.66%	56.37%
GPT-2 (large)	**57.57%**	54.44%	54.42%	55.93%	54.85%	54.84%	55.77%
BERT (base) + WSCR	55.49%	56.31%	56.90%	52.97%	51.94%	55.04%	55.71%
BERT (large) + WSCR	56.97%	56.31%	56.79%	53.12%	52.91%	55.04%	55.99%
BERT (base) + Grande	57.27%	56.43%	57.22%	53.41%	52.91%	55.24%	55.99%
BERT (large) + Grande	54.90%	56.07%	56.57%	52.67%	52.91%	55.44%	55.71%
RoBERTa (base) + WSCR	52.82%	55.61%	58.41%	53.26%	56.31%	55.04%	56.19%
RoBERTa (large) + WSCR	54.90%	58.06%	56.90%	52.08%	52.91%	56.85%	56.23%
RoBERTa (base) + Grande	56.08%	**58.88%**	58.19%	55.64%	57.28%	57.66%	58.05%
RoBERTa (large) + Grande	56.08%	58.06%	**59.59%**	**56.82%**	56.80%	**58.06%**	**58.18%**

Table 1.1: Performances of different models on WinoWhy questions. We report performances of different reason sets based on the required knowledge types. Reasons could belong to multiple categories as the original WSC questions could contain more than one knowledge type. The numbers of questions are shown in brackets.

To figure out whether language model-driven methods can truly understand the commonsense for solving the WSC questions, we invite annotators to provide reasons for why they choose the answers when they answer the questions and then create a new task "WinoWhy" with the collected reasons. For each question in the WSC task, we pair it with several reasons. Models are required to distinguish the correct reasons from very similar but wrong candidates. From examples in Figure 1.3, we can see that even though all candidates are highly related to the original question, only one of them is the correct reason for resolving the coreference relation.

We report the performance of best-performing WSC models on WinoWhy in Table 1.1.[4] Reasons are categorized into different types by humans for a better understanding. From the results, we can observe that even though pre-trained language representation models can achieve about 90% accuracy on the original WSC task, they still cannot understand the correct commonsense reasoning for making the decisions. Moreover, experimental results on different knowledge types prove that such a conclusion is universal rather than for a specific kind of knowledge. A possible reason

[4]More details about the dataset construction and experiments can be found at [186].

Methodology	Major limitations
Triplets	(1) Limited coverage of knowledge types; (2) Limited scale; (3) Cannot effectively represent preference
Language Models	(1) Not explainable; (2) Can only cover the commonsense in text.

Table 1.2: Major limitations of existing commonsense reasoning methodologies

is that even though the designers of WSC are trying to avoid any statistical correlation between the answer and the trigger word, such statistical correlation still exists. As a result, pre-trained language representation models can learn such correlation from large-scale training corpus and thus can answer WSC questions without fully understanding the reasons behind them. Besides that, we can also find that fine-tuning over a similar dataset (e.g., WSCR [126] and WinoGrande [136]) can slightly help RoBERTa, but the effect is still quite limited. This is probably because such a fine-tuning procedure only teaches pre-trained models to better answer WSC questions rather than understand the commonsense knowledge behind them.

Besides the explainability issue, the coverage of language models is another critical limitation. As discussed in [84], commonsense is often rarely explicitly expressed in human language because it is too trivial. And thus, it is infeasible to acquire all the commonsense from text like what language models have been doing.

Table 1.2 summarizes the major limitations of existing commonsense reasoning methodologies, which motivates us to think about how to develop a more reliable commonsense reasoning system. A reasonable way to do so is to achieve a balance between these two options. Unlike language models, we still need to represent commonsense in an explicit way such that we can preserve the explainability. Moreover, we can use such explicit knowledge to act as the bridge between different modalities such that we can go beyond text and collect richer knowledge from other modalities such as visual signals. At the same time, we also need to choose a new representation format rather than triplets such that it is more flexible for representing richer commonsense relations and can effectively preserve the preference property of commonsense. The methodology we propose in this thesis is representing commonsense with the higher-order selectional preference over eventualities, and we introduce more details in Section 1.2.

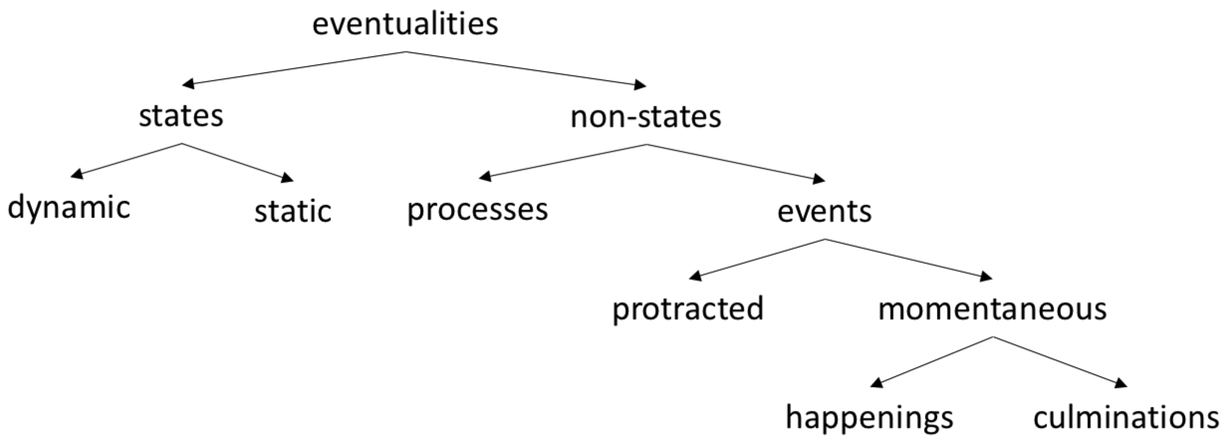


Figure 1.4: Eventuality Definitions inherited from [3].

1.2 Towards Robust Commonsense Reasoning with Higher-order Selectional Preference

As discussed in [84], commonsense knowledge refers to "millions of basic facts and understanding possessed by most people." Unlike factual knowledge like "London is the capital of UK," which is always true, commonsense knowledge is often not inevitably true and only reflects a kind of contextual preference. For example, in most cases, rocks are not used for eating, but some birds do eat rocks to help to digest. Such kind of knowledge is also called factoids [43, 42]. To effectively represent such preference-like commonsense knowledge, selectional preference [130] was proposed, which is traditionally defined on top of single dependency connections (e.g., nsubj, dobj, and amod). Given a word and a dependency relation, humans have preferences for which words are likely to be connected. For instance, when seeing the verb "sing," it is highly plausible that its object is "a song." When seeing the noun "air," it is highly plausible that its modifier is "fresh." However, such one-hop selectional preference is not enough to cover all the commonsense knowledge. Take Figure 1.3 as an example. To resolve the target pronoun "it" to "fish" rather than "worm," we need to know that the subject of the verb "eat" is more likely to be "hungry."

To effectively represent such commonsense, in [182], we propose to extend the concept of selectional preference to second-order such as "*Hungry*-amod-[Subject]-nsubj-*eat*." Experiments show that such second-order selectional preference is essential for solving WSC questions.

But still, as the inference unit in both the first-order and second-order selectional preference are words, they can only represent commonsense inside an eventuality (e.g., which eventuality is more likely to happen) and cannot express commonsense between eventualities. By "eventuality," we follow the definition from [3] and show it in Figure 1.4, from which we can see that "eventuality" is used as a general term for verb phrases. Examples of different classes are as follows:

1. **dynamic:** Sit, Stand, Lie + Location.
2. **static:** be drunk, be in New York.
3. **processes:** walk, push a cart.
4. **protracted:** build x, walk to Boston.
5. **happenings:** recognize, notice, flash once
6. **culminations:** die, reach the top

An example of the commonsense involving multiple eventualities is discussed by [168] and similar examples are frequently observed the Winogrande Schema Challenge [72]:

- The `soldiers` fired at the `women`, and we saw several of `them` fall.

To resolve the pronoun "`them`" in the above example, Wilks argued that machines need to access the commonsense knowledge or *partial information* (in MaCarthy's phrase) "*hurt things tending to fall down*," which can be translated into the following form: (hurt, `X`) $\xrightarrow{\text{ResultIn}}$ (`X`, fall), which is essentially the higher-order selectional preference that involves multiple events or states. For the clear representation, in the rest of this thesis, we refer to lower-order and higher-order selectional preference with the following definitions.

Definition 1.2.1 ***Lower-order Selectional Preference (SP)****: Preference about how likely an eventuality is going to happen.*

Definition 1.2.2 ***Higher-order Selectional Preference (SP)****: Preference about how likely a relation exists between two eventualities.*

In this thesis, to address the limitations of existing commonsense reasoning methodologies, we propose to represent commonsense with a weighted eventuality-centric knowledge graph, where

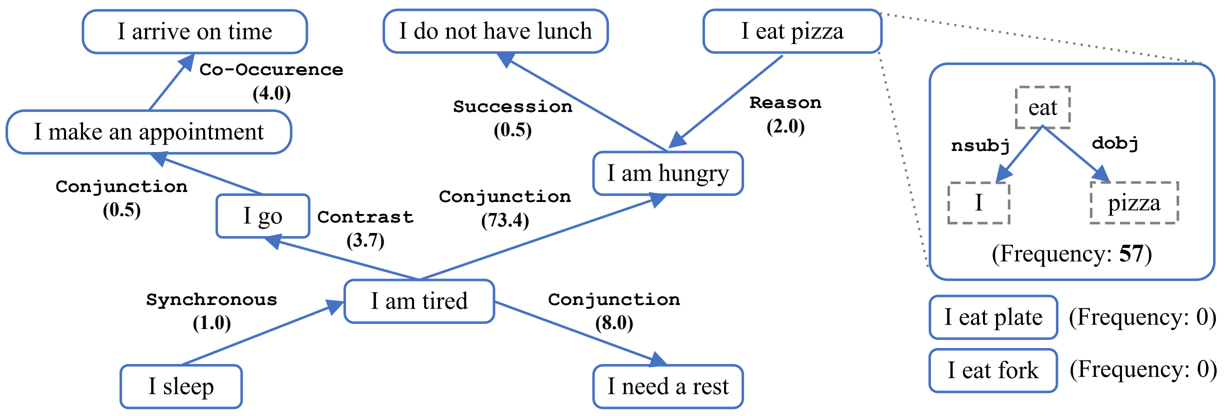


Figure 1.5: ASER Demonstration. Eventualities are connected with weighted directed edges. Each eventuality is a dependency graph.

the weight inside or across different eventualities can effectively represent humans' lower-order or higher-order selectional preference over eventualities. Even though different eventuality sub-classes are slightly different, as our goal is modeling humans' preference about the world rather than classifying the type of preference, we treat all eventualities as the same and do not carefully classify them in this thesis. We denote such a knowledge graph as Activity, State, Event, and their Relations (ASER) and demonstrate it in Figure 1.5. As shown in the example, the relations between eventualities are discourse relations (e.g., "Result") in discourse analysis, and both nodes and edges are associated with frequency-based weights to reflect higher-order selectional preferences given a specific linguistic (either dependency and discourse) pattern. As discussed by [130, 182], such frequency distribution can serve as a good fit of humans' selectional preference, which is indeed the commonsense knowledge. By observing that "I eat plate" and "I eat fork" never appear in ASER while "I eat pizza" appears 57 times. We can infer that "plate" and "fork" are not subjects that can be eaten while "pizza" is. Similarly, the frequencies of edges can be used to reflect the higher-order selectional preference between eventualities. For example, by observing that ("**PERSON** be hungry"-Result-"**PERSON** eat") appear at least 12 times while ("**PERSON** be hungry"-Reason-"**PERSON** eat") appears only once, we can know that "**PERSON** be hungry" is more likely to result in rather than be caused by "**PERSON** eat." We argue that the higher-order selectional preference in ASER can be scalable and effective to represent previously defined commonsense knowledge types in ConceptNet [84] and potentially many other types of commonsense knowledge.

To build such a large-scale eventuality knowledge graph, we first leverage existing tools (e.g., dependency/discourse parsing) and information extraction algorithms to extract eventualities as well as their relations from raw documents. For the eventuality extraction, considering that the English language's syntax is relatively fixed and consistent across domains and topics, instead of defining complex triggers and role structures of events, we use carefully designed syntactic patterns to extract all possible eventualities. We do not distinguish between semantic senses or categories of particular triggers or arguments in eventualities but treat all extracted words with their dependency relations as hyperedge in a graph to define an eventuality as a primitive semantic unit in our knowledge graph. For eventuality relations extraction, we adopt an end-to-end discourse parser [166] to automatically determine the discourse relations between eventuality spans and then create edges based on the predicted relation. Compared with previous commonsense knowledge acquisition methods, acquiring selectional preference knowledge with linguistic patterns and discourse relation prediction models is much cheaper and scalable and thus can be used to extract large-scale selectional preference knowledge from the unlabeled corpus. After that, to overcome the challenge that a large portion of the commonsense knowledge is rarely expressed in textual corpus and motivated by the observation [177] that human beings often conceptualize the events to a more abstract level such that they can be applied to new events, we propose to leverage existing conceptualization techniques [151, 152] to automatically generalize the knowledge we observed and extracted to those unseen eventualities.

In total, ASER contains **438,648,952** unique eventualities and **648,514,465** edges. Table 1.3 provides a size comparison between ASER[5] and existing eventuality-related (or simply verb-centric) knowledge bases. Essentially, they are not large enough as modern knowledge graphs and inadequate for capturing the richness and complexity of eventualities and their relations. FrameNet [5] is considered the earliest knowledge base defining events and their relations. It provides annotations about relations among about 1,000 human-defined eventuality frames, which contain 27,691 eventualities. However, given the fine-grained definition of frames, the scale of the annotations is limited. ACE [102] (and its follow-up evaluation TAC-KBP [1]) reduces the number of event types and annotates more examples in each of the event types. PropBank [107] and NomBank [92] build frames over syntactic parse trees, and focus on annotating popular verbs and nouns. TimeBank focuses only on temporal relations between verbs [119]. While the aforementioned knowledge bases are annotated by domain experts, ConceptNet[6] [84], Event2Mind [146], ProPora [21], and

[5]ASER (core) includes all eventualities that appear more than once, ASER (full) includes all extracted eventualities, and ASER (concept) includes all eventualities generated with the conceptualization

[6]Following the original definition, we only select the four relations ("HasPrerequisite", "HasFirstSubevent", "Has-

Table 1.3: Size comparison of ASER and existing eventuality-related resources. # Eventuality, # Relation, and # R types are the number of eventualities, relations between these eventualities, and relation types. For KGs containing knowledge about both entity and eventualities, we report the statistics about the eventualities subset. ASER (core) filters out eventualities that appear only once and thus has better accuracy while ASER (full) can cover more knowledge.

	# Eventuality	# Relation	# R Types
FrameNet	27,691	1,709	7
ACE	3,290	0	0
PropBank	112,917	0	0
NomBank	114,576	0	0
TimeBank	7,571	8,242	1
ConceptNet	74,989	116,097	4
Event2Mind	24,716	57,097	3
ProPora	2,406	16,269	1
ATOMIC	309,515	877,108	9
ATOMIC-2020	638,128	1,331,113	23
GLUECOSE	286,753	304,099	10
Knowlywood	964,758	2,644,415	4
ASER (core)	52,940,258	52,296,498	14
ASER (full)	438,648,952	648,514,465	14
ASER (concept)	15,640,017	224,213,142	14

ATOMIC [139], ATOMIC-2020 [55], and GLUECOSE [98] leveraged crowdsourcing platforms or the general public to annotate commonsense knowledge about eventualities, in particular the relations among them. Furthermore, KnowlyWood [160] uses semantic parsing to extract activities (verb+object) from movie/TV scenes and novels to build four types of relations (parent, previous, next, similarity) between activities using inference rules. Compared with all these eventuality-related KGs, **ASER** is larger by one or more orders of magnitude in terms of the numbers of eventualities and relations it contains.

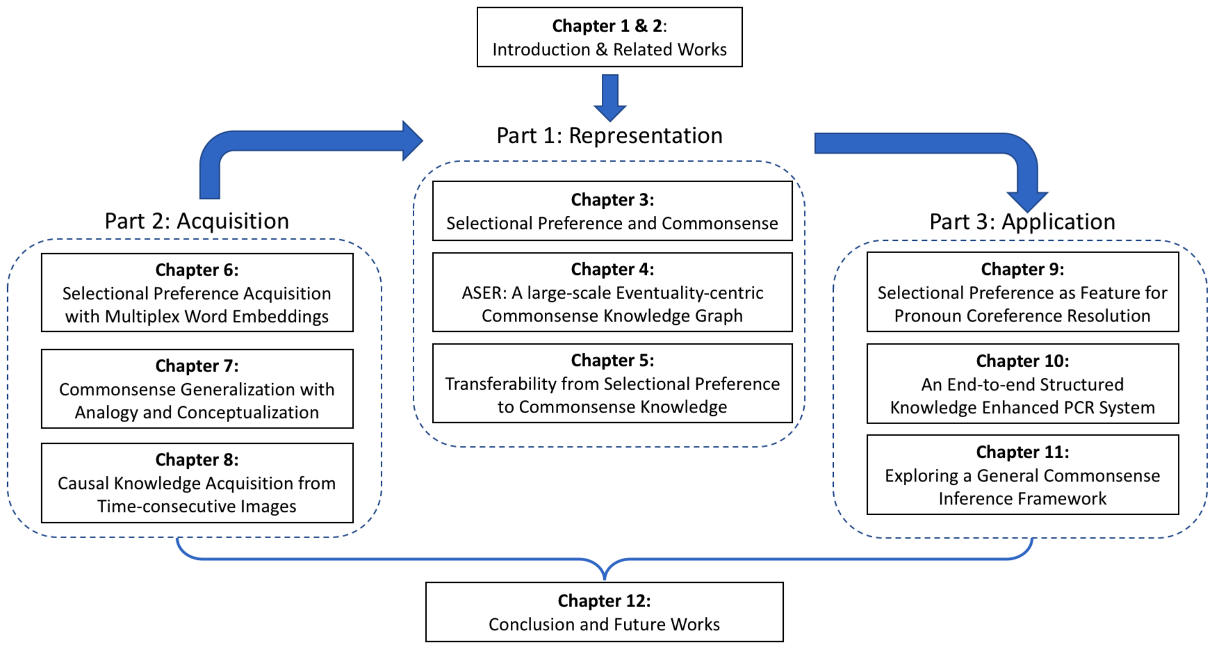
To demonstrate that the selectional preference knowledge in ASER, which is essentially the frequency information over linguistic relations, is indeed the commonsense we are interested in, we conduct experiments to prove the transferability from ASER to other human-defined commonsense knowledge graphs such as ConceptNet [84], which is also known as Open Mind CommonSense (OMCS). With a Pattern extraction-based approach, we can effectively convert ASER into TransOMCS, which is in the format of OMCS but two orders of magnitude larger. Compared with the data-driven method COMET [11] and the probing-based method LAMA [115], TransOMCS demonstrates the superiority in terms of quantity, quality, and novelty. Such success demonstrates

SubEvent", and "HasLastSubEvent") that involve eventualities.

the transferability from linguistic-based SP knowledge to commonsense knowledge and helps show that the higher-order selectional preference over eventualities is a sound commonsense representation methodology.

As aforementioned, an advantage of representing commonsense knowledge in such a structured way is that it can serve as the bridge to connect different modalities. Thus, besides proposing how should we represent the commonsense, we also explored different commonsense knowledge acquisition methods such as encoding SP knowledge with the multiplex word embeddings and then predicting unseen SP knowledge, how to leverage the analogy between eventualities and the eventuality conceptualization to transfer the knowledge from observed eventualities to unseen ones, and how to effectively acquire eventuality-centric commonsense from the visual signal.

After acquiring the knowledge, the next question to answer is how to help machines better conduct commonsense reasoning and understand human language with the knowledge. To answer this question, we first try one of the most traditional methods that converting the structured knowledge into features and then training a supervised model with the features. Experiments on pronoun coreference resolution demonstrate the effectiveness of such an approach. Later on, to address the issue that only a limited kind of knowledge can be easily converted into features, we propose an end-to-end network to directly utilize selectional preference knowledge into the model and achieve even better performance on the pronoun coreference resolution task. At the end of this thesis, we also try to move one step further to explore a general commonsense inference solution for all downstream tasks. Specifically, we assume that even though there could exist enormous kinds of commonsense reasoning tasks, the underline commonsense inference rules are limited. To testify this, we propose to convert different commonsense reasoning tasks into a unified question answering format and build a model that can learn the underline inference rules. As the ASER knowledge is provided for answering the questions, we denote this new commonsense reasoning formulation as commonsense knowledge base question answering (CKBQA). Experiments on CKBQA show that even though inference over commonsense knowledge is challenging, models can learn to conduct simple inference after training with a few examples. Another observation is that the learned model demonstrates the generalization ability across tasks, which was not observed in previous commonsense reasoning models.


Chapter 1 & 2:
Introduction & Related Works
Part 1: Representation
Chapter 3:
Selectional Preference and Commonsense
Chapter 4:
ASER: A large-scale Eventuality-centric Commonsense Knowledge Graph
Chapter 5:
Transferability from Selectional Preference to Commonsense Knowledge
Part 2: Acquisition
Chapter 6:
Selectional Preference Acquisition with Multiplex Word Embeddings
Chapter 7:
Commonsense Generalization with Analogy and Conceptualization
Chapter 8:
Causal Knowledge Acquisition from Time-consecutive Images
Part 3: Application
Chapter 9:
Selectional Preference as Feature for Pronoun Coreference Resolution
Chapter 10:
An End-to-end Structured Knowledge Enhanced PCR System
Chapter 11:
Exploring a General Commonsense Inference Framework
Chapter 12:
Conclusion and Future Works
Figure 1.6: Thesis Organization.

1.3 Thesis Organisation

As shown in Figure 1.6, this thesis is organized as follows. In Chapter 2, we introduce related works about commonsense reasoning. After that, in Part 1, we discuss the proposed commonsense representation methodology, which includes the higher-order selectional preference (Chapter 3), the construction of a large scale SP-based commonsense knowledge graph ASER (Chapter 4), and the transferability from SP knowledge in ASER to other human-defined commonsense relations (Chapter 5). Knowledge acquisition works are introduced in Part 2. Specifically, In Chapter 6, we introduce how to acquire selectional preference knowledge with a multiplex embedding approach. In Chapter 7, we introduce how to generalize our knowledge about familiar eventualities to unfamiliar ones with the help of analogy and conceptualization. In Chapter 8, we introduce the exploration we tried about acquiring higher-order selectional preference knowledge from vision signals. As the last piece of this thesis, we discuss how to apply the SP knowledge for downstream tasks in Part 3. We first use pronoun coreference resolution as the downstream task and propose two application methods: using SP knowledge as the feature, an end-to-end system that can automatically select and conduct inference over the SP knowledge, which are in Chapter 9 and 10, respectively. In Chapter 11, we introduce the proposed general commonsense inference framework. In the end, Chapter 12 includes the conclusion and discussion about remaining challenges about commonsense reasoning that we need to overcome in the future.

CHAPTER 2

RELATED WORKS

Commonsense knowledge typically refers to the knowledge shared by most people. As discussed by [45], such knowledge is often omitted in human language for communication efficiency, which brings a huge challenge for automatic natural language understanding systems. To overcome such a challenge and better understand human language, many efforts have been devoted during the past few decades in both the artificial intelligence and natural language processing communities. In this chapter, we introduce related works about commonsense reasoning from four perspectives: (1) Commonsense Representation and Resources; (2) Commonsense Knowledge Acquisition; (3) Commonsense Reasoning Evaluation; (4) Commonsense Reasoning Models.

2.1 Commonsense Representation and Resources

As an important knowledge resource for many artificial intelligence systems, commonsense knowledge covers various knowledge categories like causality [139], reasoning [141], property [84], and quantity [33], and has been proven crucial in many downstream tasks like question answering [79], dialogue system [189], reading comprehension [167], and pronoun coreference resolution [72].

To help machines understand the complex commonsense, one of the first steps is to represent the commonsense knowledge in a machine-readable format, acquire the knowledge, and construct a corresponding knowledge base. The design, construction, and curation of commonsense resources have long been a major pursuit of the AI community since its early days [90]. During the past few decades, the AI community has constructed several high-quality commonsense relevant knowledge resources, including but not limited to:

- **Cyc** [71]: As one of the earliest efforts in representing commonsense, Cyc is a long-term artificial intelligence project that aims to assemble a comprehensive ontology and knowledge base that can represent how this world works. By the time of 2017, the ontology of Cyc has grown to over 1.5 million terms.

- **ConceptNet** [84]: ConceptNet is one of the most known commonsense knowledge base in the world. It was initiated from Open Mind CommonSense Project, where careful crowd-sourcing was conducted to create a higher quality commonsense knowledge base that contains over 600 thousand triplets about 34 commonsense relations. Later on, after merging with other resources such as dictionaries, ConceptNet has grown to be ConceptNet 5.5 [154], which contains over 21 million edges about 8 million nodes. Even so, the core part of ConceptNet, which is about commonsense, is still not much bigger than the original one.

- **ATOMIC** [139]: Different from Concept, which has a broad coverage of commonsense relations, ATOMIC focuses on a niche scenario, the effect and causal relations between daily events. Specifically, it leverages crowd-sourcing to collect over 300 thousand triples about daily events that cover nine commonsense relations.

- **WordNet** [94]: As one of the most influential resources in the natural language processing community, WordNet provides the basic ontology for words and concepts. All concepts (i.e., synsets) are organized in a hierarchical structure. Even though such knowledge is not directly the "commonsense" we want, our experiments in Chapter 7 demonstrate that such knowledge can help us generalize the acquired commonsense knowledge about events.

- **FrameNet** [1]: Another important commonsense resource is the FrameNet. Instead of modeling the world with words, FrameNet proposes to model the world with the concept of frame. By merging similar events together, FrameNet collects a set of frames and their relations, which serve as one of the first event ontologies.

- **Webchild** [161]: a large collection of commonsense knowledge. Different from other resources, Webchild is automatically extracted and disambiguated from the web content. It contains triplets that connecting nouns and their properties such as "hasShape," "hasTaste," and "evokesEmotion." The most recent version of webchild contains about 7.5 million edges.

- **Visual Genome** [67]: Different from the aforementioned knowledge resources that only represent the commonsense knowledge with text, Visual Genome takes the images into consideration. It contains 108,077 images and over 2.3 million relations, which include many commonsense relations, in the images.

A common property of these knowledge resources is that they all require a pre-defined commonsense relation set and then leverage either crowd-sourcing or information extraction techniques to acquire the knowledge. However, it is often hard for a pre-defined commonsense relation set to cover all the commonsense in the world.

To address this issue, people start to think about how we can get rid of the pre-defined relations, and encoding all the knowledge with pre-trained language models becomes a popular option. Recently, with the fast development of pre-trained language models (e.g., GPT-2 [121] and GPT-3 [12]) and masked language models (e.g., BERT [25], RoBERTa [85], and Albert [68]), people start to realize that rich knowledge are encoded by the language models, and the commonsense knowledge is not an exception [116]. It has been well known that both the language models and masked language models are trained with unlabeled corpora, and the foundation of their capability of understanding language is the co-occurrence of words. As introduced in Chapter 1, the co-occurrence information or frequency can also reflect rich commonsense (e.g., it is more likely for "bird" rather than "fish" to appear before the verb "fly"). This observation explains why language models can encode rich commonsense knowledge. At the same time, it also reminds us of a severe limitation of the language models that their understanding is based on tokens rather than eventualities. As a result, they can effectively encode lower-order selectional preference, whose inference unit is words, while struggle at encoding the higher-order selectional preference whose inference unit are eventualities [176].

In this thesis, we propose to view the commonsense knowledge as selectional preference and use an eventuality-centric knowledge graph to represent the commonsense. Different from previous triplet formats, we design the knowledge graph to be weighted, not just for edges but also for nodes. As a result, we can use the weights of edge and nodes to reflect higher-order and lower-order selectional preference, respectively. Compared with the language models, representing the commonsense with such a structured format has the following advantages: (1) More explainable: we can be more clear about what is going on and how can we improve; (2) Better Coverage: we do not only have the lower-order preference knowledge but also have the higher one; (3) Multi-Modal: we can use such a format to connect different modalities such that we have the potential to go beyond text and acquire commonsense knowledge from different modalities such as the visual signal.

2.2 Commonsense Knowledge Acquisition

After introducing previous works on commonsense representation, how to efficiently and accurately acquire the commonsense is also crucial for developing a commonsense reasoning system. The most popular approach is crowd-sourcing, which has been adopted by many aforementioned knowledge bases such as ConceptNet [84] and ATOMIC [139]. An important advantage of crowd-sourcing is that the acquired knowledge is often of high quality because they are curated by humans, but at the same time, its expensive cost also limits its coverage.

Recently, to address the scale problem of crowd-sourcing, several attempts [75, 23] have been made to enrich the existing commonsense knowledge from a machine learning angle. Specifically, they train a relation prediction model with the acquired knowledge and then apply the model to predict new relations between concepts. However, these approaches cannot generate new nodes (concepts). To address this problem, several models were proposed to directly generate commonsense tuples in either supervised [11] or unsupervised [115] fashions.

Besides acquiring structured knowledge, another important acquisition method is language modeling. Compared with crowd-sourcing, language modeling is more scalable. But at the same time, its training loss determines that it is not easy to acquire complex commonsense such as the higher-order selectional preference over multiple eventualities. Besides that, another limitation of the language models is that they can only acquire commonsense knowledge from text and cannot effectively handle other modalities such as visual signals. As an important property of commonsense is that commonsense is rarely expressed explicitly in human language, the text might not be enough to cover all the commonsense.

Other than those approaches that try to acquire all kinds of commonsense at the same time, there are also many attempts that focus on a specific kind of commonsense. The Causal relation is one of the most popular ones. As a crucial knowledge for many artificial intelligence (AI) systems [111], causality has long been an important research topic in many communities with different focuses. For example, in the traditional AI community, researchers [41, 46, 7, 111, 143, 110, 29] are focusing on modeling causality from structured and formally defined data (e.g., directed acyclic graph). Different from them, researchers from the commonsense community are focusing on identifying key objects or events in images that can cause certain decision makings [39, 32], acquiring causal knowledge via either crowd-sourcing [84, 138], or linguistic pattern mining [51] and then applying the acquired knowledge for downstream tasks [101]. One thing worth mentioning is that there is a clear difference between formal causality studies and recent studies in the commonsense community. Compared with traditional causality studies, the causality in commonsense is typical "causal" and "contributory." However, they still play a critical role for many downstream (e.g., CV and NLP) tasks.

Besides the causal knowledge, another important type of commonsense is the temporal commonsense (e.g., which event is more likely to happen and the typical duration and frequency of events) [101]. One of the most influential works in this direction is [188], through a patter mining plus language model fine-tuning pipeline, it effectively encodes temporal knowledge such as the typical duration and frequency knowledge into the language models. As a result, it can predict that moving to a different city will take more time than moving a chair.

In Part 2, following the methodology of representing commonsense with selectional preference, we introduce three commonsense acquisition approaches that try to acquire commonsense knowledge from text, other knowledge graphs, and visual signal, respectively.

2.3 Commonsense Reasoning Tasks

Besides the commonsense representation and acquisition, an accurate and efficient evaluation system is also crucial for us to develop a reliable commonsense reasoning system. Recently, the community has developed many commonsense reasoning datasets to test models' commonsense reasoning abilities. Several popular ones are as follows:

- **Hard Pronoun Coreference Resolution**: The hard pronoun coreference resolution (Hard-PCR) task is one of the most famous commonsense reasoning tasks. For each question, a target pronoun and two candidate mentions are provided, and the task is to select the correct mention that the pronoun refers to. Careful expert annotations were conducted to get rid of the influence of all simple linguistic rules and ask the models to solve the problem with commonsense reasoning. In CKBQA, we include instances from WSC [72], DPR [126], and WinoGrande [136].

- **CommonnsenseQA**: CommonsenseQA [159] is a commonsense question answering dataset. For each question-answer pair, four relevant but wrong concepts are used as the other candidates, and the models are required to select the correct one out of five candidates. In CKBQA, we randomly sample a negative answer to make it a binary choice task, which is consistent with other datasets.

- **COPA**: COPA [133] focuses on evaluating whether models can understand the causality between events or not. For each head event, two candidate tail events are provided, and models are asked to predict the one caused by or the reason for the head event.

- **Timebank**: Timebank [119] is a dataset that focuses on evaluating whether models can correctly predict the temporal relations between events or not.

- **NumerSense**: Recently, language models have demonstrated a strong commonsense reasoning ability. To evaluate whether current models can understand the numerical common sense (e.g., how many legs does a bird has) or not, NumerSense [80] was proposed. It formulates the problem as a probing task.

Even though these datasets may have different formats (e.g., Winogrande [136] for slot fitting, and CommonsenseQA [159] for question answering), knowledge types (e.g., COPA [133] for causal commonsense, and NumerSense [80] for numerical commonsense), or modalities (e.g, VCR [179] for visual commonsense and many others for textual commonsense), they all follow a standard supervised learning setting, and aim at helping machines to solve a specific commonsense task in an end-to-end manner. Given this setting, it is often difficult to tell what has been learned during the training process. Was it used to acquire commonsense knowledge, learn to conduct commonsense inference, or both? Such ambiguity limits our progress in solving these commonsense reasoning tasks. To separate the effect of commonsense knowledge and inference, in Chapter 11, we convert multiple commonsense reasoning tasks into a unified format and equip each question with the essential knowledge such that the models do not need to worry about the knowledge and can focus on the inference. Moreover, as we converted different tasks into a unified format, we can easily evaluate the cross-task performance of these commonsense inference models, which can potentially motivate a general commonsense inference model in the future.

2.4 Commonsense Reasoning Models

In the last section of this chapter, we introduce previous commonsense reasoning models. In the past few decades, commonsense reason has long been one of the most challenging commonsense understanding tasks. Take the popular Winograd Schema Challenge [72] as an example. Even though researchers have tried to tackle this problem from a different angle, such as leveraging either search engines [35], linguistic knowledge [182, 185], or language representation models [65], experimental results showed that these models still cannot fully solve the problem, but we are not clear about how to further improve them.

A recent breakthrough made by the community is [65]. By fine-tuning a pre-trained language model with large-scale and high-quality training data (winogrande [136]), it achieves over 90% accuracy. However, as analyzed by [186, 34], the improvement is still largely due to observing similar questions rather than truly understanding the commonsense.

Besides fine-tuning language models, jointly using the structured knowledge and language models is another trend of commonsense reasoning models. Several representative works are as follows:

- **Kagnet**: As one of the pioneering works that utilized structured knowledge for solving commonsense reasoning tasks, Kagnet [78] first used a graph convolution network to encode the knowledge graph and then applied an LSTM based hierarchical attention mechanism to encode the knowledge path that starts with the concepts in the question and end with concepts in the answer. On the other hand, KagNet encodes the question and answers with pre-trained language models. In the end, it concatenates all representations together for the final prediction.

- **Graph based reasoning (GBR)**: Instead of only encoding paths starting with the question concepts and end with answer concepts, the follow-up work GBR [89] proposed to conduct a depth-first algorithm over the knowledge graph to generate a sequence of paths as the supporting knowledge paths.

- **Multi-Head Knowledge Attention (MHKA)**: To further exploit the provided knowledge, MHKA [109] employed a transformer network to model the paths from the question concepts and answer concepts, then concatenated the knowledge and context representation for the final prediction.

Even though these models have achieved certain improvements, there is still a notable gap between their current performance and fully understanding the commonsense. How to conduct commonsense reasoning is still an unsolved problem that worth exploring in the future.

PART 1

COMMONSENSE KNOWLEDGE REPRESENTATION

In this part, we discuss how we should represent commonsense knowledge. In Chapter 3, we start with the background of the lower bound of the semantic theory and the traditional one-hop selectional preference and then discuss why we should extend it to higher-order ones such as two-hop selectional preference. To study the connection between those one-hop and two-hop selectional preference and human-defined commonsense knowledge, we create a large-scale selectional preference resource SP-10K. From the experiments, we find out that there is a decent correlation between SP-10K and Open Mind CommonSense (OMCS), which is a human-crafted commonsense knowledge graph. In Chapter 4, we further extend the concept of selectional preference to be higher-order and construct a large-scale commonsense knowledge graph ASER following this principle. Construction details and careful analysis are presented. To demonstrate the collected selectional preference knowledge is indeed commonsense, in Section 5, we try to convert ASER into the format of OMCS [84]. Experiments show that with a simple pattern mining approach, we can effectively convert ASER into TransOMCS, which is in the same format as OMCS but two-magnitude larger.

CHAPTER 3

SELECTIONAL PREFERENCE AND COMMONSENSE

3.1 From The Lower Bound of a Semantic Theory to Selectional Preference

As discussed by the lower bound of a semantic theory [60], understanding human language requires both knowledge about the language (i.e., grammar) and knowledge about the world. As a result, if we fix the grammar structure of the linguistic descriptions, their difference will be the semantics. An example is shown in Figure 3.1. There are three sentences that share the same grammar structure but describe different events, which may have totally different reasons, effects, or sub-events. Give that the previous context is "It is dangerous," humans normally will prefer the second sentence to appear in this context because the lion is a dangerous animal. And such preference can reflect the commonsense knowledge we are after. This example shows that when the grammar is controlled, the selection we made can effectively represent the commonsense knowledge we care about.

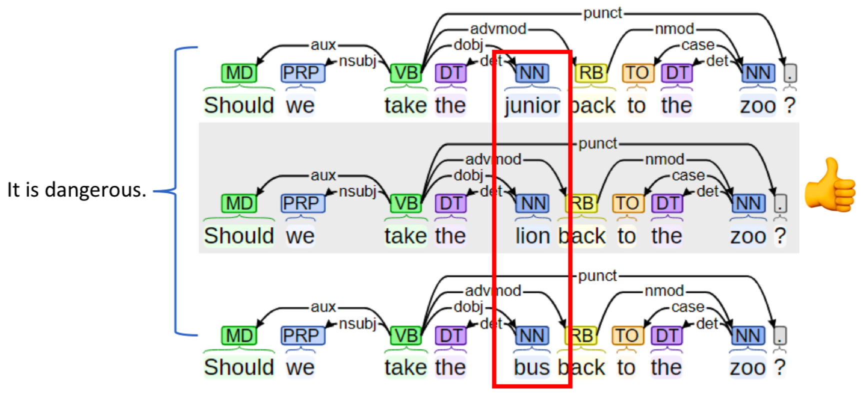

It is dangerous.
MD PRP VB DT NN RB TO DT NN .
Should we take the junior back to the zoo ?
Should we take the lion back to the zoo ?
Should we take the bus back to the zoo ?
aux
nsubj
dobj
det
advmod
punct
nmod
case
det

Figure 3.1: Principle Demonstration. When we fix the grammar, humans' preference over the linguistic description are reflecting the commonsense.

Historically, such grammar-based semantics is called selectional preference [131], which is a relaxation of selectional restrictions [60]. SP has been shown to be useful over a variety of tasks including sense disambiguation [130], semantic role classification [178], coreference clustering [53, 56, 50], and machine translation [162]. Originally, the research on selectional preference focuses on the IsA hierarchy in WordNet [38] and verb-object dependency relations. Later on, the idea of selectional preference was extended to verb-subject dependency relations. Several first-order selectional preference examples are as follows.

- SP(Cat, `IsA`, Animal) > SP(Cat, `IsA`, Plant)
- SP(Eat, `dobj`, Food) > SP(Eat, `dobj`, Rock)
- SP(Sing, `nsubj`, Singer) > SP(Sing, `nsubj`, House)

Aside from these relations, we believe that higher-order dependency relations may also reflect meaningful commonsense knowledge. Consider the following two examples of hard pronoun resolution problems from the Winograd Schema Challenge [72]:

- (A) The fish ate the worm. It was hungry.
- (B) The fish ate the worm. It was tasty.

In (A), we can resolve "it" to "the fish" because it is more plausible that the subject of the verb "eat" is hungry. On the other hand, for (B), we can resolve "it" to "the worm" because it is more likely that the object of the verb "eat" is tasty. We can formalize the commonsense that the subject of eat is more likely to be "hungry" rather than "tasty" with the following second-order selectional preference:

- SP(Eat, `Nsubj-amod`, Hungry) > SP(Eat, `Nsubj-amod`, Tasty)

To have a better understanding of the connection between selectional preference and the commonsense knowledge defined by humans, we first collect a high-quality selectional preference dataset SP-10K. To construction details are shown in Section 3.2. After collecting the SP-10K, we carefully analyze the relation between SP-10K and Open Mind CommonSense (OMCS) [145] in Section 3.3. At the end of this chapter, in Section 3.4, we conduct experiments to see whether the high-quality selectional preference knowledge in SP-10K can help us better solve the commonsense reasoning task Winograd Schema Challenge [72] or not.

SP Evaluation Set	#R	#W	#P
(McRae et al., 1998) [91]	2	641	821
(Keller and Lapata, 2003) [62]	3	571	540
(Pado et al., 2006) [106]	3	180	207
SP-10K	5	2.5K	10K

Table 3.1: Statistics of Human-labeled SP Evaluation Sets. #R, #W, and #P indicate the number of SP relation types, words, and pairs, respectively.

3.2 SP-10K

3.2.1 Design of SP-10K

As discussed in [52], a high-quality evaluation resource should be: (1) clearly defined; (2) representative; and (3) consistent and reliable.

First, similar to existing human-labeled SP evaluation sets [91, 62, 106], SP-10K uses the plausibility of selectional pairs as the annotation. Hence, SP-10K is clearly defined. Second, compared to these existing evaluation sets, as shown in Table 3.1, SP-10K covers a larger number of relations and SP pairs, making it a more representative evaluation set. Finally, as shown by the high inner-annotation agreement in Section 3.2.2, the annotation of SP-10K is consistent and reliable.

Selectional Relations

Traditionally, the study of SP has focused on three selectional relations: verb-subject, verb-object, and noun-adjective. As demonstrated in Section 3.1, some verbs have a preference for the properties of their subjects and objects. For example, it is plausible to say that the subject of "eat" is hungry and the object of "eat" is tasty, but not the other way round. To capture such preferences, we propose two novel two-hop dependency relations, "dobj_amod" and "nsubj_amod." Examples of these relations are presented in Table 3.2. In total, SP-10K contains five SP relations.

Following previous approaches [91, 106], for the "dobj" and "nsubj" relations, we take a verb as the head and a noun as the dependent. Similarly, for "dobj_amod" and "nsubj_amod" relations, we take a verb as the head and an adjective as the dependent. Moreover, for the "amod" relation, we take a noun as the head and an adjective as the dependent.

Relation	Frequent	Random
“dobj”	(ask, question) (ask, time)	(ask, voting) (ask, stability)
“nsubj”	(people, eat) (husband, eat)	(textbook, eat) (stream, eat)
“amod”	(fresh, air) (cold, air)	(rational, air) (original, air)
“dobj_amod”	(design, new) (design, original)	(design, official) (design, civil)
“nsubj_amod”	(friendly, smile) (symbolic, smile)	(young, smile) (civilian, smile)

Table 3.2: Examples of candidate pairs for annotation. For the ease of understanding, the order of head and dependent may be different for various relations.

Candidate SP Pairs

The selected vocabulary consists of 2,500 verbs, nouns, and adjectives from the 5,000 most frequent words[1] in the Corpus of Contemporary American English.

For each SP relation, we provide two types of SP pairs for our annotators to label: frequent pairs and random pairs. For each selectional relation, we first select the 500 most frequent heads. We then match each head with its two most frequently-paired dependents, as well as two randomly selected dependents from our vocabulary. As such, we retrieve 2,000 pairs for each relation. Altogether, we retrieve 10,000 pairs for five selectional relations. These pairs are composed of 500 verbs, 1,343 nouns, and 657 adjectives. Examples of sampled pairs are presented in Table 3.2.

3.2.2 Annotation of SP-10K

Survey Design

Following the SimLex-999 annotation guidelines [52], we invite at least 11 annotators to score each SP pair. We divide our 10,000 pairs into 100 surveys. Each survey contains 103 questions, three of which are checkpoint questions selected from the examples to control the labeling quality. Within a survey, all the questions are derived from the same selectional relation to improve the efficiency of survey completion.

[1]https://www.wordfrequency.info/free.asp.

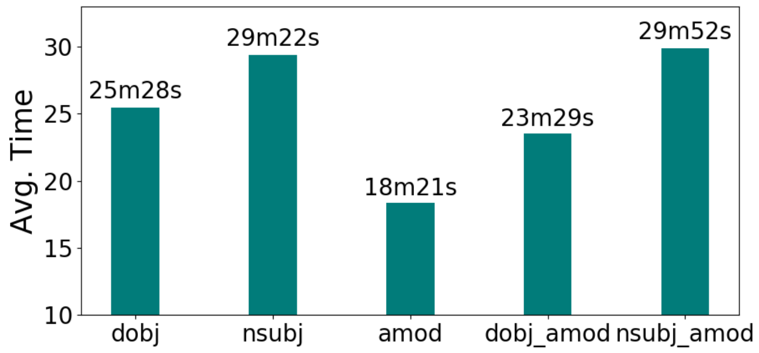


Figure 3.2: Average annotation time per 100 questions. “m” indicates minutes and “s” indicates seconds.

Each survey is consist of three parts. We begin by explaining the task to the annotators, including how to deal with the special case like multi-word expressions. Then, we present three examples to help the annotators better understand the task. Finally, we ask questions using the following templates (VERB, ADJ, and NOUN are place holders and will be replaced with the corresponding heads and dependents in the actual surveys.):

- **dobj:** How suitable do you think it is if we use NOUN as the object of the verb VERB?
- **nsubj:** How suitable do you think it is if we use NOUN as the subject of the verb VERB?
- **amod:** How suitable do you think it is if we use ADJ to describe the noun NOUN?
- **dobj_amod:** How suitable do you think it is if we use ADJ to describe the object of the verb VERB?
- **nsubj_amod:** How suitable do you think it is if we use ADJ to describe the subject of the verb VERB?

For each question, the annotator is asked to select one of the following options: Perfectly match (5), Make sense (4), Normal (3), Seems weird (2), It’s not applicable at all (1). We randomize the order of frequent and random pairs to prevent annotators from simply memorizing the question order.

Participants and Annotation

We employ the Amazon Mechanical Turk platform (MTurk) for our annotations,[2] and require that our annotators are "Master Workers," indicating reliable annotation records.[3] We also require our annotators to be either native English speakers or currently live and/or work in English-speaking locales. Based on these criteria, we identified 125 valid annotators. These annotators produced 130,575 ratings for a total cost of USD1,182.80. We support the multiple participation of annotators by ensuring that subsequent surveys are generated with their previously-unanswered questions.

From our annotation statistics, we notice that different selectional relations take different time to annotate. As shown in Figure 3.2, the annotators spent the least time on the "amod" relation, suggesting that the modifying relation is relatively easy to understand and judge. Another interesting finding is that the annotators spend more time on relations involving subjects than those involving objects, which is consistent with the observation proposed by [57] that verbs have clearer preferences for objects than subjects.

Post-processing

We excluded ratings from annotators who (1) provided incorrect answers to any of the checkpoint questions or (2) demonstrated suspicious annotation patterns (e.g., marking all pairs as "normal"). After excluding based on this criteria, we obtained 100,532 valid annotations with an overall acceptance rate of 77%. We calculate the plausibility for each SP pair by taking the average rating for the pair over all (at least 10) valid annotations, then linearly scaling this average from the 1-5 to 0-10 interval. This approach is similar to the post-processing in [52]. We present a sample of SP pairs in Table 3.3. Some of the pairs are interesting. For example, for the dobj_amod relation, annotators agree that lifting a heavy object is a usually used expression, while earning a rubber object is rare.

[2]According to [112], Amazon MTurk (https://www.mturk.com/) has the largest worker population and highest annotation quality compared to other crowdsourcing services.

[3]https://www.mturk.com/worker/help

(a) dobj

SP Pair	Plausibility
(eat, meal)	10.00
(close, door)	8.50
(convince, people)	7.75
(touch, food)	5.50
(hate, investment)	4.00
(confront, impulse)	2.78
(eat, mail)	0.00

(b) nsubj

SP Pair	Plausibility
(singer, sing)	10.00
(law, permit)	7.78
(women, pray)	5.83
(realm, remain)	3.06
(victim, contain)	2.22
(bar, act)	1.39
(textbook, eat)	0.00

(c) amod

SP Pair	Plausibility
(fresh, air)	9.77
(new, method)	8.89
(young, people)	6.82
(medium, number)	4.09
(immediate, food)	2.50
(eager, price)	1.36
(secret, wind)	0.75

(d) dobj_amod

SP Pair	Plausibility
(lift, heavy *object*)	9.17
(design, new *object*)	8.00
(recall, previous *object*)	7.05
(attack, small *object*)	5.23
(drag, drunk *object*)	4.25
(inform, weird *object*)	3.64
(earn, rubber *object*)	0.63

(e) nsubj_amod

SP Pair	Plausibility
(friendly *subject*, smile)	10.00
(evil *subject*, attack)	9.00
(recent *subject*, demonstrate)	6.00
(random *subject*, bear)	4.00
(happy *subject*, steal)	2.25
(stable *subject*, understand)	1.75
(sunny *subject*, make)	0.56

Table 3.3: Sampled SP pairs from SP-10K and their plausibility ratings. *object* and *subject* are place holders to help understand the two-hop SP relations.

Inner-Annotator Agreement

Following standard practices from previous datasets WSIM-203 [128] and Simlex-999 [52], we employ Inter-Annotator Agreement (IAA), which computes the average correlation of an annotator with the average of all the other annotators, to evaluate the overall annotation quality. As presented in Table 3.4, the overall IAA of SP-10K is ρ = 0.75, which is comparable to existing datasets WSIM-203 (0.65) and Simlex-999 (0.78).

	dobj	nsubj	amod	d_a	n_a	overall
IAA	0.83	0.77	0.81	0.71	0.63	0.75

Table 3.4: Overall Inter-Annotator Agreement (IAA) of SP-10K. “d_a” stands for dobj_amod and “n_a” stands for nsubj_amod.

Unsurprisingly, the IAA is not uniform across different SP relations. As shown in Table 3.4, complicated two-hop SP relations are more challenging and achieve relatively lower correlations than the simpler one-hop relations. This experimental result shows that two-hop relations are more difficult than one-hop SP relations. We also notice that the agreements among annotators for SP relations involving the subjects of verbs are relatively low. The above observations are consistent with our earlier discussion on annotation time, and further support the claim that verbs have stronger preferences for their objects than their subjects.

3.3 Correlation between SP-10K and OMCS

In this section, we quantitatively analyze the relationship between SP and commonsense knowledge. Currently, the largest commonsense knowledge dataset is the Open Mind Common Sense (OMCS) from the ConceptNet 5 [153] knowledge base. The OMCS contains 600k crowdsourced commonsense triplets such as (food, UsedFor, eat) and (wind, CapableOf, blow to east). All of the relations in OMCS are human-defined. In comparison, SP only relies on naturally occurring dependency relations, which can be accurately identified using existing parsing tools [142].

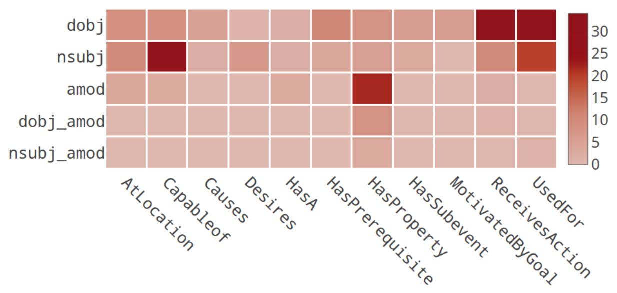
We aim to demonstrate how SP is related to commonsense knowledge. Building relationships between SP and human-defined relations has two advantages: (1) We may be able to directly acquire commonsense knowledge through SP acquisition techniques. (2) We may be able to solve commonsense reasoning tasks from the perspective of SP, as illustrated through the two Winograd examples in Section 3.1. These advantages motivate exploring the potential of using SP to represent commonsense knowledge.

3.3.1 SP Pairs and OMCS Triplets

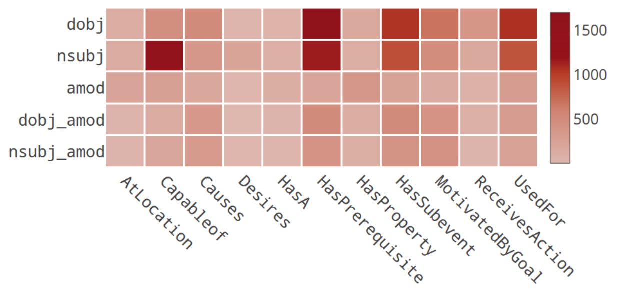
We hypothesize that the plausibility of an SP pair relates to how closely the pair aligns with human commonsense knowledge. As such, the more plausible pairs in SP-10K should be more likely to

Group	#Pairs	#Exact Match (Percentage)	#Partial Match (Percentage)
Perfect	755	85 (11.26%)	287 (38.01%)
Good	2,600	67 (2.58%)	885 (34.04%)
Normal	2,809	20 (0.71%)	504 (17.94%)
Unusual	2,396	6 (0.25%)	187 (7.80%)
Impossible	1,440	5 (0.35%)	82 (5.69%)

Table 3.5: Matching statistics of SP pairs by plausibility.



(a) Exact Match



(b) Partial Match

Figure 3.3: Matched SP relations and OMCS relations. Interesting relation matches such as "dobj" versus "UserFor," "nsubj" versus "CapableOf," and "amod" versus "HasProperty" are observed.

be covered by the OMCS dataset.

Using plausibility as our criterion, we split the 10,000 SP pairs into five groups: Perfect (8-10), Good (6-8), Normal (4-6), Unusual (2-4), and Impossible (0-2). As OMCS triplets contain phrases and SP pairs only contain words, we use two methods to match SP pairs with OMCS triplets. (1) Exact Match: we identify triplets in OMCS where the two dependents are exactly the same as the two words in an SP pair. (2) Partial Match: we identify triplets in OMCS where the two dependents contain the two words in an SP pair. We count SP pairs that fulfill either of these matching methods as covered by OMCS. Note that exact matches are not double-counted as partial matches.

As shown in Table 3.5, almost 50% of SP pairs in the perfect group are covered by OMCS. In contrast, only about 6% of SP pairs from the impossible group are covered. More plausible selectional preference pairs are more likely to be covered by OMCS, which supports our hypothesis of more plausible SP pairs being more closely aligned with human commonsense knowledge.

(a) Perfect group (Plausibility: 8-10)

SP relation	SP pair versus OMCS triplets
“dobj”	(sing, song) (9.25/10) (song, UsedFor, sing)
“nsubj”	(phone, ring) (8.75/10) (phone, CapableOf, ring)
“amod”	(cold, water) (8.86/10) (water, HasProperty, cold)
“dobj_amod”	(create, new) (8.25/10) (create idea, UsedFor, invent new things)
“nsubj_amod”	(hungry, eat) (10.00/10) (eat, MotivatedByGoal, are hungry)

(b) Impossible group (Plausibility: 0-2)

SP relation	SP pair versus OMCS triplets
“dobj”	(eat, mail) (0.00/10) (mail letter, HasSubevent, eat cheese)
“nsubj”	(library, love) (1.25/10) (love, Atlocation, library)
“amod”	(red, child) (0.68/10) (child wagon, HasProperty, red)
“dobj_amod”	(drive, bottom) (1.50/10) (drive car, HasSubevent, bottom out)
“nsubj_amod”	(fun, hurt) (1.50/10) (having fun, HasSubevent, get hurt)

Table 3.6: Examples of OMCS-covered SP pairs and their corresponding OMCS triplets.

3.3.2 SP and Human-defined Relations

To show the connection between SP relations and human-defined relations, we visualize all matching (SP pair, OMCS triplet) tuples in Figure 3.3. A darker color indicates a greater number of matched tuples, which in turn suggests a stronger connection between the two relations.

We observe some clear and reasonable matches such as (“dobj,” “UsedFor”), (“nsubj,” “CapableOf”), and (“amod,” “HasProperty”), which demonstrates that some simple human-defined relations like “UsedFor,” “CapableOf,” and “HasProperty” are related to corresponding SP relations. We also notice that the five SP relations in SP-10K seldom match some OMCS relations such as “HasA” and “HasSubevent,” which indicates a need for additional SP relations or even the combination of different SP relations. We leave it for our future work.

3.3.3 Case Study

We present a selection of covered pairs from the perfect and impossible groups in Table 3.6. For the perfect group, we find that human-defined commonsense triplets often have neatly corresponding SP pairs. On the other hand, for the impossible group, SP pairs are covered by OMCS either because of incidental overlap with a non-keyword, e.g., “child” in “child wagon,” or because of the low quality of some OMCS triplets. This further illustrates that OMCS still has room for improvement and that SP may provide an effective way to improve commonsense knowledge.

3.4 Effect of SP-10K knowledge on Winograd Schema Challenge

As introduced in Section 3.1, a novel contribution of this thesis is the two-hop Selectional Preference relations: “nsubj_amod” and “dobj_amod.” To demonstrate their effectiveness, we select a subset[4] of the Winograd Schema Challenge dataset [72], which leverages the two-hop selectional preference knowledge to solve. In total, we have 72 questions out of overall 285 questions. The selected Winograd question is defined as follows: Given one sentence s containing two candidates (n_1, n_2) and one pronoun p, which is described with one adjective a, we need to find which candidate is the pronoun referring to. One example is as follows:

- *Jim* yelled at *Kevin* because **he** was so upset.

We need to correctly finds out **he** refers to *Jim* rather than *Kevin*. These tasks are quite challenging as both the Stanford CoreNLP coreference system and the current state-of-the-art end-to-end coreference model [70] cannot solve them. To solve that problem from the perspective of selectional preference (SP), we first parse the sentence and get the dependency relations related to the two candidates. If they appear as the subject or the object of the verb h, we will then check the SP score of the head-dependent pair (h, d) on relations “nsubj_amod” and “dobj_amod” respectively.

[4]We select all examples that use one adjective to describe the targeting pronoun and all selected questions are listed in the appendix.

Model	Correct	Wrong	NA	A_p	A_o
Stanford	33	35	4	48.5%	48.6%
End2end	36	36	0	50.0%	50.0%
SP-10K	13	0	59	**100%**	59.0%

Table 3.7: Result of different models on the subset of Winograd Schema Challenge. NA means that the model cannot give a prediction, A_p means the accuracy of predict examples without NA examples, and A_o means the overall accuracy.

After that, we compare the SP score of two candidates and select the higher one as the prediction result. If they have the same SP score, we will make no prediction.

We show the result of collected human-labeled data in "SP-10K." From the result, we can see that "SP-10K" can solve that problem with very high precision. But as we only label 4,000 multi-hop pairs, the overall coverage is limited. The experimental result shows that if we can automatically build a good multi-hop SP model, we could make some steps towards solving the hard pronoun coreference task, which is viewed a vital task of natural language understanding.

CHAPTER 4

ASER: A LARGE-SCALE EVENTUALITY-CENTRIC COMMONSENSE KNOWLEDGE GRAPH

Chapter 3 has demonstrated lower-order selectional preference can effectively represent rich commonsense knowledge, but a lot of higher-order selectional preference knowledge over multiple eventualities is still not well studied. In this chapter, we introduce the proposed eventuality-centric knowledge graph ASER, which can serve as an efficient representation methodology for both the lower-order and higher-order selectional preference. Specifically, we first present a formal definition of an eventuality-centric commonsense knowledge graph in Section 4.1. After that, we introduce a linguistic rules-based extraction pipeline that can automatically extract ASER from unlabelled corpora in Section 4.2. Knowledge graph statistics, intrinsic evaluation, and extrinsic evaluations are presented in Section 4.3, 4.4, and 4.5, respectively.

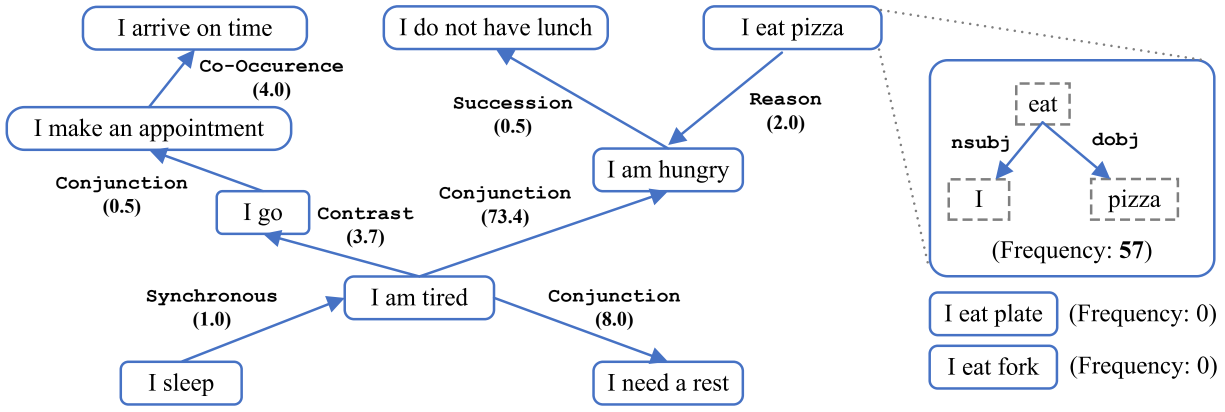


Figure 4.1: ASER Demonstration. Eventualities are connected with weighted directed edges. Each eventuality is a dependency graph.

4.1 Overview of ASER

As demonstrated in Figure 4.1, ASER is a hybrid graph combining a hypergraph $\{\mathcal{V}, \mathcal{E}\}$ where each hyperedge is constructed over vertices, and a traditional graph $\{\mathcal{E}, \mathcal{R}\}$ where each edge is built among eventualities. For example, E_h=`(I, am, hungry)` and E_t=`(I, eat, anything)` are eventualities, where we omit the internal dependency structures for brevity. They have a relation $\langle \mathsf{E}_h, \texttt{Result}, \mathsf{E}_t \rangle$, where `Result` is the relation type. We devise the formal definition of ASER as below.

Definition 4.1.1 **ASER KG** *is a hybrid graph $\mathcal{H}$ of eventualities E's. Each* **eventuality** E *is a hyperedge linking to a set of vertices ν's. Each vertex ν is a* **word** *in the vocabulary. We define $\nu \in \mathcal{V}$ in the vertex set and $\mathsf{E} \in \mathcal{E}$ in the hyperedge set. $\mathcal{E} \subseteq \mathcal{P}(\mathcal{V}) \setminus \{\emptyset\}$ is a subset of the power set of $\mathcal{V}$. We also define a* **relation** $\mathsf{R}_{i,j} \in \mathcal{R}$ *between two eventualities E_i and E_j, where $\mathcal{R}$ is the relation set. Each relation has a* **type** $\mathsf{T} \in \mathcal{T}$ *where $\mathcal{T}$ is the type set. Overall, we have ASER KG $\mathcal{H} = \{\mathcal{V}, \mathcal{E}, \mathcal{R}, \mathcal{T}\}$.*

With this formulation, we can use the frequency of nodes or edges in ASER to represent the lower-order or higher-order selectional preference, respectively.

4.1.1 Eventuality

Different from named entities or concepts, which are noun phrases, eventualities are usually expressed as verb phrases, which are more complicated in structure. Our definition of eventualities is built upon the following two assumptions: (1) syntactic patterns of English are relatively fixed and consistent; (2) the eventuality's semantic meaning is determined by the words it contains. To avoid the extracted eventualities being too sparse, we use words fitting certain patterns rather than a whole sentence to represent an eventuality. In addition, to make sure the extracted eventualities have complete semantics, we retain all necessary words extracted by patterns rather than those simple verbs or verb-object pairs in sentences. The selected patterns are shown in Table 4.1. For example, for the eventuality `(dog, bark)`, we have a relation `nsubj` between the two words to indicate that there is a subject-of-a-verb relation in between. We now formally define an eventuality as follows.

Definition 4.1.2 *An eventuality E is a hyperedge linking multiple words $\{\nu_1, \ldots, \nu_N\}$, where N is the number of words in eventuality E. Here, $\nu_1, \ldots, \nu_N \in \mathcal{V}$ are all in the vocabulary. A pair of words in E (ν_i, ν_j) may follow a syntactic relation $e_{i,j}$. The weight of E, denoted as $w_{\mathsf{E}}^{(e)}$, is defined by the frequencies of appearance in the whole corpora.*

Table 4.1: Selected eventuality patterns ("v" stands for normal verbs other than "be", "be" stands for "be" verbs, "n" stands for nouns, "a" stands for adjectives, and "p" stands for prepositions.), Code, and the corresponding examples.

Pattern	Code	Example
n_1-nsubj-v_1	s-v	"The dog barks"
n_1-nsubj-v_1-dobj-n_2	s-v-o	"I love you"
n_1-nsubj-v_1-xcomp-a	s-v-a	"He felt ill"
n_1-nsubj-v_1-xcomp-v_2	s-v-v	"I want to go"
n_1-nsubj-(v_1-iobj-n_2)-dobj-n_3	s-v-o-o	"You give me the book"
n_1-nsubj-v_1-xcomp-v_2-dobj-n_2	s-v-v-o	"I want to eat the apple"
n_1-nsubj-(v_1-dobj-n_2)-xcomp-v_2-dobj-n_3	s-v-o-v-o	"I ask you to help us"
n_1-nsubj-(v_1-dobj-n_2)-xcomp-(v_2-iobj-n_3)-dobj-n_4	s-v-o-v-o-o	"president urges the congress to make her citizen"
n_1-nsubj-a_1-cop-be	s-be-a	"The dog is cute"
n_1-nsubj-n_2-cop-be	s-be-o	"He is a boy"
n_1-nsubj-v_1-xcomp-n_2-cop-be	s-v-be-o	"I want to be a hero"
n_1-nsubj-v_1-xcomp-a_1-cop-be	s-v-be-a	"I want to be slim"
n_1-nsubj-(v_1-iobj-n_2)-xcomp-n_3-cop-be	s-v-o-be-o	"I want her to be hero"
n_1-nsubj-(v_1-iobj-n_2)-xcomp-a_1-cop-be	s-v-o-be-a	"I want her to be happy"
$there$-expl-be-nsubj-n_1	there-be-o	"There is an apple"
n_1-nsubjpass-v_1	spass-v	"The bill is paid"
n_1-nsubjpass-v_1-dobj-n_2	spass-v-o	"He is served water"
n_1-nsubjpass-v_1-xcomp-v_2-dobj-n_2	spass-v-v-o	"He is asked to help us"

We use patterns from dependency parsing to extract eventualities E's from unstructured large-scale corpora. Here $e_{i,j}$ is one of the relations that dependency parsing may return. Although in this way the recall is sacrificed, our patterns are of high precision and we use very large corpora to extract as many eventualities as possible. This strategy is also shared with many other modern KGs [37, 6, 15, 172].

4.1.2 Eventuality Relation

For relations among eventualities, we follow PDTB's [118] definition of relations between sentences or clauses but simplify them to eventualities. Following the CoNLL 2015 discourse parsing shared task [174], we select 14 discourse relation types and an additional co-occurrence relation to build our knowledge graph.

Definition 4.1.3 *A relation R between a pair of eventualities E_h and E_t has one of the following types $T \in \mathcal{T}$ and all types can be grouped into five categories:* **Temporal** *(including Precedence,*

Table 4.2: Eventuality relation types between two eventualities E_h and E_t and explanations.

Relation	Explanation
$\langle \mathsf{E}_h, \texttt{Precedence}, \mathsf{E}_t \rangle$	E_h happens before E_t.
$\langle \mathsf{E}_h, \texttt{Succession}, \mathsf{E}_t \rangle$	E_h happens after E_t.
$\langle \mathsf{E}_h, \texttt{Synchronous}, \mathsf{E}_t \rangle$	E_h happens at the same time as E_t.
$\langle \mathsf{E}_h, \texttt{Reason}, \mathsf{E}_t \rangle$	E_h happens because E_t happens.
$\langle \mathsf{E}_h, \texttt{Result}, \mathsf{E}_t \rangle$	If E_h happens, it will result in the happening of E_t.
$\langle \mathsf{E}_h, \texttt{Condition}, \mathsf{E}_t \rangle$	Only when E_t happens, E_h can happen.
$\langle \mathsf{E}_h, \texttt{Contrast}, \mathsf{E}_t \rangle$	E_h and E_t share a predicate or property and have significant difference on that property.
$\langle \mathsf{E}_h, \texttt{Concession}, \mathsf{E}_t \rangle$	E_h should result in the happening of E_3, but E_t indicates the opposite of E_3 happens.
$\langle \mathsf{E}_h, \texttt{Conjunction}, \mathsf{E}_t \rangle$	E_h and E_t both happen.
$\langle \mathsf{E}_h, \texttt{Instantiation}, \mathsf{E}_t \rangle$	E_t is a more detailed description of E_h.
$\langle \mathsf{E}_h, \texttt{Restatement}, \mathsf{E}_t \rangle$	E_t restates the semantics meaning of E_h.
$\langle \mathsf{E}_h, \texttt{Alternative}, \mathsf{E}_t \rangle$	E_h and E_t are alternative situations of each other.
$\langle \mathsf{E}_h, \texttt{ChosenAlternative}, \mathsf{E}_t \rangle$	E_h and E_t are alternative situations of each other, but the subject prefers E_h.
$\langle \mathsf{E}_h, \texttt{Exception}, \mathsf{E}_t \rangle$	E_t is an exception of E_h.
$\langle \mathsf{E}_h, \texttt{Co-Occurrence}, \mathsf{E}_t \rangle$	E_h and E_t appear in the same sentence.

Succession, and Synchronous), **Contingency** *(including Reason, Result, and Condition),* **Comparison** *(including Contrast and Concession),* **Expansion** *(including Conjunction, Instantiation, Restatement, Alternative, ChosenAlternative, and Exception), and* **Co-Occurrence**. *The detailed definitions of these relation types are shown in Table 4.2. The weight of* $\mathsf{R} = \langle \mathsf{E}_h, \mathsf{T}, \mathsf{E}_t \rangle$, *which is denoted as* $w^{(r)}_{\mathsf{R}}$, *is defined by the sum of weights of* $\langle \mathsf{E}_h, \mathsf{T}, \mathsf{E}_t \rangle$ *that appear in the whole corpora.*

4.1.3 ASER Conceptualization

As aforementioned, to overcome the challenge that trivial commonsense is often omitted in humans' communication, we propose to leverage the conceptualization to generalize the knowledge about observed eventualities to unseen ones. For each eventuality $\mathsf{E} \in \mathcal{E}$, whose weight is $w^{(e)}_{\mathsf{E}}$, and we can conceptualize E to E' with confidence $w^{(c)}_{\mathsf{E},\mathsf{E}'}$, we will get a new conceptualized eventuality $\mathbf{E}'$ with the weight $w^{(c)}_{\mathsf{E}'} = w^{(e)}_{\mathsf{E}} \cdot w^{(c)}_{\mathsf{E},\mathsf{E}'}$. Similarly, assume that an edge $\mathsf{R} \in \mathcal{R}$ is $\langle \mathsf{E}_h, \mathsf{T}, \mathsf{E}_t \rangle$ and its weight is $w^{(r)}_{\mathsf{R}}$, and E_h and E_t can be conceptualized to E'_h and E'_t with the confidence $w^{(c)}_{\mathsf{E}_h,\mathsf{E}'_h}$

and $w^{(c)}_{E_t,E'_t}$, respectively. We can then get a new conceptualized edge $\langle E'_h, T, E'_t \rangle$ with the weight $w^{(r)}_{R} \cdot w^{(c)}_{E_h,E'_h} \cdot w^{(c)}_{E_t,E'_t}$. Details about how to leverage an external hypernym knowledge base to get the conceptualized eventualities and how do we determine the confidence scores are presented in Section 4.2.

4.1.4 KG Storage

In total, we use the following three tables of the SQLite database to store ASER.

- *Eventuality*: As aforementioned, all eventualities in ASER are dependency graphs, where vertices are the words and edges are dependency relations. We generate unique "eids" for eventualities by hashing their words, pos-tags and dependencies and store eventualities in an *Eventuality* table with SQLite database where "eids" is the key, and patterns, verb(s), skeleton words, words, pos-tags, dependencies, and frequencies are the other attribute columns.

- *Concept*: To effectively distinguish the eventualities before and after the conceptualization, we store eventualities created by the conceptualization step in another *Concept* table and denote the id as "cid". As the dependency edges are inherited from the original eventualities, we only hash the conceptualized words to generate the "cids." For each concpetualized eventuality, we store its "cid," pattern, frequency, and "eids" of the original eventualities.

- *Relations*: In the end, we store the relations between eventualities in the *Relations* table. For each pair of eventualities (i.e., E_h and E_t), if there is at least an edge between them, we will create an instance and generate a "rid" for them by hashing the concatenation of their "eids". For the storage efficiency and retrieval feasibility, we store all edges and the associated weights between E_h and E_t as well as the eventuality ids of E_h and E_t in that instance.

4.2 Knowledge Extraction

In this section, we introduce the knowledge extraction methodologies for building ASER.

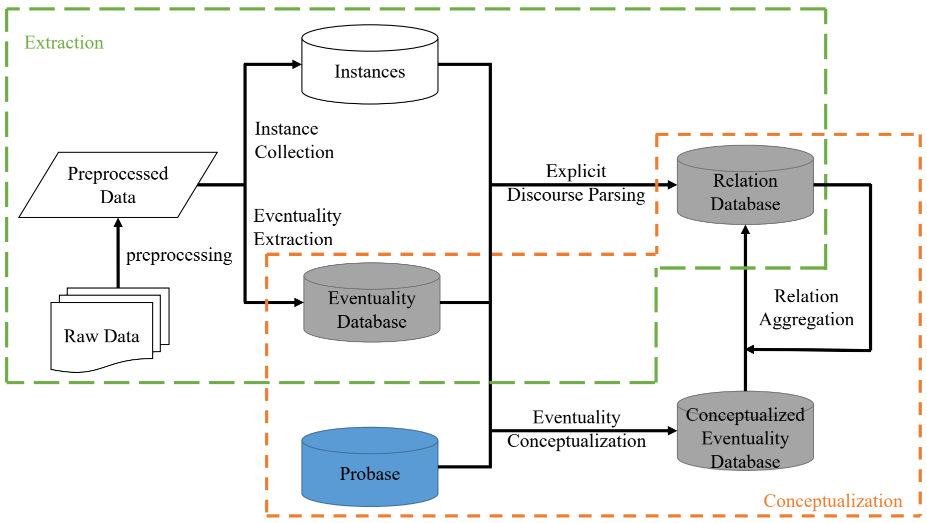


Figure 4.2: ASER construction framework. The extraction and the conceptualization process are shown in the orange dash-dotted and green dashed box respectively. The blue database is Probase and three gray databases are the resulted ASER.

4.2.1 System Overview

The overall framework of our extraction system is shown in Figure 4.2 After collecting the raw corpora, we first preprocess the texts with the dependency parser. Then we perform eventuality extraction with the pattern matching. We collect sentences and adjacent sentence pairs that contain more than two eventualities into a instance collection. After that, we extract discourse relations from these candidate instances with the help of an explicit discourse parser [166]. Considering that the discourse argument span given by the discourse parser might not be identical to the extracted eventualities, we apply token-based Simpson's similarity between the arguments spans and eventualities to determine whether the discourse arguments is enough to represent the meaning of the extracted eventualities, and we only keep the extraction results with the Simpson's similarity larger than 0.8. After the initial ASER construction, we leverage the *IS-A* relations between nouns and named entities from Probase [172] to conduct the conceptualization. In the end, we aggregate relations between conceptualized eventualities by retrieving head and tail eventualities from the conceptualized eventuality database and the eventuality relation database. In the following sub-sections, we will introduce each part of the system separately.

4.2.2 Corpora

To make sure the broad coverage of ASER, we select corpora from different resources (reviews, news, forums, social media, movie subtitles, e-books) as the raw data. The details of these datasets are as follows.

• Yelp: Yelp is a social media platform where users can write reviews for businesses (e.g., restaurants) The latest release of the Yelp dataset[1] contains over five million reviews.

• New York Times (NYT): The NYT [137] corpus contains over 1.8 million news articles from the NYT throughout 20 years (1987 - 2007).

• Wiki: Wikipedia is one of the largest free knowledge dataset. To build ASER, we select the English version of Wikipedia.[2]

• Reddit: Reddit is one of the largest online forums. In this work, we select the anonymized post records[3] over one period month.

• Movie Subtitles: The movie subtitles corpus was collected by [83] and we select the English subset, which contains subtitles for more than 310K movies.

• E-books: The last resource we include is the free English electronic books from Project Gutenberg.[4]

We merge these resources as a whole to perform the knowledge extraction. The detailed statistics are presented in Table 4.3.

4.2.3 Preprocessing

For each document, we aim to extract eventualities, relations between eventuality, conceptualized eventualities, as well as relations between conceptualized eventualities. Based on the consideration of the text parsing complicity and quality, we parse each paragraph[5] instead of a whole document

[1]`https://www.yelp.com/dataset/challenge`

[2]`https://dumps.wikimedia.org/enwiki/`

[3]`https://www.reddit.com/r/datasets/comments/3bxlg7`

[4]`https://www.gutenberg.org/`

[5]As the discourse parser extract discourse relations by the constituency tree of a sentence or trees of adjacent sentences, parsing sentences one by one would miss or misclassify some discourse relations.

Table 4.3: Statistics of used corpora. (M means millions and G means Gigabytes.)

Name	# Sentences	# Tokens	Corpus Size	Category
YELP	54.5 M	838.8 M	2.5G	Reviews
NYT	49.8 M	1,179.4 M	3G	News
Wiki	110.6 M	2,435.4 M	13G	Knowledge
Reddit	253.6 M	3,371.3 M	21G	Forum
Subtitles	444.6 M	3,229.4 M	13G	Movie Scripts
E-books	210.6 M	3,610.0 M	21G	Stories
Overall	1,123.7 M	14,664.2 M	73.5G	-

Algorithm 1 Eventuality Extraction with One Pattern p

```
INPUT: Parsed dependency graph $\mathcal{D}$, center verb $v$, positive dependency edges $\mathcal{P}_p^{(pos)}$, optional edges $\mathcal{P}_p^{(opt)}$, and negative edges $\mathcal{P}_p^{(neg)}$.
OUTPUT: extracted eventuality $E$.
 1: Initialize eventuality edge list $\mathcal{D}'$.
 2: Set the center verb $v$ as the $v_1$ in the pattern $p$.
 3: for each connection $d$ (a relation and the associated word) in positive dependency edges $\mathcal{P}_p^{(pos)}$ do
 4:     if find $d$ in $\mathcal{D}$ then
 5:         Append $d$ in $\mathcal{D}'$.
 6:     else
 7:         Return null.
 8: for each connection $d$ in optional dependency edges $\mathcal{P}_p^{(opt)}$ do
 9:     if find $d$ in $\mathcal{D}$ then
10:         Append $d$ in $\mathcal{D}'$.
11: for each connection $d$ in negative dependency edges $\mathcal{P}_p^{(neg)}$ do
12:     if find $d$ in $\mathcal{D}$ then
13:         Return null.
14: Build eventuality instance $E$ from $\mathcal{D}'$
15: Return $E$
```

with the CoreNLP tool[6] to acquire the lemazied tokens, pos-tags, named entities, the dependency graph, and the constituency tree. Before parsing, we replace urls with a special token $\langle URL \rangle$ and drop tables in Reddit data.

4.2.4 Eventuality Extraction

To make sure that all the extracted eventualities are semantically complete without being too

[6] `https://stanfordnlp.github.io/CoreNLP`

complicated, we design 18 patterns to extract the eventualities via pattern matching. Each of the patterns contains three kinds of dependency edges: positive dependency edges, optional dependency edges, and negative dependency edges. All the positives edges are shown in Table 4.1. Six more dependency relations (`advmod`, `amod`, `nummod`, `aux`, `compound`, and `neg`) are optional dependency edges that can associate with any of the selected patterns. We omit all optional edges in the table because they are the same for all patterns. All other dependency edges are considered as negative dependency edges, which are designed to make sure all the extracted eventualities are semantically complete and all the patterns are exclusive with each other. Take sentence "I have a book" as an example, we will only select <"I," "have," "book"> rather than <"I," "have"> as the valid eventuality, because "have"-dobj-"book" is a negative dependency edge for pattern "s-v."

To extract eventualities from sentence s, considering that s may contains multiple eventualities, we first split it into simple clauses based on the constituency tree. To do so, besides the commonly used *SBAR* node, we also follow previous discourse parsing systems [166] to use a connective classifier to detect possible separators. As a result, we split sentences based on both the subordinate conjunctions and connectives. After that, for each verb v in sentence s, we find the dependency graph $\mathcal{D}$ of the simple clause that contains v. We then try to match $\mathcal{D}$ with all patterns one by one. For each pattern, we put the verb v as the starting point (i.e., v_1 in the pattern), and then try to find all the positive dependency edges. If we can find all the positive dependency edges around the center verb, these match edges and words linked by these edges are considered as potential edges and words of one valid eventuality. Next, other edges and words are added via optional dependency edges. In the end, we will check if any negative dependency edge can be found in the dependency graph. If not, we will keep current edges and words as one valid eventuality. Otherwise, we will disqualify it. The pseudo-code of the eventuality extraction algorithm is in Algorithm 1. The time complexity of eventuality extraction is $\mathcal{O}(|\mathcal{S}| \cdot \overline{|\mathcal{D}|} \cdot \overline{|\mathcal{V}^{(v)}|})$ where $|\mathcal{S}|$ is the number of sentences, $\overline{|\mathcal{D}|}$ is the average number of dependency edges in a dependency parse tree, and $\overline{|\mathcal{V}^{(v)}|}$ is the average number of verbs in a sentence.

4.2.5 Eventuality Relation Extraction

We then introduce how to extract the relations between eventualities. Specifically, we employ an end-to-end discourse parser to extract the discourse relations. The job of the discourse parser is to parse a piece of text into a set of discourse relations between two adjacent or non-adjacent

Algorithm 2 Eventuality Relation Extraction

INPUT: Parsed constituency trees $\mathcal{K}_1$ and $\mathcal{K}_2$ from adjacent sentences.
OUTPUT: Extracted relations $\mathcal{R}$.

```
1:  Initialize relation list $\mathcal{R}$ as empty.
2:  Extract possible connectives $\mathcal{C}$ by a connective extractor given $\mathcal{K}_1$ and $\mathcal{K}_2$.
3:  for each possible connective $c \in \mathcal{C}$ do
4:    if two arguments of $c$ in the same sentence then
5:      Extract $A_1$ and $A_2$ by a SS arguments extractor given $c$ and the sentence.
6:    else
7:      Extract $A_1$ by a PS argument1 extractor given $c$ and $\mathcal{K}_1$
8:      Extract $A_2$ by a PS argument2 extractor given $c$ and $\mathcal{K}_2$
9:    if $A_1$ is not null and $A_2$ is not null then
10:     Classify the relation $y$ by a explicit relation classifier given $c$, $\mathcal{K}_1$, and $\mathcal{K}_2$
11:     Find eventualities $\mathcal{E}_h$ that are extracted from $A_1$
12:     Find eventualities $\mathcal{E}_t$ that are extracted from $A_2$
13:     Set weight $w$ as $1/(|\mathcal{E}_h| \cdot |\mathcal{E}_t|)$
14:     for each eventuality $E_h$ in $\mathcal{E}_h$ do
15:       for each eventuality $E_t$ in $\mathcal{E}_t$ do
16:         Build relation instance $R = \langle E_h, y, E_t \rangle$ with a weight $w$.
17:         Append $R$ in $\mathcal{R}$
18: Return $\mathcal{R}$
```

discourse units. Take sentence "I have a story book but it is not interesting." as an example. Ideally, a good discourse parser would extract "I have a story book" as arg1, "it is not interesting" as arg2, "but" as the connective, and annotate the relation as "Contrast." In our current pipeline, we use the state-of-the-art discourse parser [166], which is pretrained on the CoNLL 2015 Shared Task data (PDTB) [174]. From CoNLL 2015 results,[7] we can find out that this discourse parser can achieve 90.00% and 90.79% F1 scores on the test data from PDTB and the blind test data from Wikinews respectively on the explicit relation classification, but performance drops to 42.72% and 34.45% on the implicit relation classification. Hence, to guarantee the extraction quality, we only consider the explicit discourse relations. In explicit discourse parsing, there are two situations: both arguments are in the same sentence or not. As statistics show that less than 0.1% cases that arguments are located in non-adjacent sentences in the explicit part, we simply assume arguments of which argument1 is located in the same sentence (SS) and the previous sentence (PS). Specifically, the explicit discourse parser is consist of five components: (1) connective extractor to identify whether a word is a possible connective, (2) arg1 position classifier to decide whether the arg1 is located in the same sentence as the connective c or in the previous sentence of c; (3) SS argument

[7] https://www.cs.brandeis.edu/~clp/conll15st/results.html

extractor to extract the spans of two arguments in the same sentence; (4) PS argument extractor to extract the spans of two arguments in adjacent sentences; (5) explicit relation classifier to classify the relation type of c. Extractors in this system are essentially binary classifiers to identify whether a word is a connective or a part of any argument. The pseudo-code of eventuality relation extraction algorithm is shown in Algorithm 2.

As the extracted arguments might not be identical as the extracted eventualities, we use the Simpson's similarity to determine whether the discourse relations between arguments can be assigned to the extracted eventualities:

$$w_{A,E}^{(sim)} = \text{Simpson}(A, E) = \frac{|\mathcal{W}_A \cup \mathcal{W}_E|}{\min\{|\mathcal{W}_A|, |\mathcal{W}_E|\}}, \tag{4.1}$$

where A is an argument, E is an eventuality, $\mathcal{W}_A$ and $\mathcal{W}_E$ are token sets of A and E, $|\cdot|$ is the size of a token set. If the similarity $\text{Simpson}(A, E) \geqslant 0.8$,[8] we consider the argument-level relations relevant to A can be assigned to the eventuality E with a weight $w_{A,E}^{(sim)}$, which is inversely proportional to the size of all matched eventualities $|\mathcal{E}|$.

4.2.6 Enriching ASER with Conceptualization

We then introduce the conceptualization details. For each noun or pronoun in the extracted eventualities, we will try to conceptualized it to a higher level with the following steps. If it is a named entity, we will conceptualized it to the corresponding NER tags. Specifically, we include the 13 NER types: "TIME," "DATE," "DURATION," "MONEY," "PERCENT," "NUMBER," "COUNTRY," "STATE_OR_PROVINCE," "CITY," "NATIONALITY," "PERSON," "RELIGION," and "URL." If it is a single personal pronoun (e.g., "he" or "she"), we will conceptualize it to "PERSON."[9] As all aforementioned conceptualization is designed by experts, we set the conceptualization probability to be 1. If it is a normal noun, we will try conceptualize it with Probase [172]. Specifically, for each noun, we will retrieve its top-five hypernyms (i.e., concepts) and the associated probability from Probase.

Given an eventuality E with m tokens $t_1, t_2, \cdots, t_m$ to be mapped into concept tokens, we

[8]We choose 0.8 because it gives the best balance between quantity and quality.

[9]If there are multiple people in the same edge, we will distinguish them with "PERSON0" and "PERSON1" etc..

conceptualize it to a conceptualized eventuality C with the probability

$$\Pr(\mathsf{C}|\mathsf{E}) = \prod_{i=1}^{m} \Pr(t_i^{(c)}|t_i^{(e)}). \tag{4.2}$$

Here $t_i^{(c)}$ is the corresponding token-level concept for token $t_i^{(e)}$. And $\Pr(t_i^{(c)}|t_i^{(e)})$ is the likelihood for $\langle t_i^{(e)}, \texttt{IS-A}, t_i^{(c)} \rangle$ provided by Probase or 1.0 if $t_i^{(e)}$ can be conceptualized with seed rules. For each conceptualized eventuality C, we would have a list of eventualities $\mathcal{E}_\mathsf{C}$ that can be conceptualized to it. We can then compute the overall weight of C with

$$w_\mathsf{C}^{(c)} = \sum_{\mathsf{E} \in \mathcal{E}_\mathsf{C}} \Pr(\mathsf{C}|\mathsf{E}) \cdot w_\mathsf{E}^{(e)}, \tag{4.3}$$

where $w_\mathsf{E}^{(e)}$ is the weight of E. We then introduce how to construct the edges between a conceptualized eventuality C and an original eventuality E. For any $\mathsf{E}' \in \mathcal{E}_\mathsf{C}$, if there is an edge $\langle \mathsf{E}', \mathsf{T}, \mathsf{E} \rangle$ or $\langle \mathsf{E}, \mathsf{T}, \mathsf{E}' \rangle$, we can then construct a new edge $\langle \mathsf{C}, \mathsf{T}, \mathsf{E} \rangle$ or $\langle \mathsf{E}, \mathsf{T}, \mathsf{C} \rangle$ with the weight $\Pr(\mathsf{C}|\mathsf{E}') \cdot w_{\langle \mathsf{E}', \mathsf{T}, \mathsf{E} \rangle}^{(r)}$ or $\Pr(\mathsf{C}|\mathsf{E}') \cdot w_{\langle \mathsf{E}, \mathsf{T}, \mathsf{E}' \rangle}^{(r)}$, respectively, where $w_\mathsf{R}^{(r)}$ means of weight of the relation R. Similarly, we can construct the edges between two conceptualized eventualities C_h and C_t. For any $\mathsf{E}_h \in \mathcal{E}_{\mathsf{C}_h}$ and $\mathsf{E}_t \in \mathcal{E}_{\mathsf{C}_t}$, if there is an edge $\langle \mathsf{E}_h, \mathsf{T}, \mathsf{E}_t \rangle$, we can then construct a new edge $\langle \mathsf{C}_h, \mathsf{T}, \mathsf{C}_t \rangle$ with the weight $w_{\langle \mathsf{C}_h, \mathsf{T}, \mathsf{C}_t \rangle}^{(r)} = \Pr(\mathsf{C}_h|\mathsf{E}_h) \cdot w_{\langle \mathsf{E}_h, \mathsf{T}, \mathsf{E}_t \rangle}^{(r)} \cdot \Pr(\mathsf{C}_t|\mathsf{E}_t)$.

4.2.7 ASER Building Example

In the end of this section, we use an example to demonstrate the whole extraction pipeline. As shown in Figure 4.3, given a sentence "My army will find your boat. In the meantime, I'm sure we could find you suitable accommodations.",[10] our system will first detect the possible connective "meantime" and split this text into four simple clauses: "My army will find your boat," "In the," "I'm sure," and "we could find you suitable accommodations" with the constituency parsing. After that, our system will leverage the patterns designed in Table 4.1 to extract eventualities from raw text by Algorithm 1. At the same time, two arguments "My army will find your boat" and "we could find you suitable accommodations" are extracted by argument extractors. And this discourse parsing system predicts the corresponding discourse relation as *Synchronous*. As the first and last

[10]This case comes from Movie Subtitles.

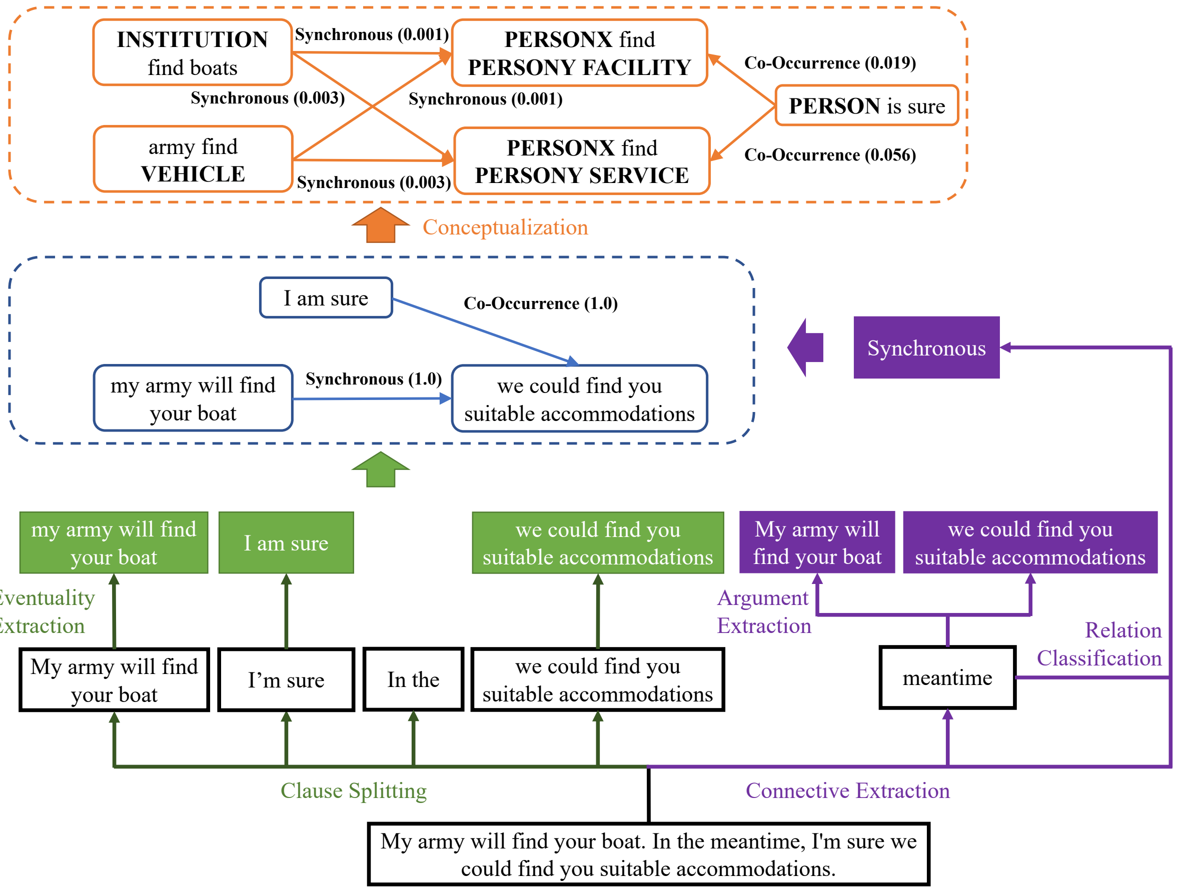

INSTITUTION find boats
Synchronous (0.001)
PERSONX find PERSONY FACILITY
Synchronous (0.003)
Synchronous (0.001)
Co-Occurrence (0.019)
PERSON is sure
army find VEHICLE
PERSONX find PERSONY SERVICE
Synchronous (0.003)
Co-Occurrence (0.056)
Conceptualization
I am sure
Co-Occurrence (1.0)
Synchronous
my army will find your boat
Synchronous (1.0)
we could find you suitable accommodations
my army will find your boat
I am sure
we could find you suitable accommodations
My army will find your boat
we could find you suitable accommodations
Eventuality Extraction
Argument Extraction
Relation Classification
My army will find your boat
I'm sure
In the
we could find you suitable accommodations
meantime
Clause Splitting
Connective Extraction
My army will find your boat. In the meantime, I'm sure we could find you suitable accommodations.

Figure 4.3: ASER building example. The eventuality extraction, the relation extraction, and the conceptualization process are shown in green, violet, and orange colors, respectively. For the clear representation, for each conceptualized eventuality, we only show the skeleton words and hide the optional ones.

extracted eventualities can perfectly match the extracted arguments, we then create an edge ⟨ "my army will find your boat", `Synchronous`, "we could find you suitable accommodations" ⟩. And we also create an edge ⟨ "I am sure," `Co-Occurrence`, "we could find you suitable accommodations" ⟩ because the two eventualities appear in the same sentence. After extracting the original eventualities and edges, we then try to expand it with the conceptualization.[11] For example, "I am sure" can be directly conceptualized as "PERSON is sure" directly because "I" is a personal pronoun. As both of the other two eventualities contain normal nouns (i.e., "army"), these eventualities can be conceptualized to multiple eventualities. After checking Probase, we find out that "army" can be conceptualized to "INSTITUTION" and "ORGANIZATION" with the weights 0.0.058 and

[11]In the real system, we first extract the original ASER, and then apply the conceptualization step over the whole KG. The presented single sentence example is just for the demonstration.

0.038, “boat” can be conceptualized to “VEHICLE” and “ITEM” with the weights 0.059 and 0.049, “accommodation” can be conceptualized to “SERVICE” and “FACILITY” with the weights 0.056 and 0.019 respectively. We show the two most likely results for each original eventuality (if it has multiple possible conceptualization results) in Figure 4.3. In the end, we can construct edges between conceptualized eventualities, where the weights are the product of conceptualization probabilities, e.g., $\langle$ “INSTITUTION find boats”, `Synchronous`, “PERSONX find PERSONY SERVICE” $\rangle$ with the weight $0.058 \times 0.056 = 0.003$, $\langle$ “PERSON is sure,” `Co-Occurrence`, “PERSONX find PERSONY FACILITY” $\rangle$ with the weight $1.000 \times 0.019 = 0.019$.

4.3 ASER Statistics

Table 4.4: Statistics of the eventuality extraction. # Eventuality and # Unique mean the total number and the unique number of extracted eventualities using corresponding patterns or conceptualized eventualities from them.

Pattern	ASER (full)		ASER (core)		ASER (concept)
	# Eventuality	# Unique	# Eventuality	# Unique	# Unique
s-v	351,082,855	100,645,728	260,663,083	14,337,769	1,022,415
s-v-o	284,103,317	159,948,356	139,031,585	18,100,360	8,252,653
s-v-a	11,546,768	6,149,584	5,951,980	752,468	139,087
s-v-v	24,549,946	11,129,566	14,624,526	1,591,424	216,413
s-v-o-o	6,154,685	3,789,253	2,765,728	460,526	514,084
s-v-v-o	29,445,708	18,659,717	12,720,497	2,187,577	1,482,783
s-v-o-v-o	3,863,478	2,674,229	1,462,883	288,326	522,613
s-v-o-v-o-o	91,532	59,290	40,428	8,499	18,461
s-be-a	79,235,136	29,845,112	52,068,570	3,733,978	465,747
s-be-o	98,411,474	53,503,410	49,979,659	6,337,042	2,312,209
s-v-be-a	1,927,990	982,438	1,035,864	123,263	29,738
s-v-be-o	2,322,890	1,574,896	909,250	184,298	139,239
s-v-o-be-a	277,087	191,973	100,917	18,793	6,151
s-v-o-be-o	307,031	231,289	95,815	22,411	32,796
there-be-o	16,021,849	6,642,438	10,013,628	953,041	39,500
spass-v	61,524,872	38,270,144	25,935,769	3,498,516	276,817
spass-v-o	5,519,982	4,129,709	1,677,244	330,229	154,410
spass-v-v-o	257,004	221,820	46,475	11,738	14,901
Overall	976,643,604	438,648,952	579,123,901	52,940,258	15,640,017

In total, we collect 976,643,604 eventualities from the raw documents. We filter those low-frequency eventualities that only appear once and retain 52,940,258 unique eventualities in ASER

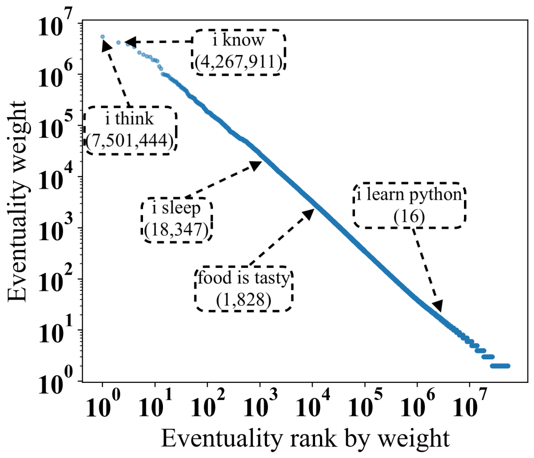
Table 4.5: Statistics of the eventuality relation extraction.

Relation	ASER (full)	ASER (core)	ASER (concept)
Precedence	14,058,213	1,790,016	4,798,015
Succession	4,939,291	663,183	1,963,820
Synchronous	19,464,898	3,123,042	8,013,943
Reason	9,775,829	2,205,076	6,439,128
Result	16,153,925	2,012,311	6,718,666
Condition	18,052,484	3,160,271	8,063,967
Contrast	59,333,901	8,655,661	24,978,311
Concession	5,684,395	477,155	1,499,276
Conjunction	82,121,343	13,978,907	45,597,200
Instantiation	1,278,381	18,496	93,266
Restatement	1,304,095	65,753	242,301
Alternative	3,539,892	583,174	123,883
ChosenAlternative	647,228	35,406	1,843,140
Exception	106,000	20,155	93,412
Co-Occurrence	412,054,590	49,232,161	113,744,814
Overall	648,514,465	86,020,767	224,213,142

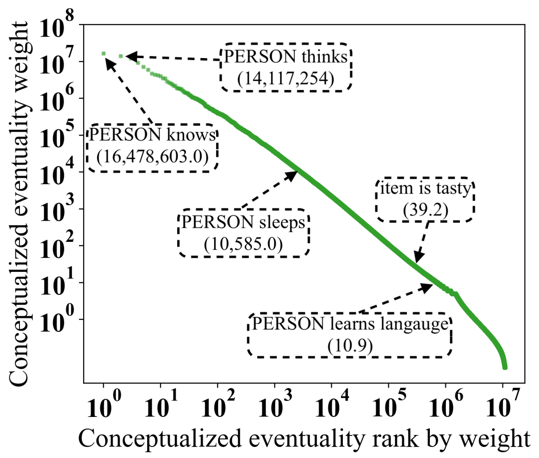
(core). From table 4.4, we can find the "s-v" and "s-v-o" are the most frequent patterns. On the other hand, even though those complex patterns appear relatively less frequent, thanks to the big scale of ASER, they still appears thousands to millions times.

The original eventuality distribution is presented in Figure 4.4 (a). In general, the distribution follows the Zipf's law, where only a small number of eventualities appear many times while the majority of eventualities appear only few times. To better illustrate the distribution of eventualities, we also show several representative eventualities along with their weights and we have two observations. First, eventualities which can be used in general cases, like "i think (7,501,444)" and "i know" (4,267,911) appear much more times than other eventualities. Second, eventualities in ASER are more closely related to our daily life like "i sleep (18,347)" or "food is tasty (1,828)" rather than domain-specific ones such as "iI learn python (16)."

Considering the quality and quantity of conceptualization results, we apply the conceptualization over eventualities whose frequencies are no less than five. After the conceptualization, we get 15,640,017 more unique eventualities. It is obvious that patterns with more nouns (e.g., "s-v-o," "s-v-v-o," "s-be-o") dominate the conceptualized eventualities. The reason is that the conceptualization is only designed for nouns and each noun phase would be replaced with a general noun phase if such hypernym relation appears in Probase. For conceptualized eventualities, we



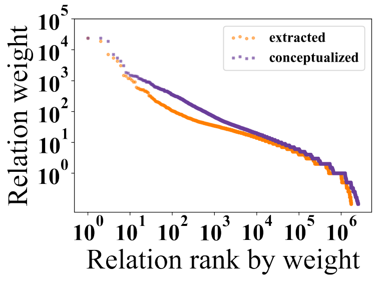
(a) Extracted eventualities



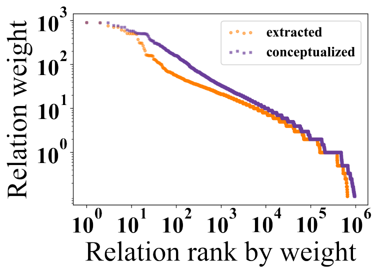
(b) Conceptualized eventualities

Figure 4.4: Eventuality distributions.

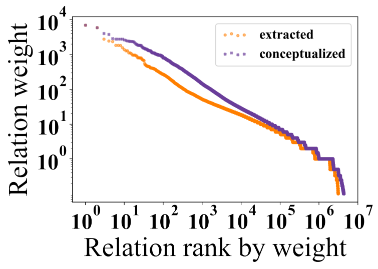
can observe the similar distribution in Figure 4.6 (b). The top three conceptualized eventualities are “PERSON knows” (16,478,603.0), “PERSON thinks” (14,117,254.0), and “PERSON says” (12,113,913.0). Although “i think” (7,501,444) appears the most in the raw data (“i know” appears 3,447,429 times in the raw data), but “you think” (1,444,333), “he thinks” (314,806), “they thinks” (205,432), “we think” (196,633), “it thinks” (174,729), “she thinks” (142,628), ... appear much less than “you know” (4,726,264), “he knows” (396,013), “they know” (260,656), “we know” (457,115), “it knows” (247,409), “she knows” (190,803), ..., respectively. Finally, the weight of “PERSON knows” exceeds that of “PERSON thinks”.



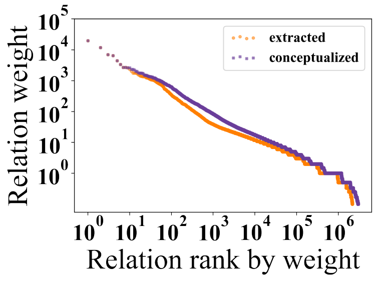
(a) Precedence



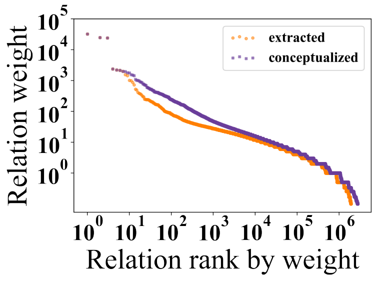
(b) Succession



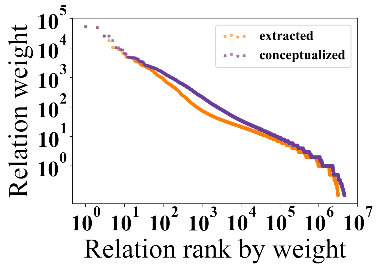
(c) Synchronous



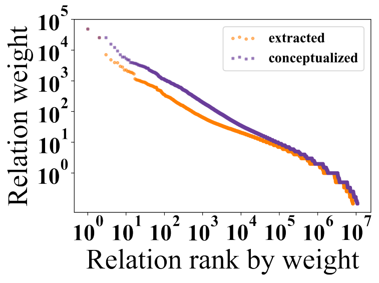
(d) Reason



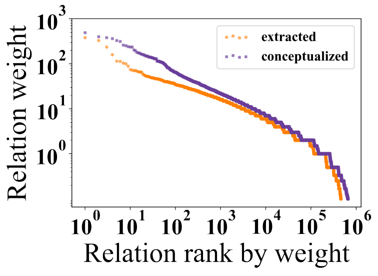
(e) Result



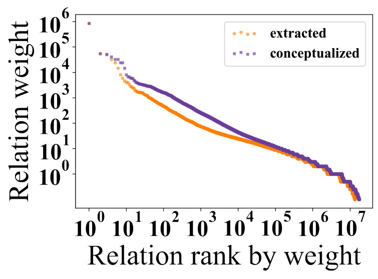
(f) Condition



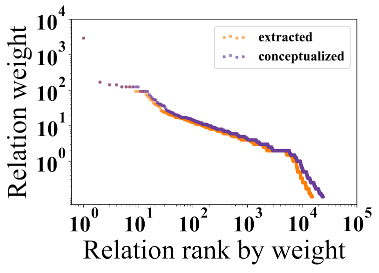
(g) Contrast



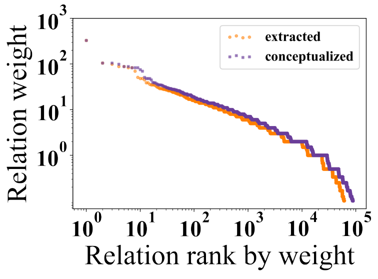
(h) Concession



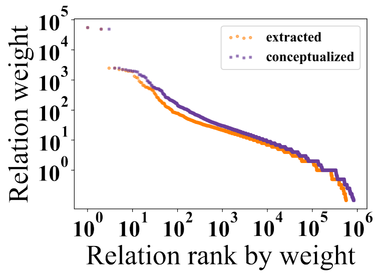
(i) Conjunction



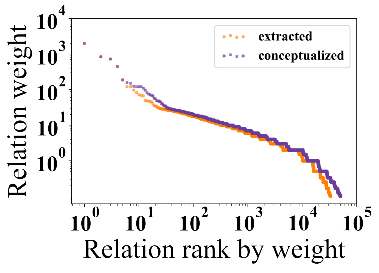
(j) Instantiation



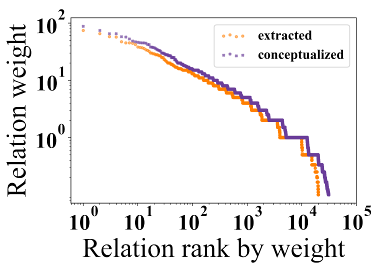
(k) Restatement



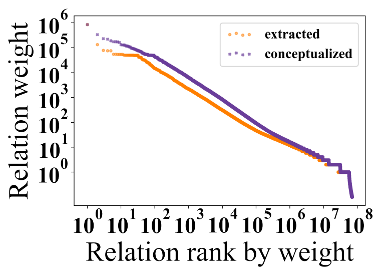
(l) Alternative



(m) ChosenAlternative

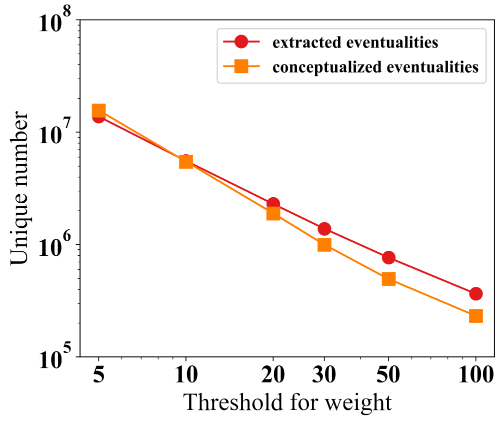


(n) Exception

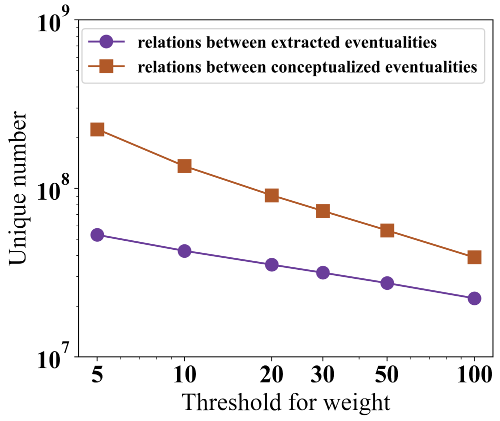


(o) Co-Occurrence

Figure 4.5: Edge distributions by relation types.



(a) Eventualities



(b) Relations

Figure 4.6: Distributions of eventualities and relations.

As for relations, we collect 648,514,465 unique relations from six data resources across different categories. To reduce noises in parsing and extraction, we also filter out relations that $\sum_{T' \in \mathcal{T}} w^{(r)}_{\langle E_h, T', E_t \rangle} <= 1$ where E_h and E_t are the head eventuality and the tail eventuality. Further more, if the head or the tail is filtered out by eventuality filtering, the relation would be dropped as well. Finally, we keep 86,020,767 unique relations in ASER (core), among of which there are 36,788,606 relations belonging to 14 discourse relation types depending on the connectives and arguments, like *Conjunction* (e.g., "and"), *Contrast* (e.g., "but"), *Condition* (e.g., "if"), *Synchronous* (e.g., "meanwhile"), *Reason* (e.g., "because"), *Result* (e.g., "so"). When we filter out more low-frequency eventualities, the number of relations decreases slightly. For example, when we keep high-frequency eventualities whose frequencies are no less than five, 26.0% of eventualities (13,766,746) and 61.5% of relations (88,629,385) are preserved. We apply the conceptualization over these preserved eventualities and relations based on quantity and quality considerations. And we get 15,640,017 unique conceptualized eventualities and 224,213,142 relations between these conceptualized eventualities. It is interesting to see four-fold increases on *Instantiation* and *Exception* which have the lowest numbers and a one-fold increase on *Co-Occurrence* which is the major relation. Overall, we aggregate 260.7% relations between conceptualized eventualities.

To better understand the distributions of extracted and conceptualized knowledge, we show the number of eventualities and edges over different filtering thresholds in Figure 4.6 (a) and Figure 4.6 (b), respectively. For the extracted knowledge, the number of eventualities and relations decreases exponentially when the threshold ranges from 100 to 5. For the conceptualized knowledge, the rate of diminishing becomes further larger. When the threshold is less than 10, the conceptualized eventuality size outperforms the original size. But it is significantly less than the size of extracted eventualities as the threshold is larger than 10. On the other hand, the number of relations of conceptualized eventualities always exceeds that the original relation size, which results in a denser conceptualized knowledge graph. Above all, conceptualization is an extraordinary and excellent method to mine commonsense knowledge.

4.4 Evaluation

In this section, we leverage human annotation to evaluate the quality of ASER from following perspectives:

1. **Eventuality Extraction**: We first evaluate how well the extracted eventualities can represent the semantics of the original sentence. For example, if the original sentence is "The kid goes to study", eventuality "kid-go-to-study" with pattern "s-v-v" can fully represent the semantics, but eventuality "kid-go" with the pattern "s-v" cannot. We show the percentage of all extracted eventualities that can fully represent the core semantic of the original sentences based different eventuality patterns.

2. **Lower-order Selectional Preference**: Besides the extraction quality, we also care about how well the eventuality statistics in ASER can reflect human's selectional preference. For example, the frequency of "I eat food" should be higher than "I eat house." As such preference appears inside eventualities, we denote them as the lower-order selectional preference.

3. **Discourse Extraction**: After evaluating the eventualities, we then evaluate how well the extracted edges can correctly represent the discourse relations in the original sentence. For example, assume that the original sentence is "he went to school while I was still preparing the breakfast." and we have successfully extracted two eventualities "he went to school" and "I was preparing breakfast", the correct discourse relation between them should be "Synchronous" rather than "Contrast." In this evaluation, we report the accuracy based on different discourse relations.

4. **Higher-order Selectional Preference**: Last but not least, we annotate whether edge frequencies in ASER can reflect the higher-order selectional preference among eventualities or not.

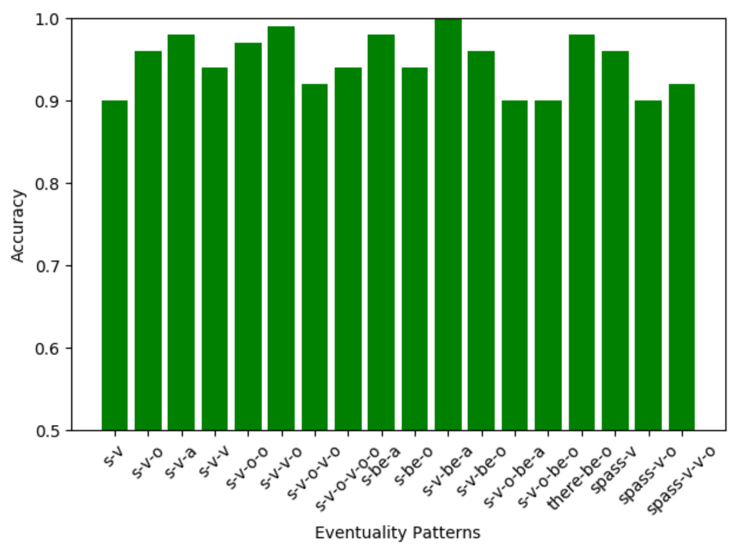


Figure 4.7: Human annotation of eventuality extraction quality.

For example, the frequency of "I am hungry"-`Result`-"I eat food" should be larger than "I am hungry"-`Reason`-"I eat food."

Evaluation details and result analysis are as follows.

4.4.1 Eventuality Extraction

To evaluate the correctness of the selected eventuality patterns and effectiveness of the extraction algorithms, we first employ the Amazon Mechanical Turk platform (MTurk)[12] to evaluate the quality of eventuality extraction. For each eventuality pattern, we randomly select 50 extracted eventualities and then provide these extracted eventualities along with their original sentences to the annotators. For each pair of eventuality and sentences, the annotators are asked to label whether the extracted eventuality phrase can fully and precisely represent the semantic meaning of the original sentence. If so, they should label them with "Valid." Otherwise, they should label it with "Not Valid." For each eventuality, we invite six workers to label and if at least four of them label it as "Valid," we will consider it to be valid. In total, we collected 5,400 annotations. To make sure the high annotation quality, we require all the annotators to be the master annotator on the MTurk.

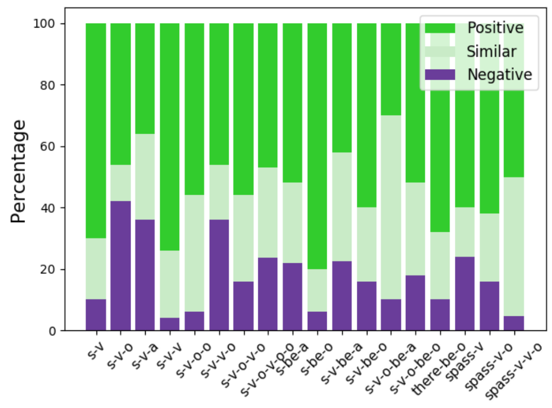
The annotation results are shown in Figure 4.7. From the result we can see that all patterns achieve over 90% accuracy, which demonstrates the high quality of the selected patterns and the

[12]`https://www.mturk.com/`

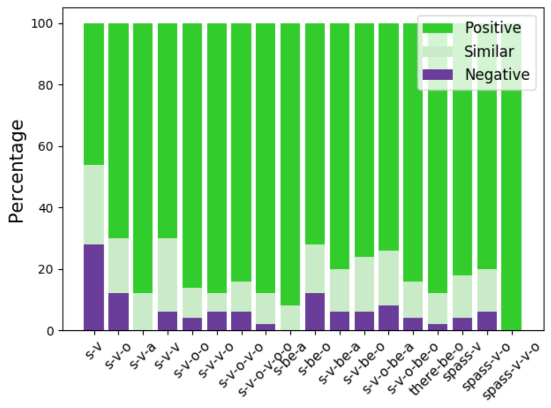
association extraction algorithm. As introduced in Algorithm 1, to guarantee the quality of extracted eventualities, we require the extraction algorithm to be strict and selective. Specifically, if there is an extra dependency edge not in the positive or possible relations of a corresponding pattern, we will discard the whole sentence. By doing so, even though we sacrifice the overall recall, we guarantee the high accuracy. Luckily, as our approach is unsupervised, we can remedy the recall problem with larger scale corpus. Among the 18 patterns, we notice that the more complex patterns tend to have relatively lower accuracy. This makes sense because the more complex an extracted eventuality is, the more likely that some of the words are redundant to the eventuality semantics, and thus the annotator may think that the extracted eventuality is not elegant enough.

4.4.2 Low-order Selectional Preference

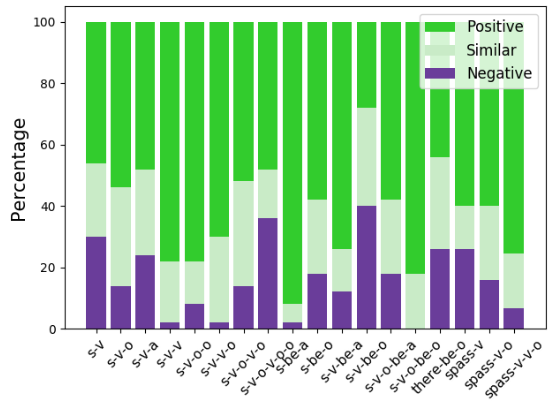
To evaluate whether the eventuality frequencies in ASER can reflect human's low-order selectional preference, we first compare the plausibility of more frequent eventualities versus less frequent ones. For each eventuality pattern, we randomly select 50 eventuality pairs such that they only have one-word difference but with significant frequency difference. Specifically, we require the frequency of the high frequent one to be larger than five, which is the medium frequency of all eventualities, and the frequency of the high frequent one must be at least five times larger than the frequency of the low frequent one. For each eventuality, we invite six annotators from MTruk to ask them which one of the eventuality seems more plausible to them. If more annotators agree that the more frequent one makes more sense, we will label that pair as positive correlation. On the other hand, if more annotator agrees that the less frequent makes more sense, we will label that pair as negative correlation. If the voting draws, we will label it as similar. As this evaluation fails to consider the eventualities with the frequency zero (i.e., they do not exist in ASER) and whether an eventuality exist or not is also a good preference indicator, we add another evaluation to prove that. For each eventuality pattern, we randomly select 50 eventualities. And for each eventuality, we randomly select an negative example by randomly changing a word inside the eventuality with another word of the same postag label such that their grammar structure is the same. We also conduct filtering to guarantee the negative examples do not appear in ASER. Last but not least, to show the influence of the conceptualization, we conduct the aforementioned two experiments on



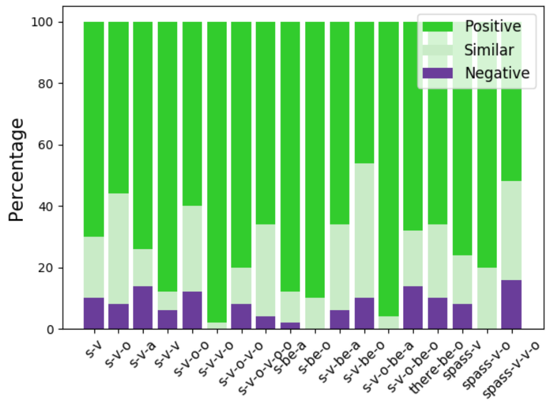
(a) High Frequency VS Low Frequency (before).



(b) Exist VS Non-exist (before).



(c) High Frequency VS Low Frequency (after).



(d) (C) Exist VS Non-exist (after).

Figure 4.8: Human annotation of lower selectional preference in ASER. Experiments on the eventualities before and after the conceptualization are denoted with (O) and (C), respectively. Green color indicates the number of eventuality pairs that the more frequent eventuality makes more sense, and red color indicates the number of eventualities pairs the less frequent eventuality makes more sense.

both the original ASER before the conceptualization and the final one after the conceptualization.[13]

We present the annotation results in Figure 4.8 and indicate the experiments on the eventualities before and after the conceptualization with (O) and (C), respectively. Green color indicates the number of eventuality pairs that the more frequent eventuality makes more sense, and red color indicates the number of eventualities pairs the less frequent eventuality makes more sense. From the result in Figure 4.8 (a), we can see that more than 70% of the eventuality pairs as positive correlated, which is consistent with the previous study on the correlation between frequency and

[13]For the experiment on the ASER after the conceptualization, we only sample the conceptualized eventualities and ignore the original ones.

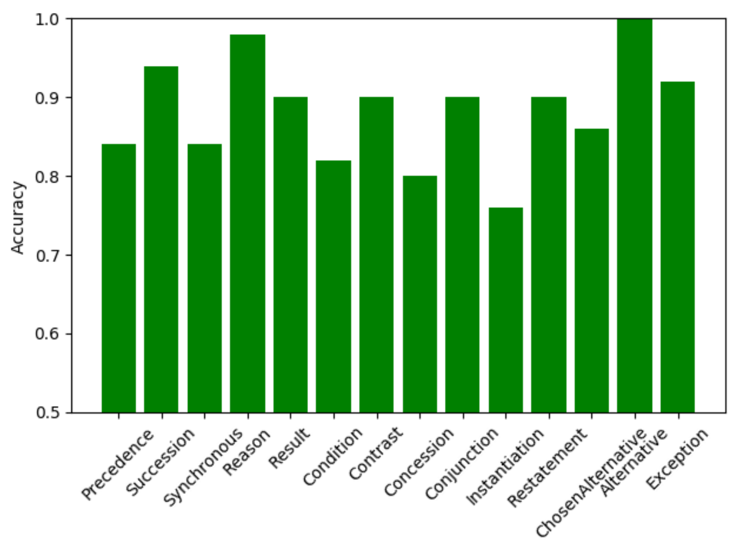


Figure 4.9: Human annotation of discourse relation extraction quality.

selectional preference [182]. At the same time, we also observe that about 30% of the less frequent eventualities are also quite plausible, which is mainly because the frequency of an eventuality is also severely influenced by the rareness of the words inside the eventuality. For example, the eventuality "I eat avocado" appears much less than "I eat apple" because avocado is much more rare than apple rather than "I eat apple" makes more sense than "eat avocado." The results in Figure 4.8 (b) helps prove that the low-frequent eventualities still contains rich low-order selectional preference because for more than 90% of the pairs, the randomly extracted pairs in ASER makes more sense than those out of ASER. In the end, the experimental results in Figure 4.8 (c) and (d) show that even though the conceptualization process significantly improve the coverage of ASER, it would not hurt the overall quality. This is mainly because during the conceptualization step, we carefully design the new weights based on the original weight and the confidence scores provided by Probase [172].

4.4.3 Relation Extraction

Besides the eventuality extraction, we also care about the extraction quality of the discourse relations between them. For each relation type, we randomly select 50 edges and the corresponding sentences and for each pair of them, we generate a question by asking the annotators if they think the extracted discourse relation can represent the correct relation in the original sentence. If so, they should label as "Valid." Otherwise, they should label it as "Not Valid'.' Similar to the eventuality extraction experiment, we invite six annotators for each edge and is more than four of them agree that the extracted relations is "Valid," we will label it as "Valid."

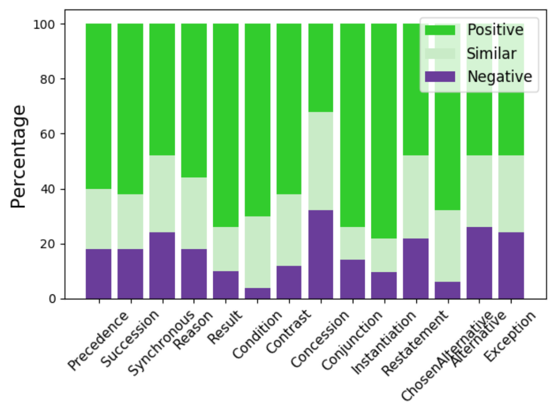
From the results in Figure 4.9 we can see that the overall accuracy is about 80%, which is consistent with the reported performance of the used discourse relations extraction system [166]. Besides that, we also notice that the model performance varies on different relation types. For example, the model tends to perform well on simple types such as "Reason" and "Alternative" because the popular connectives (i.e., "because" and "or") are less ambiguous. As a comparison, when the connective is more ambiguous (e.g., "while" for "Synchronous"), the overall performance will drop.

4.4.4 Higher-order Selectional Preference

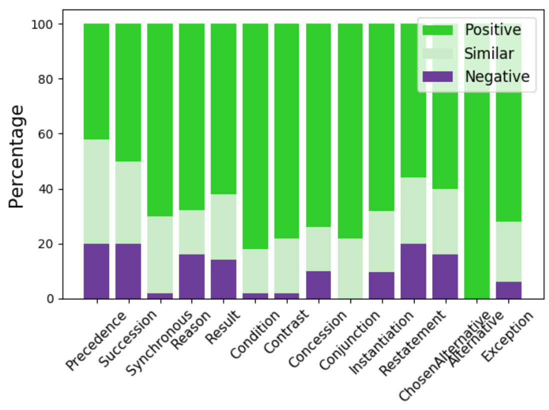
In the end, we evaluate whether the edge frequency in ASER can be used to reflect human's high-order selectional preference about eventualities. Similar to the evaluation on the lower-order selectional preference, we conduct two experiments (i.e., (1) High frequency VS Low Frequency; (2) Exist VS None-exist) on the ASER before and after the conceptualization.[14] For the "High frequency VS Low Frequency" experiment, we randomly sample 50 edge pairs for each relation type such that the two edges in each pair share the same head eventuality, relation type, but different tail eventuality (e.g., "I am hungry"-`Result`-"I eat food" versus "I am hungry"-`Result`-"I exercise"). More importantly, the two sampled edge should have significantly different frequencies. Specifically, we require the frequency of the high frequent one to be larger than five and the frequency of the high frequent one must be at least five times larger than the frequency of the low frequent one. For the "Exist VS None-exist" experiment, for each relation type, we first randomly sample 50 edges, and then for each edge, we randomly replace a single word of the the tail eventuality such that the new tail is very similar to the original one but the created edge does not exist in ASER.

The annotations results are presented in Figure 4.10. In general, we can make the similar observations as the lower SP that the correlation is more significant when we compare the existing and non-existing edges. Besides that, the experiments on the conceptualized ASER help demonstrate that the conceptualization module will not influence the overall quality of edge frequencies.

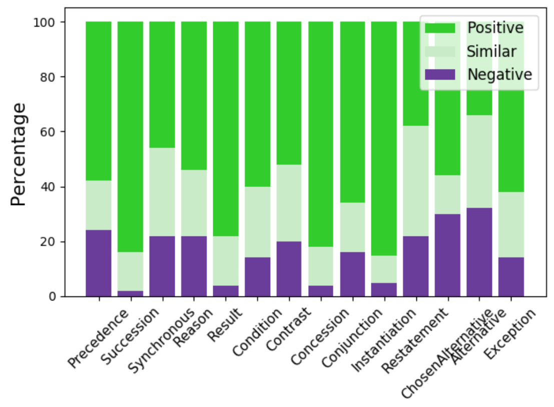
[14]For the experiment on the ASER after the conceptualization, we only sample the conceptualized eventualities and ignore the original ones.



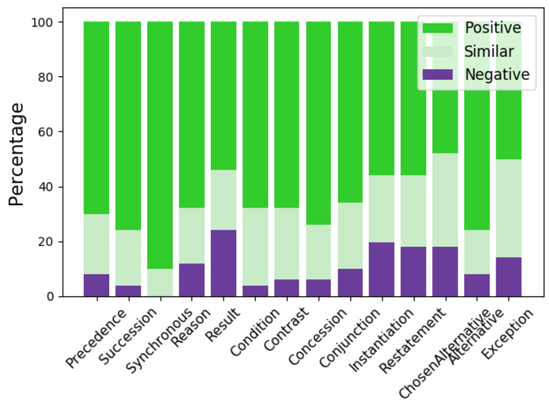
(a) High Frequency VS Low Frequency (O).



(b) Exist VS Non-exist (O).



(c) High Frequency VS Low Frequency (C).



(d) Exist VS Non-exist (C).

Figure 4.10: Human annotation of the higher selectional preference in ASER. Experiments on the eventualities before and after the conceptualization are denoted with (O) and (C), respectively. Green color indicates the number of edge pairs that the more frequent edge makes more sense, and red color indicates the number of edge pairs the less frequent edge makes more sense.

4.5 Applications

In this section, we introduce the application of the selectional preference knowledge contained in ASER on downstream tasks. First of all, as shown in [182, 185], even with a trivial approach like string match, such knowledge can be used to answer a subset of the hard commonsense reasoning benchmark (winograd schema challenge [72]) with high precision. Besides that, the ASER knowledge can also help the machines to understand longer documents. For example, as shown in [184], after converting the ASER knowledge into the ConceptNet format, we can apply that knowledge on the commonsense reading comprehension task [104] and achieve more significant improvement than the original ConceptNet [84]. Last but not least, as the higher-order selectional preference reflects humans' understandings about daily life, we can apply them to help understand the daily dialogue and generate better response. For example, on the daily dialogue dataset [76], compared with adding the original conceptnet, adding ASER knowledge can double the improvement in terms of the BLEU score. We leave more applications on other natural language processing and understanding tasks as the future work.

CHAPTER 5

TRANSFERABILITY FROM SELECTIONAL PREFERENCE TO COMMONSENSE KNOWLEDGE

In the last chapter of Part 1, we conduct extensive experiments to demonstrate the selectional preference knowledge in ASER is indeed the commonsense we want. Commonsense knowledge is defined as the knowledge that people share but often omit when communicating with each other. In their seminal work, [84] defined commonsense knowledge as the knowledge "used in a technical sense to refer to the millions of basic facts and understandings possessed by most people." After around 20 years of development, ConceptNet 5.5 [154], built based on the original ConceptNet [84], contains 21 million edges connecting over 8 million nodes. However, most of the knowledge assertions in ConceptNet 5.5 are still facts about entities integrated from other knowledge sources. The core of ConceptNet, which is inherited from the Open Mind CommonSense (OMCS) project [84], only contains 600K pieces of high-quality commonsense knowledge in the format of tuples, e.g., ("song," *UsedFor*, "sing").

In this chapter, we propose to investigate if we can efficiently convert the selectional preference knowledge in ASER, which is essentially a linguistic graph (i.e., all nodes and edges are defined with linguistic relations), into commonsense knowledge. Specifically, we develop an algorithm for discovering patterns from the overlap of existing commonsense and linguistic knowledge bases and use a commonsense knowledge ranking model to select the highest-quality extracted knowledge. As a result, we can build TransOMCS, a new commonsense knowledge graph.

5.1 Problem Definition

We start by defining the task of mining commonsense knowledge from linguistic graphs. Given a seed commonsense knowledge set $\mathcal{C}$ (which contains m tuples) and a linguistic graph set $\mathcal{G}$ (which contains n linguistic graphs G) with $m \ll n$. Each commonsense fact is in a tuple format $(h, r, t) \in \mathcal{C}$, where $r \in \mathcal{R}$, the set of human-defined commonsense relations (e.g., "UsedFor," "CapableOf,"

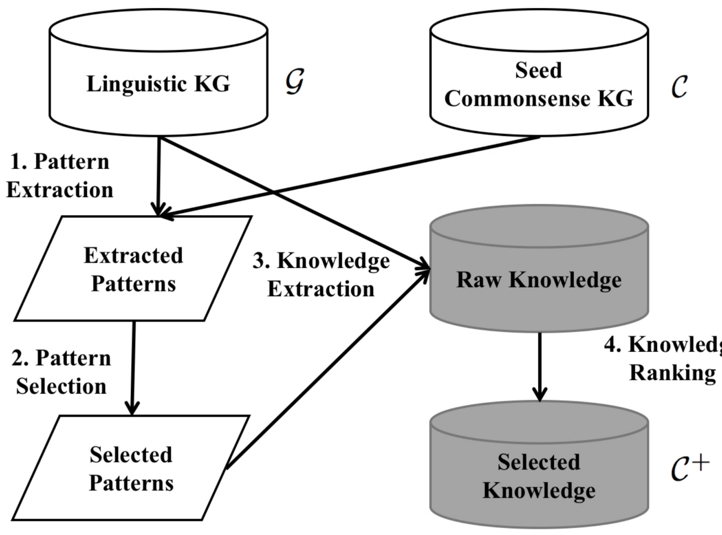


Figure 5.1: The overall framework.

"AtLocation," "MotivatedByGoal"), and h and t are arbitrary phrases. Our objective is to infer a new commonsense knowledge set $\mathcal{C}^+$ (with m^+ pieces of commonsense knowledge) from $\mathcal{G}$ with the help of $\mathcal{C}$ such that $m^+ \gg m$.

5.2 A Knowledge Transformation Framework

The proposed framework is shown in Figure 5.1. In the beginning, for each seed commonsense tuple $(h, r, t) \in \mathcal{C}$, we match and select supporting linguistic graphs that contain all the terms in h and t, and then extract the linguistic patterns for each commonsense relation with the matched commonsense tuple and linguistic graph pairs. Next, we use a pattern filtering module to select the highest quality patterns. Finally, we train a discriminative model to evaluate the quality of the extracted commonsense knowledge. The details are as follows.

5.2.1 Knowledge Resources

We first introduce the details about the selected commonsense and linguistic knowledge resources. For the seed commonsense knowledge, we use the English subset of ConceptNet 5.5 [154]. Following conventional approaches [135, 11], we consider only the relations covered by the original OMCS project [84], except those with vague meanings (i.e., "RelatedTo") or those well-acquired

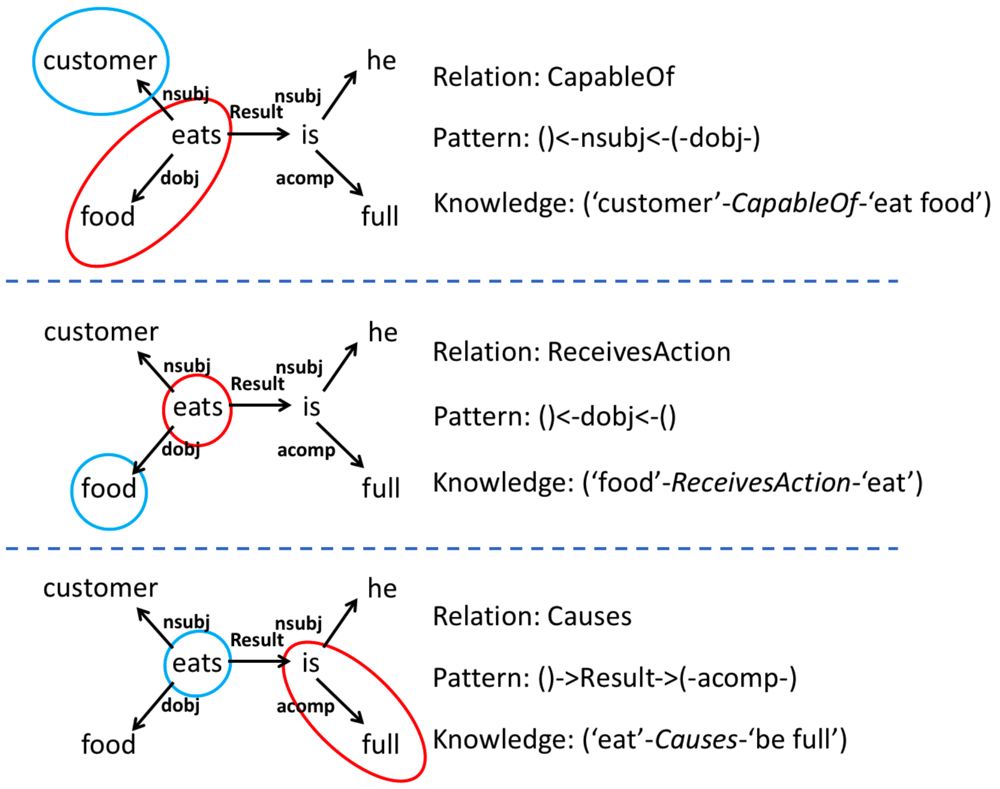

customer
nsubj
eats
Result
is
nsubj
he
dobj
food
acomp
full
Relation: CapableOf
Pattern: ()<-nsubj<-(-dobj-)
Knowledge: ('customer'-*CapableOf*-'eat food')
Relation: ReceivesAction
Pattern: ()<-dobj<-()
Knowledge: ('food'-*ReceivesAction*-'eat')
Relation: Causes
Pattern: ()->Result->(-acomp-)
Knowledge: ('eat'-*Causes*-'be full')

Figure 5.2: Example of linguistic graphs and extracted patterns for different commonsense relations, which are extracted with the matching of words in seed commonsense tuples and the graphs. Given a linguistic graph as the input, these patterns can be applied to extract OMCS-like commonsense knowledge. Extracted head and tail concepts are indicated with blue and red circles respectively.

by other knowledge resources (i.e., "IsA"[1]). In total, 36,954 words, 149,908 concepts, and 207,407 tuples are contained in the selected dataset as $\mathcal{C}$. For the linguistic knowledge resource, we use the core subset of ASER [185] with 37.9 million linguistic graphs[2] to form $\mathcal{G}$.

5.2.2 Pattern Extraction

Given a matched pair of a commonsense tuple $(h, r, t) \in \mathcal{C}$ and a linguistic graph $G \in \mathcal{G}$, the goal of the pattern extraction module is to find a pattern over linguistic relations such that given r, we can accurately extract all the words in h and t from G. Here, we formally define each pattern P as follows:

[1]The extraction of "IsA" relations belongs to the task of hyponym detection and such knowledge has been well preserved by knowledge resources like Probase [172].

[2]As both the internal structure of eventualities and external relations between eventualities could be converted to commonsense knowledge, we treat all the eventualities and eventuality pairs in ASER as the linguistic graphs.

Definition 5.2.1 *Each pattern* P *contains three components: head structure* $\mathsf{p_h}$*, tail structure* $\mathsf{p_t}$*, and internal structure* $\mathsf{p_i}$*.* $\mathsf{p_h}$ *and* $\mathsf{p_t}$ *are the smallest linguistic sub-graph in* G *that can cover all the words in* h *and* t*, respectively.* $\mathsf{p_i}$ *is shortest path from* $\mathsf{p_h}$ *to* $\mathsf{p_t}$ *in* G.

First, we extract $\mathsf{p_h}$ and $\mathsf{p_t}$ from G to cover all the words in h and t. We take the head pattern as an example. For each word in h, we first find its position in G. To avoid any ambiguity, if we find more than one match in G, we discard the current pair and record no pattern. Then, with the position of the first word in h as the start node, we conduct a breadth-first search (BFS) algorithm over G to find a sub-structure of G that covers all and only the words in h. If the BFS algorithm finds such a sub-structure, we treat it as $\mathsf{p_h}$, otherwise, we discard this example and return no pattern. We extract the tail pattern $\mathsf{p_t}$ with G and t in a similar way. After extracting $\mathsf{p_h}$ and $\mathsf{p_t}$, we then collect the internal structure $\mathsf{p_i}$, which is the shortest path from $\mathsf{p_h}$ to $\mathsf{p_t}$ over G. To do so, we collapse all the nodes and edges in $\mathsf{p_h}$ and $\mathsf{p_t}$ into single "head" and "tail" nodes, respectively. Then we use "head" as the starting node to conduct a new BFS to find the shortest path from node "head" to "tail." We aggregate $\mathsf{p_h}$, $\mathsf{p_i}$, and $\mathsf{p_t}$ together to generate the overall pattern P. Examples of patterns are shown in Figure 5.2. For each commonsense relation r, we first collect all the commonsense tuples of relation r in $\mathcal{C}$ to form the subset $\mathcal{C}^r$. Then for each $(h, r, t) \in \mathcal{C}^r$, we go over the whole $\mathcal{G}$ to extract matched patterns with the aforementioned algorithm. The time complexity of the overall algorithm is $O(|\mathcal{C}| \cdot |\mathcal{G}| \cdot N^2)$, where $|\mathcal{C}|$ is the size of $\mathcal{C}$, $|\mathcal{G}|$ is the size of $\mathcal{G}$ and N is the maximum number of the nodes in G.

5.2.3 Pattern Selection and Knowledge Extraction

We evaluate the plausibility of pattern P regarding commonsense relation r as follows:

$$P(P|r) = \frac{F(P|r)}{\sum_{P' \in \mathcal{P}^r} F(P'|r)}, \tag{5.1}$$

where $F(P|r)$ is the scoring function we use to determine the quality of P regarding r and $\mathcal{P}^r$ is the set of patterns extracted for r. We design $F(P|r)$ as follows:

$$F(P|r) = C(P|r) \cdot L(P) \cdot U(P|r), \tag{5.2}$$

where $C(P|r)$ indicates counts of observing P regarding r, $L(P)$ indicates the length of P to encourage complex patterns, and $U(P|r) = \frac{C(P|r)/\sqrt{|\mathcal{C}^r|}}{\sum_{r' \in \mathcal{R}} C(P|r')/\sqrt{|\mathcal{C}^{r'}|}}$ is the uniqueness score of P regarding

r. We select the patterns with the plausibility score higher than 0.05. On average, 2.8 high confident patterns are extracted for each relation. After extracting the matched patterns, we then apply the extracted patterns to the whole linguistic graph set $\mathcal{G}$ to extract commonsense knowledge. For each $G \in \mathcal{G}$ and each relation r, we go through all the extracted patterns $P \in \mathcal{P}^r$. If we can find a matched P in G, we will then extract the corresponding words of p_h and p_t in G as the head words and tail words, respectively. The extracted head and tail word pairs are then considered as a candidate knowledge, related via r.

5.2.4 Commonsense Knowledge Ranking

To minimize the influence of the pattern noise, we propose a knowledge ranking module to rank all extracted knowledge based on the confidence. To do so, we first use human annotators to label a tiny subset of extracted knowledge and then use it to train a classifier. We assign a confidence score to each extracted knowledge via the learned classifier.

Dataset Preparation

For each commonsense relation type, we randomly select 1,000 tuples[3] to annotate. For each tuple, five annotators from Amazon Mechanical Turk are asked to decide if they think the tuple is a plausible commonsense statement. If at least four annotators vote for plausibility, we label that tuple as a positive example. Otherwise, we label it as a negative example. Additionally, we include all the matched examples from OMCS as positive examples. In total, we obtain 25,923 annotations for 20 commonsense relations. Among these annotations, 18,221 are positive examples and 7,702 are negative examples. On average, each tuple has 12.7 supporting linguistic graphs. We randomly select 10% of the dataset as the test set and the rest as the training set.[4]

Task Definition

The goal of this module is to assign confidence scores to all extracted knowledge so that we can rank all the knowledge based on their quality. As for each extracted tuple, we may observe multiple

[3]For relation “DefinedAs” and “LocatedNear,” as our model only extracted 26 and 7 tuples respectively, we annotate all of them and exclude them when we compute the overall accuracy in Section 5.3.

[4]As the annotated dataset is slightly imbalanced, when we randomly select the test set, we make sure the number of positive and negative examples are equal.

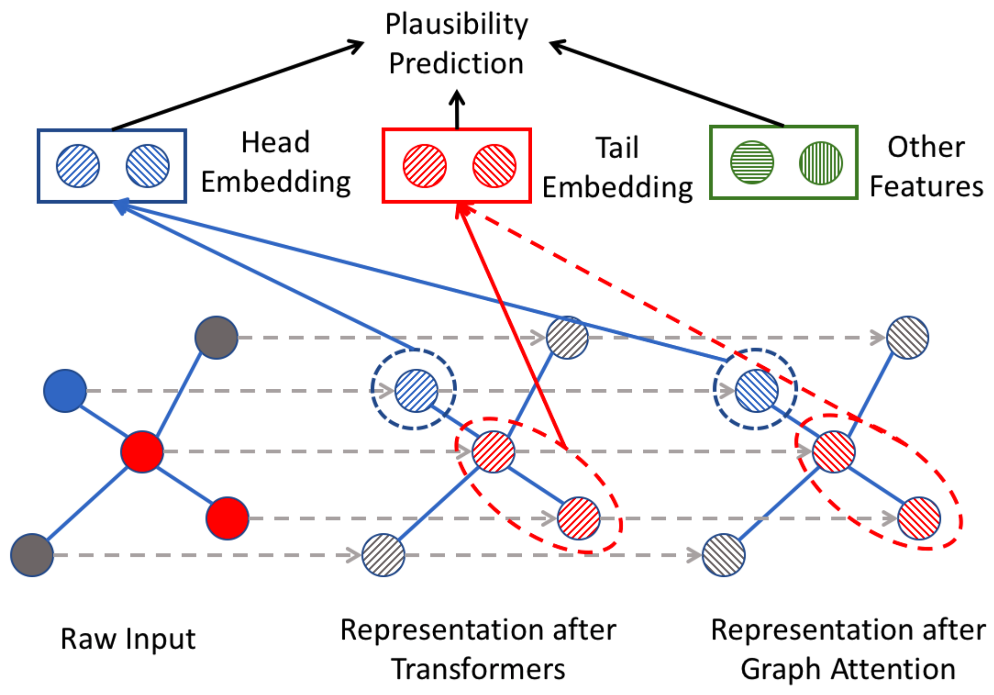


Figure 5.3: Demonstration of the model. The blue and red colors denote the head words and tail words, respectively. The gray and green colors and indicate the other words and features, respectively.

supporting linguistic graphs. We model it as a multi-instance learning problem. Formally, we define the overall annotated knowledge set as $\mathcal{K}$. For each $k \in \mathcal{K}$, denote its supporting linguistic graph set as $\mathcal{G}^k$. We use $F(k|\mathcal{G}^k)$ to denote the plausibility scoring function of k given $\mathcal{G}^k$, and it can be defined as follows:

$$F(k|\mathcal{G}^k) = \frac{1}{|\mathcal{G}^k|} \cdot \sum_{g \in \mathcal{G}^k} f(k|g), \tag{5.3}$$

where $|\mathcal{G}^k|$ is the number of graphs in $\mathcal{G}^k$ and $f(k|g)$ is the plausibility score of k given g.

Model Details

The proposed model, as shown in Figure 5.3, contains three components: transformer, graph attention and the final plausibility prediction.

Transformer: We use the transformer module as the basic layer of our model. Formally, assuming g contains n words $w_1, w_2, ..., w_n$, we denote the representation of all words after the transformer module[5] as $\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_n$. In our model, we adopt the architecture of BERT [25] and their pre-trained

[5]For technical details of the Transformer network, you may refer to the original paper [164].

	Random	BERT	Proposed model
Accuracy	50%	70.91%	73.23%

Table 5.1: Performance of different plausibility prediction models.

parameters (BERT-base) as the initialization.

Graph Attention: Different from conventional text classification tasks, linguistic relations play a critical role in our task. Thus in our model, we adopt the graph attention module [165] to encode the information of these linguistic relations. For each w in g, we denote its representation as $\hat{\mathbf{e}}$, which is defined as follows:

$$\hat{\mathbf{e}} = \sum_{\mathbf{e}' \in N(\mathbf{e})} a_{\mathbf{e},\mathbf{e}'} \cdot \mathbf{e}', \tag{5.4}$$

where $N(e)$ is the representation set of words that are connected to w and $a_{\mathbf{e},\mathbf{e}'}$ is the attention weight of $\mathbf{e}'$ regarding $\mathbf{e}$. Here, we define the attention weight as:

$$a_{\mathbf{e},\mathbf{e}'} = \frac{e^{\text{NN}_a([\mathbf{e},\mathbf{e}'])}}{\sum_{\bar{\mathbf{e}} \in N(\mathbf{e})} e^{\text{NN}_a([\mathbf{e},\bar{\mathbf{e}}])}}, \tag{5.5}$$

where [.] indicates vector concatenation and NN_a is the dense neural network we use to predict the attention weight before the softmax module. After the graph attention module, the representation of words are then denoted as $\hat{\mathbf{e}_1}, \hat{\mathbf{e}_2}, ..., \hat{\mathbf{e}_n}$.

Plausibility Prediction: In the last part of our model, we first concatenate $\mathbf{e}$ and $\hat{\mathbf{e}}$ together for all the words and then create head embedding $\mathbf{o}_{head}$ and tail embedding $\mathbf{o}_{tail}$ using mean pooling over $[\mathbf{e}, \hat{\mathbf{e}}]$ of all words appear in the head or tail respectively. Besides embedding features, two important features, graph frequency $\mathbf{o}_{fre}$ (how many times this graph appears) and graph type $\mathbf{o}_{type}$ (whether this graph is node or edge), are also considered for the final plausibility prediction. We define plausibility prediction function $f(k|g)$ as:

$$f(k|g) = \text{NN}_p([\mathbf{o}_{head}, \mathbf{o}_{tail}, \mathbf{o}_{fre}, \mathbf{o}_{type}]), \tag{5.6}$$

where NN_p is a fully connected layer we use to predict the plausibility. We use the cross-entropy as the loss function and stochastic gradient descent as the optimization method.

Model Performance

We train the classifier for different relations separately as some relations are not exclusive with each other (e.g., "RecievesAction" and "UsedFor") and test our model on our collected data. We compare our model with random guess and directly using BERT, which is identical to our model excluding the graph attention part, in Table 5.1. Experimental results show that both the Transformer and the graph attention modules make significant contributions to the prediction quality of the extracted knowledge.

5.3 Intrinsic Evaluation

5.3.1 Evaluation Metrics

We evaluate different approaches with the following metrics.

Quantity: We first measure quantity of the algorithms. Specifically, we are interested in two metrics: the number of acquired commonsense knowledge tuples and the number of unique words.

Novelty: For measuring novelty of the outputs, we follow [11] and report the proportion of novel tuples and concepts, denoted as Novel_t and Novel_c, respectively. Here "novel," similar to the definition in [11], means that one cannot find the whole tuple or concept in the training/development data via string match. For COMET and LAMA, as the head concept is given as input and thus only the tail concept is generated by the models, we only select the tail concepts to calculate the concept novelty.

Quality: We use Amazon Mechanical Turk to evaluate the quality of the acquired commonsense tuples. For each model, we randomly select 100 tuples from the overall set to test the quality. For each tuple, five workers are invited to annotate whether this tuple is plausible or not. If at least four annotators label the tuple as plausible, we then consider it as a plausible tuple. Additionally, to investigate the quality of tuples with novel concepts, for each model, we also report the performance of all sampled novel tuples. As we use the accuracy (# valid tuples/ # all tuples) to evaluate the quality, we denote the overall quality of tuples with novel concepts and the whole tuple set as ACC_n and ACC_o, respectively.

5.3.2 Baseline Methods

We compare TransOMCS with recently developed commonsense knowledge acquisition methods COMET [11] and LAMA [115], which aim at extracting commonsense knowledge from pre-trained language models.

1. **COMET:** Proposed by [11], COMET leverages the pre-trained language model GPT [120] to learn from annotated resources to generate commonsense knowledge.
2. **LAMA:** Proposed by [115], LAMA leverages BERT [25] to acquire commonsense knowledge. Unlike COMET, LAMA is unsupervised.

We denote our method as **TransOMCS**, indicating that we transfer knowledge from linguistic patterns to OMCS-style commonsense knowledge.

5.3.3 Implementation Details

For both COMET and LAMA, we include two variants in our experiments. The first one follows COMET's paper and uses concept/relation pairs among the most confident subset of OMCS as input. Due to the small size of this subset (only 1.2K positive examples), even if we use the 10-beam search decoding, we can only generate 12K tuples. To overcome this limitation and test whether these models can be used for generating large-scale commonsense knowledge, we consider a different evaluation setting where we randomly select 24K concepts from the concepts extracted by our approach and then randomly pair the selected concepts with commonsense relations as the input. We refer to the first and modified settings with $_{\text{Original}}$ and $_{\text{Extended}}$ subscripts, respectively.

5.3.4 Result Analysis

The summary of model evaluations is listed in Table 5.2. Compared with all baseline methods in their original settings, TransOMCS outperforms them in quantity. Even the smallest subset (top 1%) of TransOMCS outperforms their largest generation strategy (10-beam search) by ten times. TransOMCS also outperforms COMET in terms of novelty, especially the percentage of novel concepts. The reason behind this is that COMET is a pure machine-learning approach and it learns to generate the tail concept in the training set. It is possible that the stronger their models are, the

Model	# Vocab	# Tuple	Novel_t	Novel_c	ACC_n	ACC_o
$\text{COMET}_{\text{Original}}$ (Greedy decoding)	715	1,200	33.96%	5.27%	58%	90%
$\text{COMET}_{\text{Original}}$ (Beam search - 10 beams)	2,232	12,000	64.95%	27.15%	35%	44%
$\text{COMET}_{\text{Extended}}$ (Greedy decoding)	3,912	24,000	**99.98%**	55.56%	34%	47%
$\text{COMET}_{\text{Extended}}$ (Beam search - 10 beams)	8,108	240,000	**99.98%**	78.59%	23%	27%
$\text{LAMA}_{\text{Original}}$ (Top 1)	328	1,200	-	-	-	49%
$\text{LAMA}_{\text{Original}}$ (Top 10)	1,649	12,000	-	-	-	20%
$\text{LAMA}_{\text{Extended}}$ (Top 1)	1,443	24,000	-	-	-	29%
$\text{LAMA}_{\text{Extended}}$ (Top 10)	5,465	240,000	-	-	-	10%
$\text{TransOMCS}_{\text{Original}}$ (no ranking)	33,238	533,449	99.53%	89.20%	72%	74%
TransOMCS (Top 1%)	37,517	184,816	95.71%	75.65%	**86%**	87%
TransOMCS (Top 10%)	56,411	1,848,160	99.55%	92.17%	69%	74%
TransOMCS (Top 30%)	68,438	5,544,482	99.83%	95.22%	67%	69%
TransOMCS (Top 50%)	83,823	9,240,803	99.89%	96.32%	60%	62%
TransOMCS (no ranking)	**100,659**	**18,481,607**	99.94%	**98.30%**	54%	56%
OMCS in ConceptNet 5.0	36,954	207,427	-	-	-	**92%**

Table 5.2: Main Results. For our proposed method, we present both the full TransOMCS and several subsets based on the plausibility ranking scores. $\text{TransOMCS}_{\text{Original}}$ means the subset, whose head/relation appear in the test set as used by $\text{COMET}_{\text{Original}}$ and $\text{LAMA}_{\text{Original}}$. As LAMA is unsupervised, the Novelty metric is not applicable. For our model, for the fair comparison, we exclude all annotated knowledge.

more likely they overfit the training data, the fewer novel concepts are generated. As for the quality, when the training data is similar to the test data, COMET provides the best quality. For example, in the original setting, the greedy decoding strategy achieves 90% overall accuracy. As a comparison, the quality of LAMA is less satisfying, which is mainly because LAMA is fully unsupervised. Besides that, in the extended setting, the quality of both models drops, which is mainly because the randomly generated pairs could include more rare words. Compared with them, TransOMCS (top 1%) provides the comparable quality with COMET, but with a larger scale. When the quantity is comparable, TransOMCS outperforms both of them in terms of quality. We summarize the comparisons in Table 5.3.

5.3.5 Case Study

We show case studies in Figure 5.4 to further analyze different acquisition methods.

COMET: COMET is the only one that can generate long concepts. At the same time, it also suffers

	Annotation	Scale	Novelty	Quality
OMCS	full	small	high	high
COMET	supervision	large	low	depends
LAMA	no	large	high	low
TransOMCS	supervision	large	high	high

Table 5.3: Comparison of different commonsense resources or acquisition methods. The quality of COMET depends on the scale of its generated knowledge (high quality on the test set, but low quality on the large scale generation.) COMET and TransOMCS require a small number of annotations as supervision.

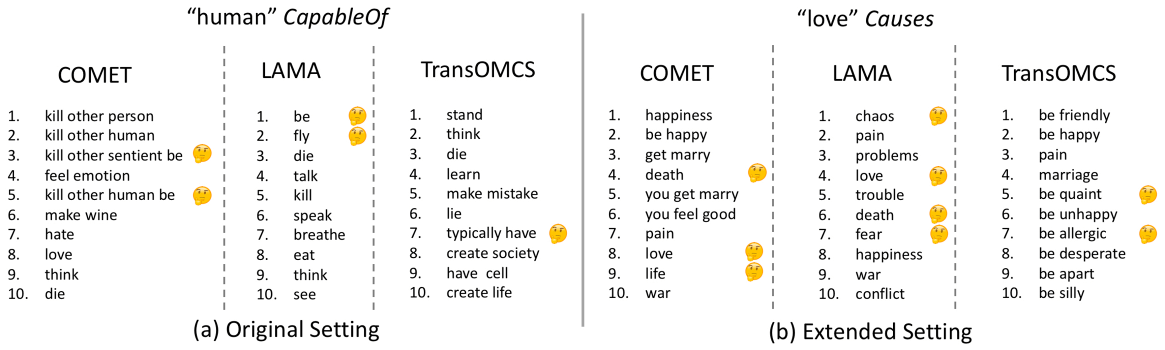

"human" *CapableOf*
COMET
1. kill other person
2. kill other human
3. kill other sentient be
4. feel emotion
5. kill other human be
6. make wine
7. hate
8. love
9. think
10. die
LAMA
1. be
2. fly
3. die
4. talk
5. kill
6. speak
7. breathe
8. eat
9. think
10. see
TransOMCS
1. stand
2. think
3. die
4. learn
5. make mistake
6. lie
7. typically have
8. create society
9. have cell
10. create life
(a) Original Setting
"love" *Causes*
COMET
1. happiness
2. be happy
3. get marry
4. death
5. you get marry
6. you feel good
7. pain
8. love
9. life
10. war
LAMA
1. chaos
2. pain
3. problems
4. love
5. trouble
6. death
7. fear
8. happiness
9. war
10. conflict
TransOMCS
1. be friendly
2. be happy
3. pain
4. marriage
5. be quaint
6. be unhappy
7. be allergic
8. be desperate
9. be apart
10. be silly
(b) Extended Setting

Figure 5.4: Case study of three knowledge acquisition methods. Two head-relation pairs are selected from the original and extended settings respectively. We indicate low quality tuples with confusing face emojis. For each model, we select the top ten most confident results.

from generating meaningless words. For example, two of the generated concepts end with "be," which is confusing and meaningless. This is probably because COMET only generates lemmas rather than normal words. Besides that, COMET could overfit the training data, even though the ten outputs are not exactly the same, four of them mean the same thing ("kill others").

LAMA: The most significant advantage of LAMA is that it is unsupervised. However, it has two major drawbacks: (1) it can only generate one-token concepts, which is far away from enough for commonsense knowledge; (2) the quality of LAMA is not as good as the other two methods.

TransOMCS: Compared with COMET, TransOMCS can generate more novel commonsense knowledge. For example, our model knows that "human" is capable of having cells and creating life. Besides that, unlike LAMA, TransOMCS can generate multi-token concepts. At the same time, our approach also has two limitations: (1) it cannot extract long concepts, which are difficult to find an exact pattern match; (2) as the extraction process strictly follows the pattern matching, it could extract semantic incomplete knowledge. For example, "human" is capable of "have." The

Commonsense Knowledge Resource	Reading Comprehension		Dialog Generation	
	Accuracy (%)	Δ (%)	BLEU	Δ
Base model (no external knowledge resource)	82.90	-	0.54	-
+OMCS	83.11	+0.21	0.72	+0.18
+COMET$_{\mathtt{Original}}$ (Greedy decoding)	83.12	+0.22	0.61	+0.07
+COMET$_{\mathtt{Extended}}$ (Beam search - 10 beams)	83.03	+0.13	0.68	+0.14
+LAMA$_{\mathtt{Original}}$ (Top 1)	83.13	+0.23	0.56	+0.02
+LAMA$_{\mathtt{Extended}}$ (Top 10)	83.17	+0.27	0.57	+0.03
+TransOMCS (Top 1%)	**83.27**	**+0.37**	**1.85**	**+1.31**

Table 5.4: Experimental results on downstream tasks. For COMET and LAMA, we report the performance of their most accurate and largest setting. For TransOMCS, as current models cannot handle large-scale data, we only report the performance of the most confident 1%. All the numbers are computed based on the average of four different random seeds rather than the best seed as reported in the original paper.

original linguistic graph should be "human have something", but as the pattern is "()<-nsubj<-()," the object is missing.

5.4 Extrinsic Evaluation

We conduct experiments on two downstream tasks: commonsense reading-comprehension [104] and dialogue generation [76]. We select [167] and [88, 185] as our base models for the comprehension and the generation task, respectively. For the fair comparison, we keep all the model architecture and parameters the same across different trials. We report the results with the common metric of each dataset: accuracy on the comprehension task and the BLEU score [108] on the dialog generation task.

The overall result is shown in Table 5.4. For the reading comprehension task, adding the top 1% of TransOMCS contributes 0.37 overall accuracy, compared to 0.21 contribution of OMCS. Meanwhile, the contributions of COMET and LAMA are minor for this task. For the dialogue generation task, TransOMCS shows remarkable improvement in the quality of generated responses. At the same time, adding other knowledge resources to OMCS does not provide any meaningful improvements to the performance. The reason behind this could be that COMET and LAMA provide limited high quality novel commonsense knowledge. For example, the original OMCS on average contributes 1.46 supporting tuples[6] and TransOMCS contribute another 3.36 supporting tuples. As a comparison, $\text{COMET}_{\mathtt{Original}}$, $\text{COMET}_{\mathtt{Extended}}$, $\text{LAMA}_{\mathtt{Original}}$, and $\text{LAMA}_{\mathtt{Extended}}$ only provide 0.01, 0.07, 0.49, 0.01 additional tuples respectively. This experiment result shows that TransOMCS has more novel knowledge.

[6]Here by supporting tuple, we mean that the head and tail concept appear in the post and response respectively.

PART 2

COMMONSENSE KNOWLEDGE ACQUISITION

A significant advantage of representing commonsense knowledge in a structured way is that we can use it as a bridge to connect different modalities. As a result, it would be easier for us to acquire knowledge (i.e., selectional preference) from different resources. In this part, we present three different approaches to acquire commonsense knowledge in the format of selectional preference. Specifically, in Chapter 6, we introduce how to encode words with multiplex word embeddings such that we can efficiently model different relations among words and use that to predict the selectional preference knowledge. After that, in Chapter 7, we investigate how we can generalize the knowledge about familiar eventualities to unfamiliar ones with the help of event analogy and conceptualization. In the end of this part, we discuss how to acquire causal relations between eventualities, which is a very important selectional preference relation, from visual signals in Chapter 8.

CHAPTER 6

SELECTIONAL PREFERENCE ACQUISITION WITH MULTIPLEX WORD EMBEDDINGS

Representing words as distributed representations (i.e., embeddings) is an important way for machines to process lexical semantics, which attracts much attention in natural language processing (NLP) in the past few years [93, 113, 149, 150, 148] with respect to its usefulness in many downstream tasks, such as parsing [17], machine translation [190], and coreference resolution [70]. Conventional word embeddings (e.g., word2vec [93] and GloVe [113]) leverage the co-occurrence information among words to train a unified embedding for each word. Such models are popular and the resulting embeddings are widely used owing to their effectiveness and simplicity. However, these embeddings are not helpful for scenarios requiring words functionalizing separately under different situations, where selectional preference (SP) [169] is a typical scenario.

Conventional SP acquisition methods are either based on counting [130] or complex neural network [24], and the SP knowledge acquired in either way can not be directly leveraged into downstream tasks. On the other hand, the information captured by word embeddings can be seamlessly used in downstream tasks, which makes embedding a potential solution for the aforementioned problem. However, conventional word embeddings using one unified embedding for each word are not able to distinguish different relations types (such as various syntactic relations, which is crucial for SP) among words. For example, such embeddings treat "food" and "eat" as highly relevant words but never distinguish the function of "food" to be a subject or an object to "eat." To address this problem, the dependency-based embedding model [73] is proposed to treat a word through separate ones, e.g., "food@dobj" and "food@nsubj," under different syntactic relations, with the skip-gram [93] model being used to train the final embeddings. However, this method is limited in two aspects. First, sparseness is introduced because each word is treated as two irrelevant ones (e.g., "food@dobj" and "food@nsubj"), so that the overall quality of learned embeddings is affected. Second, the resulting embedding size is too large,[1] which is not appropriate either for storage or usage.

[1]Assuming one has 200,000 words in the vocabulary and 20 dependency relations, and follow conventional ap-

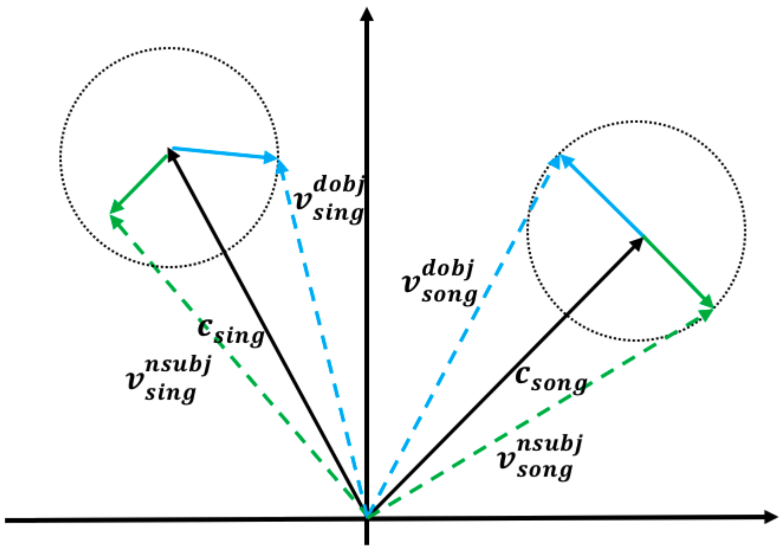


Figure 6.1: Illustration of the multiplex embeddings for "sing" and "song." Black arrows present center embeddings for words' overall semantics; blue and green arrows refer to words' relational embeddings for relation-dependent semantics. All relational embeddings for each word are designed to near its center embedding. nsubj and dobj relations are used as examples.

Therefore, in this chapter, we propose a multiplex word embedding (MWE) model, which can be easily extended to various relations between two words. A multiplex network embedding model was originally proposed for modeling multiple relations among people in a social network [181]. Interestingly, we found it also useful in capturing various relations among different words. An example is shown in Figure 6.1. "sing" and "song" are highly related to each other with the "predicate-objective" rather than the "predicate-subject" relation. In our model, each word has a group of embeddings, including a center embedding representing its general semantics, and several embeddings representing their relation-dependent semantics. To ensure that the embeddings of the same word (under different relations) are similar to each other, we limit the Euclidean norm of relation-dependent embeddings within a small range, as shown in Figure 6.1. Moreover, considering that there could be many relations among words, if using a conventional dimension setting for embeddings to encode relations, the overall embedding size would be too big to be used in downstream tasks. To deal with it, we propose to use a small dimension for relation-dependent embeddings and use a transformation matrix for each relation to project them into the same space of the center embeddings. Thus the two types of embeddings can be jointly trained and the quality of the relation-dependent ones is guaranteed.

proaches [93, 113] to set the embedding dimension to 300, the resulting embedding size will be about 10 Gigabytes.

Experiments are conducted on SP acquisition over different dependency relations to evaluate whether the learned embeddings effectively capture words' semantics over these relations. In addition, the word similarity measurement is used to assess how well words' general semantics are learned in our model. Both evaluations confirm the superiority of our model, where the SP information is effectively preserved and the words' overall semantics are enhanced. Particularly, further analysis also indicates that our MWE embeddings are more powerful than all existing embedding methods in SP acquisition with around 1/20 in their size comparing to previous embeddings [73].

6.1 Multiplex Word Embeddings

6.1.1 Model Overview

As aforementioned, encoding selectional preference information into embeddings can be conducted by modeling word association patterns under different dependency relations. Similar to [73], the proposed MWE model also distinguishes different relations among words and learns separate embeddings for them on each dependency edge.

Formally, let $\mathcal{W}$ be the vocabulary set containing n words $w_1, w_2, ..., w_n$ and $\mathcal{R}$ the relations set containing m relations $r_1, r_2, ..., r_m$, the proposed model is expected to produce $m + 1$ embeddings for each word, where there are m relational embeddings representing relations and a center embedding $\mathbf{c}$ for the general semantics. Particularly, each relational embedding has an overall and a local version, denoted as $\mathbf{v}$ and $\mathbf{u}$, where $\mathbf{u}$ records the difference between $\mathbf{v}$ and $\mathbf{c}$. For both the relational embeddings ($\mathbf{v}$ and $\mathbf{u}$) and the center embedding, we use a similar way as that in word2vec [93] that each one has two variants to represent the semantics of a word w being the head or tail in different relations. We denote the head and tail embeddings as $\mathbf{u}^r_{h,w} \in \mathbb{R}^s$ and $\mathbf{u}^r_{t,w} \in \mathbb{R}^s$ for the local embeddings of word w under relation type r and $\mathbf{c}_{w,h} \in \mathbb{R}^d$ and $\mathbf{c}_{w,t} \in \mathbb{R}^d$ for the center embedding, respectively, where s is the dimension for $\mathbf{u}$ and d the dimension for $\mathbf{c}$.

Although learning on similar information as that in [73], to reduce the sparseness of introducing m relational embeddings for each word, we do not treat each word as multiple separate prototypes. Instead, we use its center embedding to transfer information among different relations and the sum of the center embedding and a local embedding to represent the final embedding for the corresponding relation.[2] Moreover, considering that there are various relations among words, we use a lower

[2]Conceptually shown in Figure 6.1, to ensure the resulting final embeddings not too far away from the center embed-

dimension s for storage compression, with $s < d$. Thus, a transformation matrix $\mathbf{X}$ is introduced to transform $\mathbf{u}$ into the same vector space of $\mathbf{c}$. As a result, the final embeddings $\mathbf{v}$ for head and tail of w under relation r are formulated as:

$$\begin{aligned} \mathbf{v}^r_{h,w} &= \mathbf{c}_{h,w} + {\mathbf{X}^r_h}^\top \mathbf{u}^r_{h,w}, \\ \mathbf{v}^r_{t,w} &= \mathbf{c}_{t,w} + {\mathbf{X}^r_t}^\top \mathbf{u}^r_{t,w}. \end{aligned} \tag{6.1}$$

6.1.2 Learning the MWE model

To train $\mathbf{v}^r_{h,w}$ and $\mathbf{v}^r_{t,w}$ for each w under r, we adopt the negative sampling strategy to conduct the learning process. Specifically, for each r, we use a relation tuple set $\mathcal{T}_r$, with each tuple $t = (w_h, r, w_t) \in \mathcal{T}_r$, where w_h is the head word and w_t the tail word. For each t, we randomly generate two negative tuples by replacing w_h and w_t with the randomly selected fake head w'_h and tail w'_t respectively. Then the learning process is expected to distinguish the positive tuple against the negative ones. Therefore, formally, we maximize

$$\mathcal{J} = \frac{1}{|\mathcal{T}_r|} \sum_{t \in \mathcal{T}_r} \log \frac{e^{f(w_h,r,w_t)}}{e^{f(w_h,r,w_t)} + e^{f(w_h,r,w'_t)} + e^{f(w'_h,r,w_t)}}, \tag{6.2}$$

over all tuples in $\mathcal{T}_r$ for each r, with $f(\cdot)$ evaluating w_h and w_t being positive samples under r through

$$f(w_h, r, w_t) = {\mathbf{v}^r_{h,w_h}}^\top \cdot \mathbf{v}^r_{t,w_t}. \tag{6.3}$$

For each (w_h, w_t), we use cross entropy as the loss function

$$\begin{aligned} E = &- \log \sigma({\mathbf{v}^r_{h,w_h}}^\top \cdot \mathbf{v}^r_{t,w_t}) - \log \sigma(-{\mathbf{v}^r_{h,w_h}}^\top \cdot \mathbf{v}^r_{t,w'_t}) \\ &- \log \sigma(-{\mathbf{v}^r_{h,w'_h}}^\top \cdot \mathbf{v}^r_{t,w_t}), \end{aligned} \tag{6.4}$$

to measure the learning effect for Eq. (6.2), with σ denoting the sigmoid function.

Combined with Eq. (6.1), the training process is thus to update c, u^r, and X^r with the gradients passed from the loss function using stochastic gradient descent (SGD).

dings, we add restriction to the Euclidean norm of the local relational embeddings.

Algorithm 3 Multiplex Word Embedding

Input: Relation specific tuple sets $\mathcal{T}_1, \dots, \mathcal{T}_m$, d, s, a, and η.

Output: $\mathbf{c}_w$, $\mathbf{u}_w^r$, and $\mathbf{X}^r$.

```
1:  Initialize $\mathbf{c}_w$, $\mathbf{u}_w^i$ and $\mathbf{X}^i$ randomly.
2:  for Each Iteration k do
3:      Update $\lambda$.
4:      for Each tuple $t_p = (w_h, r, w_t) \in \mathcal{T}_r$ do
5:          Randomly generate the two negative examples $t_{n1} = (w'_h, r, w_t)$ and $t_{n2} = (w_h, r, w'_t)$.
6:          $\mathcal{T}_{tmp} = [t_p, t_{n1}, t_{n2}]$
7:          for $t \in \mathcal{T}_{tmp}$ do
8:              $e(w_h, w_t) = \sigma({\mathbf{v}'_{h,w_h}}^\top \cdot \mathbf{v}^i_{t,n_t}) - t_k$
9:              Update $\mathbf{c}_{h,w_h}$, $\mathbf{u}^r_{h,w_h}$, $\mathbf{X}^r_h$, $\mathbf{c}_{t,w_t}$, $\mathbf{u}^r_{t,w_t}$, and $\mathbf{X}^r_t$ based on Eqs. (6.5), 6.6), and (6.7).
10:             if $\|{\mathbf{X}^r_h}^\top \mathbf{u}^r_{h,w_h}\| > a$ then
11:                 Update $\mathbf{X}^r_h$ and $\mathbf{u}^r_{h,w_h}$.
12:             if $\|{\mathbf{X}^r_t}^\top \mathbf{u}^r_{t,w_t}\| > a$ then
13:                 Update $\mathbf{X}^r_t$ and $\mathbf{u}^r_{t,w_t}$.
```

In detail, take w_h as an example, its center embedding $\mathbf{c}_{h,w_h}$ is updated by

$$\begin{aligned}\mathbf{c}_{h,w_h} &= \mathbf{c}_{h,w_h} - \lambda \cdot \eta \cdot \frac{\partial E}{\partial \mathbf{v}^r_{h,w_h}} \\ &= \mathbf{c}_{h,w_h} - \lambda \cdot \eta \cdot e(w_h, w_t) \cdot \mathbf{v}^r_{t,w_t},\end{aligned} \tag{6.5}$$

where $e(w_h, w_t) = \sigma({\mathbf{e}^r_{h,w_h}}^\top \cdot \mathbf{e}^r_{t,w_t}) - t_k$, with $t_k = 1$ if (w_h, w_t) is positive sample and $t_k = 0$ for negative ones. η is the discounting learning rate. λ is an alternating weight to control the contribution of the gradient, ranging from 0 to 1.

Moreover, X^r and $\mathbf{u}^r$ are updated as follows:

$$\begin{aligned}\mathbf{u}^r_{h,w_h} &= \mathbf{u}^r_{h,w_h} - (1-\lambda) \cdot \eta \cdot \frac{\partial E}{\partial \mathbf{v}^r_{h,w_h}} \cdot \frac{\partial \mathbf{X}^r_h \mathbf{u}^r_{h,w_h}}{\partial \mathbf{u}^r_{h,w_h}} \\ &= \mathbf{u}^r_{h,w_h} - (1-\lambda) \cdot \eta \cdot e(w_h, w_t) \cdot \mathbf{X}^r_h \cdot \mathbf{v}^r_{t,w_t},\end{aligned} \tag{6.6}$$

$$\begin{aligned}\mathbf{X}^r_h &= \mathbf{X}^r_h - (1-\lambda) \cdot \eta \cdot \frac{\partial E}{\partial \mathbf{v}^r_{h,w_h}} \cdot \frac{\partial \mathbf{X}^r_h \mathbf{u}^r_{h,w_h}}{\partial \mathbf{X}^r_h} \\ &= \mathbf{X}^r_h - (1-\lambda) \cdot \eta \cdot e(w_h, w_t) \cdot \mathbf{u}^r_{h,w_h} \cdot \mathbf{v}^r_{t,w_t},\end{aligned} \tag{6.7}$$

Meanwhile, $\mathbf{c}_{t,w_t}$, $\mathbf{u}^r_{t,w_t}$, and $\mathbf{X}^r_t$ are updated in the same way of $\mathbf{c}_{h,w_h}$, $\mathbf{u}^r_{h,w_h}$, and $\mathbf{X}^r_h$ following Eqs. (6.5), (6.6), and (6.7).

The above learning process is conducted on all m relations at the same time. During the training, the Euclidean norm of $\mathbf{X}^{\mathsf{T}}\mathbf{u}$ for all embeddings is constrained to have a maximum semantic drifting range a, whose value controls the distance between the local relational embeddings and the center embedding. Specifically, if $\|\mathbf{X}^{\mathsf{T}}\mathbf{u}\|$ equals to a', which is greater than a, we scale down $\mathbf{X}^{\mathsf{T}}$ and $\mathbf{u}$ with $\sqrt{a' \cdot \frac{1}{ka}}$, where k is a scaling parameter set to 0.8 throughout this work.

The learning processing is summarized in Algorithm 3. For complexity analysis, it is obvious drawn that m relation types and n words result in $O(mn)$ and $O((d + s * m) * n)$ for time and space complexities, respectively.

6.1.3 The Alternating Optimization Strategy

To balance the learning process between the overall semantics and the relation-specific information, we adopt an alternating optimization strategy[3] [9] to adjust λ based on different stage of the training instead of using a fixed weight for λ. Specifically, considering $\mathbf{c}$ is more reliable and possesses more information while $\mathbf{u}$ is learned with shared $\mathbf{c}$, we alternatively update $\mathbf{c}$ and $\mathbf{u}$ upon the convergence of $\mathbf{c}$. As a result, we set λ to 1 in the first half of the training process and 0 afterwards.

6.2 Experiments

Experiments are conducted to evaluate how our embeddings are performed on SP acquisition and word similarity measurement.

6.2.1 Implementation Details

We use the English Wikipedia[4] as the training corpus. The Stanford parser[5] is used to obtain dependency relations among words. For the fair comparison, we follow existing work and set $d = 300$, $s = 10$, and $a = 1$. Following [62] and [24], we select three dependency relations (nsubj, dobj, and amod) as follows:

[3]Alternating optimization was originally proposed to train different parameters in a sequential manner and applied in various areas.

[4]https://dumps.wikipedia.org/enwiki/

[5]https://stanfordnlp.github.io/CoreNLP/

- nsubj: The preference of subject for a given verb. For example, it is plausible to say "dog barks" rather than "stone barks." The verb is viewed as the predicate (head) while the subject as the argument (tail).
- dobj: The preference of object for a given verb. For example, it is plausible for "eat food" rather than "eat house." The verb is viewed as the predicate (head) while the object as the argument (tail).
- amod: The preference of modifier for a given noun. For example, it is plausible to say "fresh air" rather than "solid air." The noun is viewed as the predicate (head) while the adjective as the argument (tail).

6.2.2 Baselines

We first compare the proposed multiplex embedding with the following embedding models. As it is trivial to apply these embedding models in downstream tasks, we label these models as downstream friendly.

- **word2vec**, the embedding model proposed by [93]. We use the skip-gram model for this baseline.
- **GloVe** [113], learning word embeddings by matrix decomposition on word co-occurrences.
- **D-embeddings**, the model proposed by [73] uses a skip-gram framework to encode dependencies into embeddings with multi-prototypes of words.

To investigate whether the SP knowledge can be captured by the pretrained contextualized word embedding models, we also treat following pre-trained models as baselines.

- **ELMo** [114], a pretrained language model with contextual awareness. We use its static representations of words as the word embedding.
- **BERT** [25], a pretrained bi-directional contextualized word embedding model with state-of-the-art performance on many NLP tasks.

Besides those embedding methods, we also compare with following conventional SP acquisition methods to demonstrate the effectiveness of the proposed multiplex embedding model, as it is still unclear how to leverage these methods in downstream tasks, we label these methods as downstream unfriendly.

- **Posterior Probability (PP)** [130], a counting based method for the selectional preference acquisition task.
- **Distributional Similarity (DS)** [36], a method that uses the similarity of the embedding of the target argument and average embedding of observed golden arguments in the corpus to predict the preference strength.
- **Neural Network (NN)** [24], an NN-based method for the SP acquisition task. This model achieves the state-of-the-art performance on the pseudo-disambiguation task.

For word2Vec and GloVe, we use their released code. For D-embedding, we follow their original paper using the Gensim package.[6] The dimensions of the aforementioned embeddings are set to 300 according to their original settings. For ELMo and BERT, we use their pre-trained models. As BERT was not originally designed for word level semantics tasks, for the selectional preference acquisition task, we compare with two variations of the original BERT model: (1) BERT (static), which extracts static embedding from the BERT model in a similar way as that from the ELMo model; (2) BERT (dynamic), which takes a pair of words as input and produces a plausibility score for that pairs of words. The main difference between the static and dynamic version of BERT is that in the dynamic version, the contextual information can be fully utilized, which is more similar to the training objective of BERT.

6.2.3 Selectional Preference Acquisition

The first task in our experiment is SP acquisition. Currently, there are two methods that we can use to evaluate the quality of extracted SP knowledge: Pseudo-disambiguation [132] and human labeled datasets [91, 62]. However, as shown in [182], pseudo-disambiguation selects positive SP tuples from the corpus and generate negative SP tuples by randomly replacing the head/tail. As a result, pseudo-disambiguation only evaluates how good the model fits the corpus rather than

[6]https://radimrehurek.com/gensim/models/word2vec.html

SP Evaluation Set	#W	#P
Keller [62]	571	360
SP-10K [182]	2,500	6,000

Table 6.1: Statistics of Human-labeled SP Evaluation Sets. #W and #P indicate the numbers of words and pairs, respectively. As different datasets have different SP relations, we only report statistics about "nsubj," "dobj," and "amod" (if available).

evaluating the SP acquisition models based on ground truth. Thus, in this section, we evaluate different SP acquisition methods with ground truth (human labeled datasets). Two representative datasets, Keller [62] and SP-10K [182], are selected as the benchmark datasets.

In each dataset, for each SP relation, various word pairs are provided. For each of the word pairs, the datasets also provide the annotated plausibility score of how likely a preference exists between that word pair under the corresponding SP relation. Thus the job for all the models is to predict the plausibility score for all word pairs under different SP relation settings. After that, we follow the conventional setting that uses the Spearman's correlation to assess the correspondence between the predicted plausibility scores and human annotations on all word pairs for all SP relations. The statistics of these datasets are shown in Table 6.1.

For embedding based methods (word2vec, GloVe, D-embedding[7], and MWE[8]), we follow previous work [93, 73] and use the cosine similarity between embeddings of head and tail words to predict their relations. For conventional SP acquisition methods (PP, DS, and NN), we follow their original paper to compute the plausibility scores.

The experimental results are shown in Table 6.2. As Keller is created based on the PP distribution and have relatively small size while SP-10K is created based on random sampling and has a much larger size, we treat the performance on SP-10K as the major evaluation. Our embeddings significantly outperform other baselines, especially embedding based baselines. The only exception is PP on the Keller dataset due to its biased distribution. In addition, there are other interesting observations. First, compared with "dobj" and "nsubj," "amod" is simpler for word2vec and GloVe. The reason behind is that conventional embeddings only capture the co-occurrence information,

[7]For D-embedding, since it provides embeddings for different relations (e.g., "food@nsubj" and "food@dobj"), we follow their work and directly use embeddings over certain relations to predict the score for the tested pairs. For example, given the test tuple ("eat," dobj, "food"), we compute the cosine similarity of "eat" and "food@dobj" rather than "food@subj."

[8]For the proposed model MWE, we use the cosine similarity between v_{h,w_h} and v_{t,w_t} as the predicted plausibility.

Model	Downstream	Keller			SP-10K			
		dobj	amod	average	nsubj	dobj	amod	average
word2vec	Friendly	0.29	0.28	0.29	0.32	0.53	0.62	0.49
GloVe	Friendly	0.37	0.32	0.35	0.57	0.60	0.68	0.62
D-embeddings	Friendly	0.19	0.22	0.21	0.66	0.71	0.77	0.71
ELMo	Friendly	0.23	0.06	0.15	0.09	0.29	0.38	0.25
BERT (static)	Friendly	0.11	0.05	0.08	0.25	0.32	0.27	0.28
BERT (dynamic)	Friendly	0.19	0.23	0.21	0.35	0.45	0.51	0.41
PP	Unfriendly	**0.66**	0.26	0.46	0.75	0.74	0.75	0.75
DS	Unfriendly	0.53	0.32	0.43	0.59	0.65	0.67	0.64
NN	Unfriendly	0.16	0.13	0.15	0.70	0.68	0.68	0.69
MWE	Friendly	0.63	**0.43**†	**0.53**†	**0.76**	**0.79**†	**0.78**	**0.78**†

Table 6.2: Results on different SP acquisition evaluation sets. As Keller is created based on the PP distribution and has relatively small size while SP-10K is created based on random sampling and has a much larger size, we treat the performance on SP-10K as the main evaluation metric. Spearman's correlation between predicated plausibility and annotations are reported. The best performing models are denoted with bold font. † indicates statistical significant (p <0.005) overall baseline methods.

which is enough to predict the selectional preference of "amod" rather than "nsubj" or "dobj."[9] Second, even though large-scale contextualized word embedding models like ELMo and BERT have been proved useful in many other tasks, they are still limited in learning specific and detailed semantics and thus perform inferior to our model in the SP acquisition task. For Example, ELMo and BERT can know that there is a strong semantic connection between "eat" and "food," but they do not know whether "food" is a plausible subject or object of "eat." Third, surprisingly, although NN achieves the state-of-the-art performance on the pseudo-disambiguation task, its performance is not satisfying against human annotation, especially on Keller, which is probably because the NN model overfits the training data, whose distribution is different from human SP knowledge.

6.2.4 Word Similarity Measurement

In addition to SP acquisition, we also evaluate our embeddings on word similarity (WS) measurement to test whether the learned embedding can effectively capture the overall semantics. We use SimLex-999 [52] as the evaluation dataset for this task because it contains different word types, i.e., 666 noun pairs, 222 verb pairs, and 111 adjective pairs. We follow the conventional setting

[9]The only possible SP relation between nouns and adjectives is "amod," while multiple SP relations could exist between nouns and verbs, and co_occurrence information cannot effectively distinguish them.

Model	WS	Dimension	Training Time
ELMo	0.434	512	≈40
BERT	0.486	768	≈300
MWE	0.476	300	4.17

Table 6.3: Comparison of MWE against language models on the WS task. Overall performance, embedding dimension, and training time (days) on a single GPU are reported.

that uses the Spearman's correlation to assess the correspondence between the similarity scores and human annotations on all word pairs. Evaluations are conducted on the final embeddings **v** for each relation and the center ones.

Results are reported in Table 6.4 with several observations. First, our model achieves the best overall performance and significantly better on nouns, which can be explained by that nouns appear in all three relations while most of the verbs and adjectives only appear in one or two relations. This result is promising since it is analyzed by [147] that two-thirds of the frequent words are nouns; thus there are potential benefits if our embeddings are used in downstream NLP tasks. Second, the center embeddings achieve the best performance against all the other relation-dependent embeddings, which demonstrates the effectiveness of our model in learning relation-dependent information over words and also enhancing their overall semantics.

We also compare MWE with pre-trained contextualized word embedding models in Table 6.3 for this task, with overall performance, embedding dimensions, and training times reported. It is observed that that MWE outperforms ELMo and achieves comparable results with BERT with smaller embedding dimension and much less training complexities.

6.3 Analysis

With the experiments on SP acquisition and word similarity measurement, we conduct further analysis for different settings for hyper-parameters and learning strategies, as well as the scalability of our model.

Model		noun	verb	adjective	overall
word2vec		0.41	0.28	0.44	0.38
Glove		0.40	0.22	0.53	0.37
D-embedding		0.41	0.27	0.38	0.36
nsubj	h	0.46	0.29	0.54	0.43
	t	0.45	0.25	0.48	0.40
	h+t	0.44	0.23	0.50	0.40
	[h,t]	0.47	0.27	0.51	0.42
dobj	h	0.46	0.27	0.45	0.41
	t	0.45	0.23	0.46	0.40
	h+t	0.45	0.20	0.45	0.38
	[h,t]	0.46	0.25	0.48	0.42
amod	h	0.47	0.25	0.52	0.37
	t	0.46	0.24	0.50	0.38
	h+t	0.46	0.24	0.52	0.38
	[h,t]	0.47	0.26	0.52	0.38
center	h	0.51	**0.33**	**0.57**	**0.48**
	t	0.51	0.30	0.56	0.47
	h+t	**0.52**	0.31	0.54	0.46
	[h,t]	0.51	0.32	0.57	**0.48**

Table 6.4: Spearman's correlation of different embeddings for the WS measurement. "nsubj," "dobj," "amod" represents the embeddings of the corresponding relation and "center" indicates the center embeddings. h, t, h+t, and [h,t] refer to the head, tail, sum of two embeddings, and the concatenation of them, respectively. The best scores are marked in bold fonts.

6.3.1 Effect of the Local Embedding Dimension

We investigating the performance of MWE on SPA (SP acquisition) and WS tasks with different s. As shown in Figure 6.2, the performance of our model is in an increasing trend when we increase the value of s. When the dimension reaches 10, the performance almost reaches the top, which confirms that the local relational embeddings can be effective in small dimensions (about 1/30 of the center dimension) when using center embeddings as the constraint.

6.3.2 Effect of the Semantic Drifting Range

Similar to s, we also investigate the model performance with the influence of different constraint α. It is observed in Figure 6.3 that, the constraint gets looser with the increase of α. For SP acquisition, in a small range, the model performance gradually improves along with the increasing value of

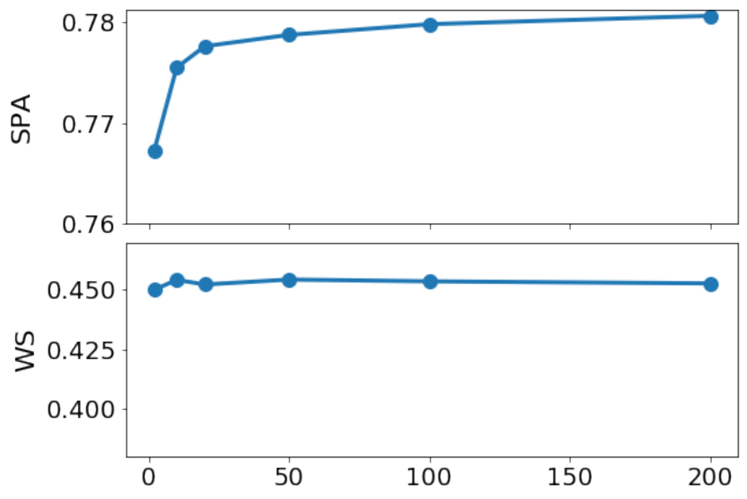


Figure 6.2: Effect of the different s on SP acquisition (SP-10K) and WS tasks.

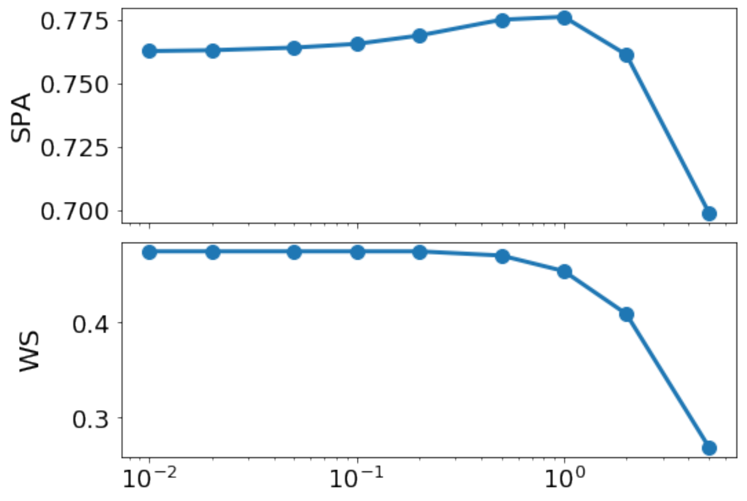


Figure 6.3: Effect of different α on SP acquisition (SP-10K) and WS tasks.

α because these embeddings are more flexible to capture intense relation-dependent information. However, once the range passes a threshold (1 in our experiment), the embeddings get farther away from their general semantics and start to overfit, which is also observed in the WS experiment, and thus the performance drops monotonically.

6.3.3 Effect of Alternating Optimization

An analysis is also conducted to demonstrate the effectiveness of the alternating optimization strategy. As shown in Table 6.5, we compare our model with several different strategies. The first one

Training Strategy	Averaged SPA	Overall WS
$\lambda = 1$	0.762	0.476
$\lambda = 0$	0.073	0.018
$\lambda = 0.5$	0.493	0.323
Alternating optimization	0.775	0.476

Table 6.5: Comparisons of different training strategies.

is to put all weights to the center embedding (fix λ to 1), which never updates the local relational embeddings. As a result, it can achieve similar performance on word similarity measurement but is inferior in SP acquisition because no relation-dependent information is preserved. The second strategy is to put all weights to the local relational embeddings (fix λ to 0). In this case, it performs very poor owing to the loss of general semantics. Last but not least, the alternating optimization also outperforms the setting with a fixed $\lambda = 0.5$, which can be explained by alternating optimization can first get a good overall semantic representation and then acquire the SP knowledge on top of that. To summarize, the alternating optimization provides an effective solution to training our model with a smart process in adjusting the contribution of different embeddings as well as stabilizing their optimization.

6.3.4 Scalability

Scalability of embeddings is an important factor in real applications. Practically, GPU or similar computation support is required to accommodate embeddings to applying neural models. Normally, the current most affordable GPUs typically have a memory size limitation around 10 GB. Thus, to ensure the learned embeddings can be used in downstream tasks, their size becomes an important concern when evaluating different embedding models, especially when there exist many different relations between words in our model.

Assuming that there are 20 relations among words, the size of all embedding models with the increasing word numbers in Figure 6.4. word2Vec and GloVe have the smallest size because they only train one embedding for each word while sacrificing the relation-dependent information among them. As a comparison, D-embeddings are trained on multiple prototypes for each word to record relation information, thus their size dramatically explodes with the increasing of the vocabulary. Consequently, it is easily computed for D-embeddings that 200,000 words will result

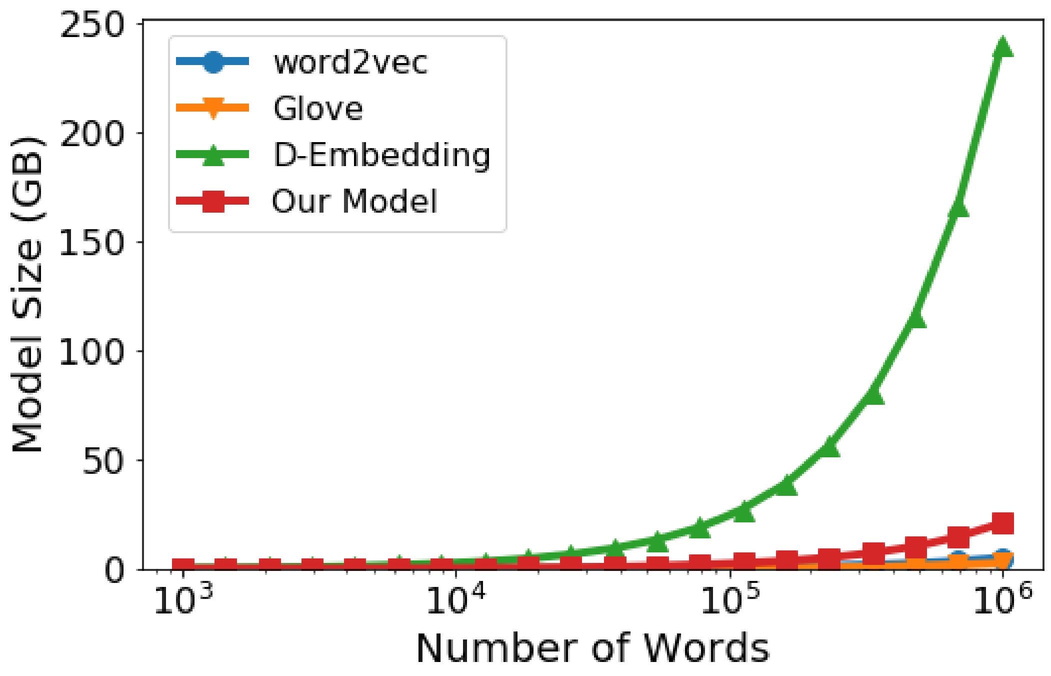


Figure 6.4: Scalability on embedding size against number of words.

in a 10GB model, which is unfeasible to be used in downstream tasks. Effectively with the small dimension for our local relational embeddings, relation information can be preserved in a small-sized model, which shows a compatible space requirement with the conventional embeddings.

Experiments on SP acquisition and word similarity measurement illustrated that our model better encodes SP knowledge than different baselines without harming its capability in representing general semantics. Analysis is also conducted with respect to different settings and optimization strategies, as well as the effect of model size, which further illustrates the validity and effectiveness of the proposed model and its learning process.

CHAPTER 7

COMMONSENSE GENERALIZATION WITH ANALOGY AND CONCEPTUALIZATION

In Chapter 4, we introduced a linguistic pattern-based approach that can effectively extract eventuality knowledge from the unlabeled corpus. However, through that way, we still cannot cover all the eventuality knowledge in the world because many eventualities may not be mentioned in all the literature we have at all. Such a sparsity problem brings a huge challenge for machines to understand those rare eventualities. However, humans would not suffer from this. For example, we can know that tigers can eat chicken without seeing it by ourselves or being told by others. An important reason behind this is that humans can effectively generalize their knowledge from familiar eventualities to unfamiliar but analogous ones with the help of conceptualization and conceptualization. For example, assume that we have not seen a tiger eats a chicken, but we know it can eat other small animals such as duck, and we know that both chicken and duck belongs to small animals, it is reasonable for us to guess that a tiger might be able to eat a chicken. Motivated by this, in this chapter, we investigate how we can mimic humans and generalize the knowledge about familiar eventualities to unfamiliar ones with the help of analogy and conceptualization. As compared with states, events are more complex because they are describing a change of states and typically involve multiple steps. We focus on events in this chapter.

Understanding events has long been a challenging task in NLP, to which many efforts have been devoted by the community. However, most existing works are focusing on procedural (or horizontal) event prediction tasks. Examples include predicting the next event given an observed event sequence [122] and identifying the effect of a biological process (i.e., a sequence of events) on involved entities [8]. These tasks mostly focus on predicting related events in a procedure based on their statistical correlations in previously observed text. As a result, understanding the meaning of an event might not be crucial for these horizontal tasks. For example, simply selecting the most frequently co-occurring event can offer acceptable performance on the event prediction task [44].

Computational and cognitive studies [140, 177] suggest that inducing and utilizing the hierar-

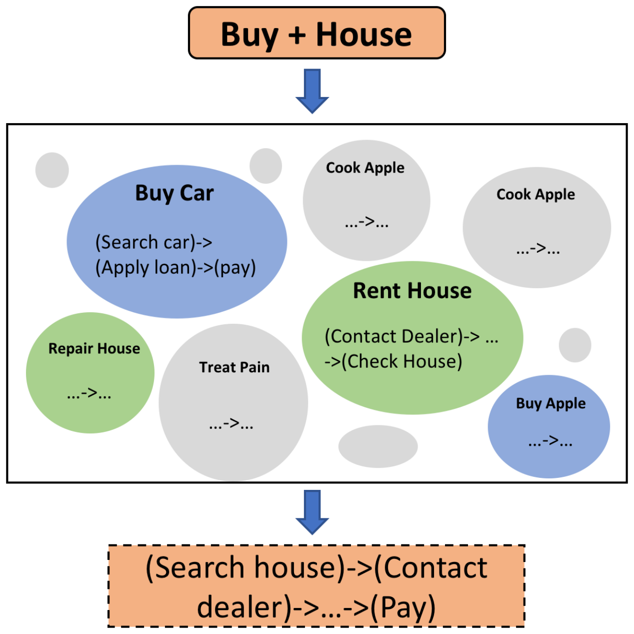


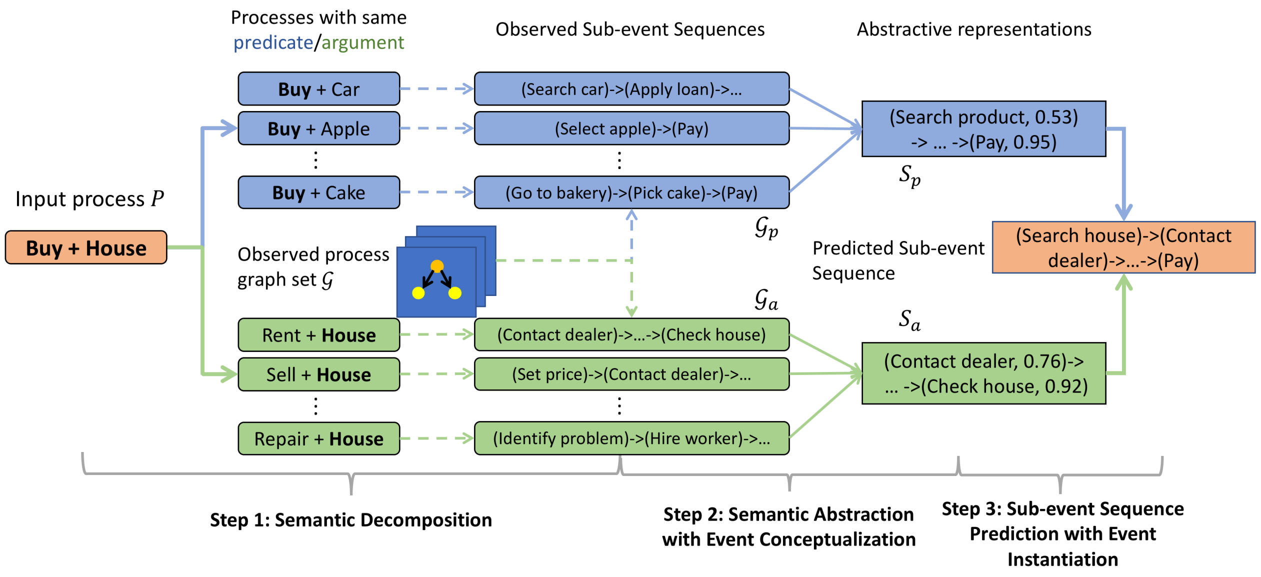
Figure 7.1: An illustration of leveraging known processes to predict the sub-event sequence of a new process.

chical structure[1] of events is a crucial component of how humans understand new events and can help many aforementioned horizontal event prediction tasks. Consider the example in Figure 7.1. Assume that one has never bought a house but is familiar with how to "buy a car" and "rent a house;" referring to analogous steps in these two relevant processes would still provide guidance for the target process of "buy a house." Motivated by this hypothesis, we try to directly evaluate a model's event understanding ability. We define this as the ability to identify vertical relations, that is, to predict the sub-event sequence of a new process.[2] We require models to generate the sub-event sequence for a previously unobserved process given observed processes along with their sub-event sequences, which we refer to as "the observed process graphs" in the rest of this chapter. This task is more challenging than "conventional" event predictions tasks, since it requires the generation of a sub-event sequence given a new, previously unobserved, process definition.

To address this problem, we propose an Analogous Process Structure Induction (APSI) framework. Given a new process definition (e.g., "buy a house"), we first decompose it into two dimensions: predicate and argument. For each of these, we collect a group of processes that share the same predicate (i.e., "buy-*ARG*") or same argument (i.e., "*PRE*-house"), and then induce an

[1]The original paper refers to the knowledge about processes and their sub-events as event schemata.

[2]A process is a more coarse-grained event by itself. In this chapter, we use this term to distinguish it from sub-events.



Processes with same predicate/argument
Observed Sub-event Sequences
Abstractive representations
Input process P
Buy + House
Buy + Car
Buy + Apple
Buy + Cake
(Search car)->(Apply loan)->...
(Select apple)->(Pay)
(Go to bakery)->(Pick cake)->(Pay)
(Search product, 0.53) -> ... ->(Pay, 0.95)
S_p
$\mathcal{G}_p$
Observed process graph set $\mathcal{G}$
Predicted Sub-event Sequence
(Search house)->(Contact dealer)->...->(Pay)
$\mathcal{G}_a$
S_a
Rent + House
Sell + House
Repair + House
(Contact dealer)->...->(Check house)
(Set price)->(Contact dealer)->...
(Identify problem)->(Hire worker)->...
(Contact dealer, 0.76)-> ... ->(Check house, 0.92)
Step 1: Semantic Decomposition
Step 2: Semantic Abstraction with Event Conceptualization
Step 3: Sub-event Sequence Prediction with Event Instantiation

Figure 7.2: Demonstration of the proposed APSI framework. Given a target process P, we first decompose its semantics into two dimensions (i.e., predicate and argument) by grouping processes that share a predicate or an argument. For each such group of processes, we then leverage the observed process graphs $\mathcal{G}$ to generate an abstract and probabilistic representation for their sub-event sequences. In the last step, we merge them with an instantiation module to produce the sub-event sequence of P.

abstract and probabilistic sub-event representation for each group. Our underlying assumption is that processes that share the same predicate or argument could be analogous to each other, and thus could share similar sub-event structures. Finally, we merge these two abstract representations, using an instantiation module, to predict the sub-event structure of the target process. By doing so, we only need a small number of analogous processes (as we show, 20, on average) to generate unseen sub-events for the target process. Intrinsic and extrinsic evaluations show that APSI outperforms all baseline methods and can generate meaningful sub-event sequences for unseen processes, which are proven to be helpful for predicting missing events.

The rest of the chapter is organized as follows. Section 7.1 introduces the Analogous Process structure induction (APSI) framework. Section 7.2 describes our intrinsic and extrinsic evaluation, demonstrating the effectiveness of APSI and the quality of the induced process knowledge.

7.1 The APSI Framework

Figure 7.2 illustrates the details of the proposed APSI framework. Given an unseen process P, a target sub-event sequence length k, and a set of observed process graphs $\mathcal{G}$, the task is to predict a

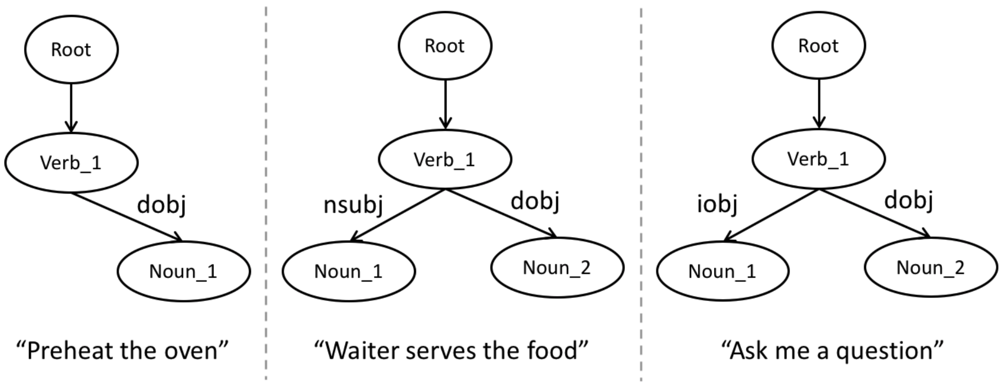


Figure 7.3: Examples of Sub-Event Representations.

k-step sub-event sequence $[E'_1, E'_2, ..., E'_k]$ for P. Each process graph $G \in \mathcal{G}$ in the input contains a process definition P^G and an n-step temporally ordered sub-event sequence $[E^G_1, E^G_2, ..., E^G_n]$. We assume that each process P is described as a combination of a predicate and an argument (e.g., "buy+house") and each sub-event $E \in \mathcal{E}$ is given as verb-centric dependency graph as used in [185] (see examples in Figure 7.3). In APSI, we decompose the target process into two dimensions (i.e., predicate and argument). For each target process, we collect a group of observed process graphs that share either the predicate or the argument with the target process; we assume that processes in these groups have sufficient information for predicting the structure of the target process. We then leverage an event conceptualization module to induce an abstract representation of each process group. Finally, we merge the two abstract, probabilistic representations and instantiate it to generate a ground sub-event sequence as the final prediction. Detailed descriptions of APSI components are introduced as follows.

7.1.1 Semantic Decomposition

Each process definition P is given as a predicate and its argument, which we term below the two "dimensions" of the process definition. We then collect all process graphs in $\mathcal{G}$ that have the same predicate as P into $\mathcal{G}_p$ and those that have the same argument into $\mathcal{G}_a$. We assume that these two sets provide the information needed to generate an abstract process representation that would guide the instantiation of the event steps for P.

7.1.2 Semantic Abstraction

The goal of the semantic abstraction step is to acquire abstract representations S_p and S_a for $\mathcal{G}_p$ and $\mathcal{G}_a$, respectively, to help transfer the knowledge from the grounded observed processes to the target new process. To do so, we first need to conceptualize observed sub-events in $\mathcal{G}_p$ and $\mathcal{G}_a$ (e.g., "eat an apple") to a more abstract level (e.g., "eat fruit"). Clearly, each event could be conceptualized to multiple abstract events. For example, "eat an apple" can be conceptualized to "eat fruit" but also to "eat food," and the challenge is to determine the appropriate level of abstraction. On one hand, the conceptualized event cannot be too general, as we do not want to lose touch with the original event, and, on the other hand, if it is too specific, we will not aggregate enough instances of sub-events into it, thus we will have difficulties transferring knowledge to the new unseen process. To automatically achieve the balance between these conflicting requirements and select the best abstract event for each observed sub-event, we model it as a weighted mutually exclusive set cover problem [86] and propose an efficient algorithm, described below, to solve it. We then merge the repeated conceptualized events and determine their relative positions.

Modeling Event Conceptualization

For each event E, we first identify all potential events that it can be conceptualized to. If two sub-events E_1 and E_2 can be conceptualized to the same event C, we place E_1 and E_2 into the set $\mathcal{E}_C$. To qualitatively guide the abstraction process we introduce below a notion of semantic loss that we incur as we move up to more abstract representations. To measure the semantic loss during the conceptualization, we assign weight to each set:

$$W(\mathcal{E}_C) = \frac{1}{\sum_{E \in \mathcal{E}_C} F(E, C)}, \tag{7.1}$$

where $F(E, C)$ is a scoring function, defined below in Eq. 7.2, that captures the amount of "semantic details" preserved due to abstracting from E to C. With this definition, the event conceptualization problem can be formalized as finding exclusive[3] sets (such as C) that cover all observed events with minimum total weight. In the rest of this section, we first introduce how to collect potential conceptualized events for each E, how we define F, and how we solve this discrete optimization problem.

[3]No sub-event can appear in two selected sets.

Identifying Potential Conceptualizations Assume that sub-event E contains m words $w_1^E, w_2^E, ..., w_m^E$, each corresponds to a node in Figure 7.3; for each of these, we can retrieve a list of hypernym paths from WordNet [94]. For example, given the word "house," WordNet returns two hypernym paths:[4] (1) "house"$\rightarrow$"building"$\rightarrow$"structure"$\rightarrow$...; (2) "house"$\rightarrow$"firm"$\rightarrow$"business"$\rightarrow$.... As a result, we can find $\prod_{w \in E} L(w)$ potential conceptualized events for E, where $L(w)$ is the number of w's hypernyms. We denote the potential conceptualized event set for E as $\mathcal{C}_E$ and the overall set as $\mathcal{C}$.

Conceptualization Scoring As mentioned above, for each pair of a sub-event E and its potential conceptualization C, we propose a scoring function $F(E, C)$ to measure how much "semantic information" is preserved after the conceptualization. Motivated by Budanitsky and Hirst [13] and based on the assumption that the more abstract the conceptualized event is, the more semantic details are lost, we define $F(E, C)$ to be:

$$F(E, C) = \prod_{i=1}^{m} w^{D(w_i^E, w_i^C)}, \tag{7.2}$$

where $D(w_i^E, w_i^C)$ is the depth from w_i^E to w_i^C on the taxonomy path, and w is a hyper-parameter[5] measuring how much "semantics" is preserved following each step of the conceptualization.

Conceptualization Assignment Now we are able to model the procedure of finding proper conceptualized events as a weighted mutually exclusive set cover problem. Note that this is an NP-complete problem and requires a prohibitive computational cost to obtain the optimum solution [59]. To obtain an efficient solution that is empirically sufficient for assigning conceptualized events with reasonable amount of instances, we develop a greedy procedure as described in Algorithm 4. For each retrieved process graph set $\mathcal{G}_p$ or $\mathcal{G}_a$, we collect all its sub-events as $\mathcal{E}$ and use it as the input for the conceptualization algorithm. In each iteration, we first compute the conceptualization score F for all the (E, C) pairs and then compute the weight score for all conceptualization sets $\mathcal{E}_C$. After selecting the set with minimum weight, $\mathcal{E}_{C_{min}}$, we remove all the events covered by it from $\mathcal{E}$ and repeat the process until no event is left. After the conceptualization, we merge sub-events that are conceptualized to the same event and represent them with the resulting conceptualized event C, whose weight is defined to be $\overline{W}(C) = \frac{1}{W(\mathcal{E}_C)}$. Compared with the naive algorithm, which first expands all possible subsets (i.e., it includes all subsets of $\mathcal{E}_C$ for all C) and then leverages the sort and filter technique to select the final subsets, we reduce the time complexity from $O(|\mathcal{C}| \cdot |\mathcal{E}|^2)$ to

[4]We omit the synset number for clear representation.

[5]In practice, we use two separate hyper-parameters w_v and w_n for verbs and nouns, respectively.

Algorithm 4 Event Conceptualization

INPUT: Set of events $\mathcal{E}$. Each $\mathbf{E} \in \mathcal{E}$ is associated with a set of potential conceptualization events $\mathcal{C}_E$. The overall conceptualized event set $\mathcal{C}$.

OUTPUT: Partition of n event subsets $\mathcal{P} = \{\mathcal{E}_1, \mathcal{E}_2, ..., \mathcal{E}_n\}$, where each subset $\mathcal{E}_i$ corresponds to a unique conceptualized event C_i.

```
1:  Initialize event partition set $\mathcal{P} := \emptyset$.
2:  while $\mathcal{E} \neq \emptyset$ do
3:      for Each $E \in \mathcal{E}$ do
4:          for Each $C \in \mathcal{C}_E$ do
5:              $\mathcal{E}_C := \emptyset$.
6:              Compute $F(E, C)$ using Eq. (7.2).
7:      for Each $C \in \mathcal{C}$ do
8:          for Each $E \in \mathcal{E}$ do
9:              if $C \in \mathcal{C}_E$ then
10:                 $\mathcal{E}_C := \mathcal{E}_C \cup \{E\}$.
11:         Compute $W(\mathcal{E}_C)$ using Eq. (7.1).
12:     Select $\mathcal{E}_{C_{min}}$ with the minimum $W$ score.
13:     $\mathcal{E} := \mathcal{E} \setminus \mathcal{E}_{C_{min}}$
14:     $\mathcal{P} := \mathcal{P} \cup \{\mathcal{E}_{C_{min}}\}$.
15: Return $\mathcal{P}$.
```

$O(n \cdot |\mathcal{C}| \cdot |\mathcal{E}|)$, where n is the number of conceptualized events and is typically much smaller than $|\mathcal{E}|$.

Conceptualized Event Ordering

After conceptualizing and merging all sub-events, we need to determine their loosely temporal order (e.g., whether they typically appear at the beginning or the end of these sub-event sequences). Let the set of selected conceptualized events be $\mathcal{C}^*$. For each $C \in \mathcal{C}^*$, we define its order score $T(C)$, indicating how likely C is to appear first, as:

$$T(C) = \sum_{C' \in \mathbf{C}^*} \theta\Big(\sum_{E_C \in \mathcal{E}_C} \sum_{E_{C'} \in \mathcal{E}_{C'}} t(E_C, E_{C'}) - t(E_{C'}, E_C)\Big), \tag{7.3}$$

where θ is the unit step function and $t(E_C, E_{C'})$ represents how many times E_C appears before $E_{C'}$ in an observed process graph.

7.1.3 Sub-event Sequence Prediction

In the last step, we leverage the two abstract representations we got for the predicate and argument of the target process definition to predict its final sub-events. To do so, we propose the following instantiation procedure. We are given the abstract representations S_p and S_a, for the predicate and argument, respectively. Each is a set of conceptualized events associated with weights and order scores. For each conceptualized event $C_p \in S_p$, using each event $C_a \in S_a$, we can generate a new instantiated event $\hat{C_p}$. For example, if C_p is "cut fruit" and C_a is "buy an apple," then our model would create the new event "cut an apple." Specifically, for each $w \in C_p$, if we can find a word $\hat{w}$ such that $\hat{w}$ is a hyponym of w, we will replace w with $\hat{w}$ and repeat this process until no hyponym can be detected in C_p. We denote the generated event by $\hat{C_p}$. To account for the semantic loss during the instantiation procedure, we define the weight and order score of $\hat{C_p}$ as follows:

$$\hat{W}(\hat{C_p}) = \overline{W}(C_p) \cdot F(\hat{C_p}, C_p) \cdot \frac{\sum_{C'_a \in S_a} \overline{W}(C'_a)}{\overline{W}(C_a)} \tag{7.4}$$

$$\hat{T}(\hat{E_p}) = T(C_p) \cdot F(\hat{C_p}, C_p) \cdot \frac{\sum_{C'_a \in S_a} \overline{W}(C'_a)}{\overline{W}(C_a)}, \tag{7.5}$$

Similarly, we apply the same procedure to C_a with C_p, and denote the resulted event $\hat{C_a}$. We then repeatedly merge instantiated events by summing up their weights and averaging their order scores. In the end, we select top k sub-events based on the weights and sort them based on the order score as the sub-event sequence prediction.

7.2 Experiments

In this section, we conduct intrinsic and extrinsic evaluations to show that APSI can generate meaningful sub-event sequences for unseen processes, which can help predict the missing events.

7.2.1 Dataset

We collect process graphs from the WikiHow website.[6] [66] In WikiHow, each process is associated with a sequence of temporally ordered human-created steps. For each step, as shown in Figure 7.3, we use the tool released by ASER [185] to extract events and construct the process graphs. We select all processes, where each step has one and only one event, and randomly split them into the train and test data. As a result, we got 13,501 training process graphs and 1,316 test process graphs,[7] whose average sub-event sequence length is 3.56.

[6]https://www.wikihow.com.

[7]We do not need a development set because the proposed solution APSI is not a learning-based method.

7.2.2 Baseline Methods

We compare with the following baseline methods:

Sequence to sequence (Seq2seq): One intuitive solution to the sub-event sequence prediction task would be modeling it as a sequence to sequence problem, where the process is treated as the input and the sub-event sequence the output. Here we adopt the standard GRU-based encoder-decoder framework [158] as the base framework and change the generation unit from words to events. For each process or sub-event, we leverage pre-trained word embeddings (i.e., GloVe-6b-300d [113]) or language models (i.e., RoBERTa-base [85]) as the representation, which are denoted as Seq2seq (GloVe) and Seq2seq (RoBERTa).

Top One Similar Process: Another baseline is the "top one similar process." For each new process, we can always find the most similar observed process. Then we can use the sub-event sequence of the observed process as the prediction. We employ different methods (i.e., token-level Jaccard coefficient or cosine similarity of GloVe/RoBERTa process representations) to measure the process similarity. We denote them as Top one similar process (Jaccard), (GloVe), and (RoBERTa), respectively.

For each process, we also present a randomly generated sequence and a human-generated sequence[8] as the lower-bound and upper-bound for sub-event sequence prediction models.

7.2.3 Intrinsic Evaluation

We first present the intrinsic evaluation to show the quality of the predicted sub-event sequences of unseen processes. For each test process, we provide the process name and the sub-event sequence length[9] to evaluated systems and ask them to generate a fixed-length sub-event sequence.

Evaluation Metric

Motivated by the ROUGE score [81], we propose an event-based ROUGE (E-ROUGE) to evaluate the quality of the predicted sub-event sequence. Specifically, similar to ROUGE, which evaluates the generation quality based on N-gram token occurrence, we evaluate how much percentage of

[8]The human-generated sequence is randomly selected from the WikiHow and excluded during the evaluation.

[9]We select the majority length of all references.

the sub-event and time-ordered sub-event pairs in the induced sequence is covered by the human-provided references. We denote the evaluation over single event and event pairs as E-ROUGE1 and E-ROUGE2, respectively. We also provide two covering standards to better understand the prediction quality: (1) "String Match": all words in the predicted event/pairs must be the same as the referent event/pairs; (2) "Hypernym Allowed": the predicted and referent event must have the same dependency structure, and for the words on the same graph position, they should be the hypernym of or same as each other. For example, if the referent event is "eat apple" and the predicted event is "eat fruit", we still count it as a match. The "String Match" setting is stricter, but the "Hypernym Allowed" setting also has its unique value to help better understand if our system is predicting relevant sub-events.

Implementation Details

In terms of training, we set both w_v and w_n to be 0.5 for our model. For the seq2seq baselines, we set the learning rate to be 0.001 and train the models until they converge on the training data. All other hyper-parameters following the original paper. In terms of the evaluation, we also provide two settings. (1) Basic: we follow previous works [40] to predict and evaluate events based on verbs; (2) Advanced: we predict and evaluate events based on all words.

Result Analysis

We show the results in Table 7.1. In general, there is still a notable gap between current models' performance and human performance, but the proposed APSI framework can indeed generate sufficiently relevant sub-events. For example, if we only consider the verb. Even in the string match setting, 14.8% of the predicted event and 6.6% of the ordered event pairs are covered by the references, which is much better than the random guess and nearly half of the performance of human beings. If hypernym is allowed, 36% and 19% of the predicted event and event pairs are covered. Besides that, if we take all words in the event into consideration, the task becomes more challenging. Specifically, even human can only achieve 11.63 E-ROUGE1 and 5.59 E-ROUGE2, which suggests that low scores achieved by current models are probably due to the limitation of the current dataset (e.g., on average, we only have 1.7 references for each test process). If more references are provided, the performance of all models will also increase. In the rest of the intrinsic

(a) Basic Setting (for each sub-event, we only predict and evaluate the verb)

Model	String Match		Hypernym Allowed	
	E-ROUGE1	E-ROUGE2	E-ROUGE1	E-ROUGE2
Random	2.9165	0.4664	23.5873	8.1089
Seq2seq (GloVe)	5.0323	1.4965	27.8710	13.0946
Seq2seq (RoBERTa)	4.5455	0.4831	28.0032	12.8502
Top one similar process (Jaccard)	8.8589	5.1000	28.6548	14.6231
Top one similar process (GloVe)	9.8797	5.1452	29.4203	13.6001
Top one similar process (RoBERTa)	9.2599	4.7390	30.6599	15.8417
Analogous Process Structure Induction (APSI)	**14.8013**	**6.6045**	**36.1648**	**19.2418**
Human	29.0189	15.2542	50.4647	29.4423

(b) Advanced Setting (for each sub-event, we predict and evaluate all words)

Model	String Match		Hypernym Allowed	
	E-ROUGE1	E-ROUGE2	E-ROUGE1	E-ROUGE2
Random	0.0000	0.0000	0.5104	0.0903
Seq2seq (GloVe)	0.1935	0.0534	0.9677	0.1069
Seq2seq (RoBERTa)	0.4870	0.0000	1.7857	0.2899
Top one similar process (Jaccard)	0.6562	0.2257	2.4797	0.5867
Top one similar process (GloVe)	0.8750	0.2106	2.8801	0.7372
Top one similar process (RoBERTa)	0.9479	0.3009	3.2811	0.9929
Analogous Process Structure Induction (APSI)	**3.4988**	**0.4513**	**6.1611**	**1.1885**
Human	11.6351	5.5905	18.0034	8.2695

Table 7.1: Intrinsic evaluation results of the induced process structures. On average, we have 1.7 human-generated sub-event sequences as the references for each test process. Best performing models are marked with the bold font.

evaluation, we present more detailed analysis based on the advanced setting (string match) and a case study to help better understand the performance of APSI.

Effect of the Instantiation Module

A key step in our framework is how to leverage the two abstract representations to predict the final sub-event sequence. In APSI, we propose an instantiation module, which jointly leverages the two representations to generate detailed events. To show its effect, we compare it with two other options: (1) Simple Merge: Merge two representation and select the top k sub-events based on the

Model	E-ROUGE1	E-ROUGE2
Simple Merge	2.5884	0.4062
Normalized	2.2238	0.3611
APSI (Instantiation)	**3.4988**	**0.4513**

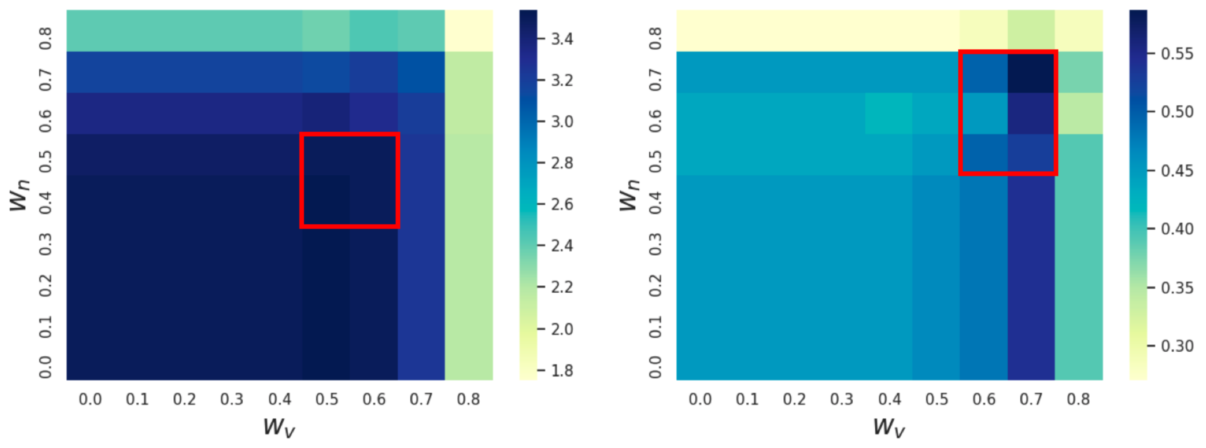
Table 7.2: Performance of different merging methods.

weight; (2) Normalized: First normalize the weight of all sub-events based on each representation and then select the top k sub-events.

From the result in Table 7.2, we can see that due to the imbalanced distribution of the two representations, simply choosing the most weighted sub-events is problematic. On average, for each predicate, we can collect 18.04 processes, while we can only collect 1.92 processes for each argument. As a result, the sub-events in the predicate representation typically have a larger weight. Thus if we simply merge them, most of the predicted sub-events will come from the predicate representation. Ideally, the "normalized" method can eliminate the influence of such imbalance, but it also amplifies the noise and achieves worse empirical performance. Differently, the proposed instantiation module uses events in one representation as the reference to help instantiate the events in the other one. As a result, we jointly use these two representations to generate a group of detailed events, and then we can select the top k generated new events. By doing so, we do not only go detailed from the abstract representation but also avoid the imbalanced distribution issue.

Hyper-parameter Analysis

In APSI, we use two hyper-parameters w_v and w_n to control the conceptualization and instantiation depth we want over verbs and nouns respectively. 0 means no conceptualization and the larger value indicates more conceptualization we encourage. We show the performance of APSI with different hyper-parameter combinations in Figure 7.4, from which we can see that a suitable level of conceptualization is the key to the success of APSI. If no conceptualization is allowed, all the predicted events are restricted to the observed sub-event, thus we cannot predict "search house" after seeing "search car" and some events about the house. On the other hand, if we do not restrict the depth of conceptualization, all the sub-events will be conceptualized to be too general. As a result, even with the instantiation module, we could not predict the detailed sub-event as we want.



(a) E-ROUGE1 (b) E-ROUGE2

Figure 7.4: Hyper-parameter influence on the quality of APSI generated sub-event sequences. For both w_v and w_n, 0 indicates no conceptualization and the larger the value, the deeper the conceptualization is. Best performing ranges are marked with red boxes, which indicate that the suitable conceptualization level is the key to APSI's success.

Process Name:	**Treat Pain**
References:	('learn cause'->'identify symptom'->'see doctor')
	('identify cause'->'learn injury'->'recognize symptom'->`recognize symptom')
APSI Prediction:	('Identify symptom'->'see doctor'->'recognize symptom'->'take supplement')

Figure 7.5: Case Study. We mark the covered and not covered predictions with green and red colors.

Case Study

Figure 7.5 shows an example that we use to analyze the current limitations of APSI. We can see that APSI can successfully predict events like "identify symptoms", but fails to predict event "identify causes". Instead, it predicts "take supplements." This is because APSI learns to predict such sequence from other processes like "treat diarrhea" or other diseases in the observed process graphs. Treating those diseases typically does not involve identifying the cause, which is not the case for treating pain. And, treating diseases often involves taking medicines, which can be conceptualized to "take supplement." As no events about pain helps instantiate "supplement," APSI just predicts it.

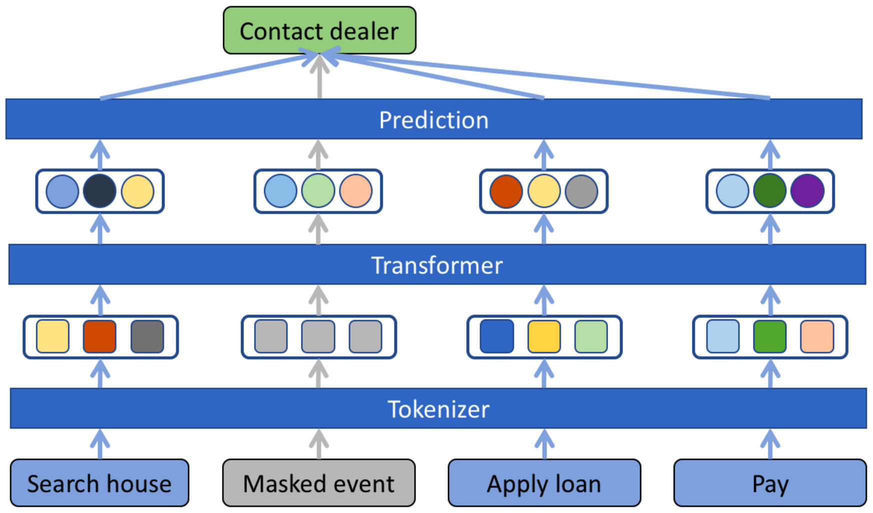


Figure 7.6: Demonstration of the event masked LM. Pre-trained language models are trained to predict the masked event given other events as the context.

7.2.4 Extrinsic Evaluation

As discussed by [134], the knowledge about process and sub-events can help understand event sequences. Thus, in this section, we investigate whether the induced process knowledge can help predict the missing events. Given a sub-event sequence, for each event in the sequence, we can use the rest of the sequence as the context and ask models to select the correct event against one negative event example. To make the task challenging, instead of random sampling, we follow Zellers et al. [180] to select similar but wrong negative candidates based on their representation (i.e., BERT [25]) similarity. We use the same training and test as the intrinsic experiment and as a result, we got 13,501 training sequences and 7,148 test questions.

The baseline method we are comparing with is the event-based masked language model,[10] whose demonstration is shown in figure 7.6. We use pre-trained RoBERTa-base [85] to initialize the tokenizer and transformer layer and all sequences of training processes as the training data. To show the value of understanding the relationship between process and their sub-event sequence, for each sub-event sequence in the test data, we first leverage the process name and different structure prediction methods to predict sub-event sequences and use them as additional context to help the event masked LM to predict the missing event. To show the effect upper bound of adding process knowledge, we also tried adding the process structure provided by human beings as the context,[11]

[10]On our dataset, the RoBERTa based event LM model outperforms existing LSTM-based event prediction models.

[11]We randomly select another sub-event sequence that describes the same process from WikiHow, which could be

Model	Accuracy	Δ
RoBERTa-based Event LM	73.59%	-
+ Seq2seq (GloVe)	73.06%	-0.53%
+ Seq2seq (RoBERTa)	72.33%	-1.26%
+ Top1 similar (Jaccard)	72.76%	-0.83%
+ Top1 similar (GloVe)	74.14%	0.55%
+ Top1 similar (RoBERTa)	74.16%	0.57%
+ APSI	74.78%†	1.19%
+ Human	76.97%‡	3.38%

Table 7.3: Results on the event prediction task. † and ‡ indicate the statistical significance over the baseline with p-value smaller than 0.01 and 0.001 respectively.

which is denoted as "+Human." All models are evaluated based on accuracy.

From the results in Table 7.3, we can make the following observations. First, adding high-quality process knowledge (i.e., APSI and Human) can significantly help the baseline model, which indicates that adding knowledge about the process can help better understand the event sequence. Second, the effect of process knowledge is positively correlated with their quality as shown in Table 7.1. Adding a low-quality process structure may hurt the performance of the baseline model due to the introduction of the extra noise. Third, the current way of using process knowledge is still very simple and there is room for better usage of the process knowledge, as the research focus of this work is predicting process structure rather than applying it, we leave that for the future work.

different from the currently tested sequence. As a result, adding such sequence cannot help predict all missing events.

CHAPTER 8

CAUSAL KNOWLEDGE ACQUISITION FROM TIME-CONSECUTIVE IMAGES

As aforementioned, a significant advantage of representing commonsense knowledge with a structured format rather than language models is that we have the potential to go beyond text and acquire knowledge from other modalities. In this chapter, we investigate how to acquire causal knowledge, which is difficult to be extracted from textual data due to its sparsity, from vision signals. As an important commonsense knowledge type, such causal knowledge has been shown to be helpful for many artificial intelligence tasks [103, 47, 101]. Thus, it is valuable to teach machines to understand causality [111].

Causal relations in the commonsense domain typically appear between daily eventualities (i.e., events and states) and are generally contributory and contextual [14]. By contributory,[1] we mean that the cause is neither necessary nor sufficient for the effect, but it strongly contributes to the effect. By contextual, we mean that some causal relations only make sense in a certain context. The contextual property of causal relations is important for both the acquisition and application of causal knowledge. For example, if some people tell the AI assistant (e.g., Siri) "they are hungry" in a meeting, a basic assistant may suggest that they order food because it knows that "being hungry" causes "eat food." A better assistant may recommend ordering food **after** the meeting because it knows that the causal relation between "being hungry" and "eat food" may not be plausible in the meeting context. Without understanding the contextual property of causal knowledge, achieving such a level of intelligence would be challenging.

To help machines better understand the causality commonsense, many efforts have been devoted to developing the causal knowledge bases. For example, ConceptNet [84] and ATOMIC [139] leverage human-annotation to acquire the causal knowledge. Even though the annotated causal knowledge bases are often of high quality, their effect on real applications is limited by the small

[1]The other two levels are absolute causality (the cause is necessary and sufficient for the effect) and conditional causality (the cause is necessary but not sufficient for the effect), which commonly appear in the scientific domain rather than our daily life.

Resource	Example
ConpetNet	“being hungry” causes “eat”
ATOMIC	“X repels Y’s attack” causes “X feels angry”
ASER	“I am tired” causes “I go to sleep”

Table 8.1: Causal-related triplets in ConceptNet [84], ATOMIC [139], and ASER. No context information is preserved.

scales. To address this issue, people try to leverage linguistic patterns (e.g., two events connected with “*Because*”) [51, 87] to acquire causal knowledge from textual corpus. However, causal knowledge is rarely formally expressed in textual data [84]. A pure text-based approach might struggle at covering all causal knowledge. Besides that, as shown in Table 8.1, none of them take the aforementioned contextual property of causal knowledge into consideration, which may restrict their usage in downstream tasks.

Thus, to address the limitation of previous text-only approaches, we propose to ground causal knowledge into the real world and explore the possibility of acquiring causal knowledge from visual signals (i.e., images in time sequence, which are cropped from videos). By doing so, we have three significant advantages: (1) Videos can be easily acquired and can cover rich commonsense knowledge that may not be mentioned in the textual corpus; (2) Events contained in videos are naturally ordered by time. As discussed by [101], there exists a strong correlation between temporal and causal relations, and thus such time-consecutive images can become a dense causal knowledge resource; (3) Objects from the visual signals can act as the context for detected causal knowledge, which can remedy the aforementioned “lack of contextual property” issue of existing approaches.

Specifically, we first define the task of mining causal knowledge from time-consecutive images and propose a high-quality dataset (Vis-Causal). To study the contextual property of causal relations, we provide two kinds of causality annotations for each pair of events: one is the causality given specific context, and the other one is the causality without context. Distribution analysis and case studies are conducted to analyze the contextual property of causality. An example from Vis-Causal is shown in Figure 8.1, where the causal relation between “dog is running” and “blowing leaves” only makes sense when the context is provided because the dog is running on the leaves, its high speed and quickly-moved pow cause the leaves to blow around. Without the context “leaves on the ground,” this causal relation is implausible. After that, we propose a Vision-Contextual Causal (VCC) model, which can effectively leverage both the pre-trained textual representation

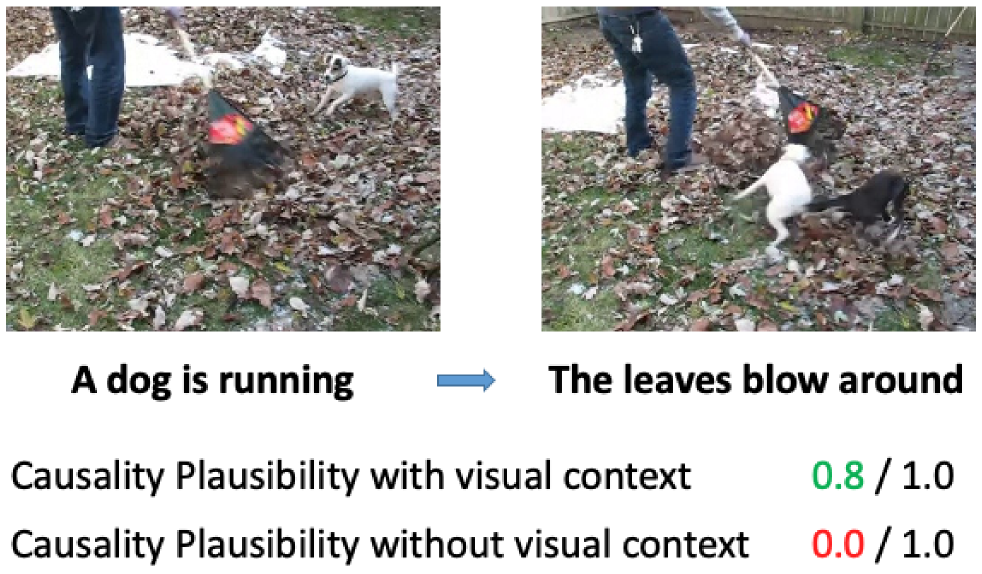

A dog is running
The leaves blow around
Causality Plausibility with visual context
0.8 / 1.0
Causality Plausibility without visual context
0.0 / 1.0

Figure 8.1: An example of Vis-Causal. Based on the annotated plausibility, there exists a strong causal relation from “A dog is running” to “The leaves blow around” with the visual signal as context (i.e., on leaves), but that causal relation is no longer plausible without the context.

and visual context to acquire causal knowledge and be used as a baseline method for future works. Experimental results demonstrate that even though the task is still challenging, by jointly leveraging the visual and contextual representation, the proposed model can better identify meaningful causal relations from time-consecutive images.

The rest of the chapter is organized as follows. In Section 8.1, we formally define the task of learning contextual causality from the visual signal. After that, we present the construction details about Vis-Causal in Section 8.2. In Section 8.3 and 8.4, we present the details about the proposed VCC model and the experiments.

8.1 The Task Definition

As introduced in the introduction, our ultimate goal is to acquire contextual causal knowledge from videos. However, as current models cannot afford to process videos directly, we simplify the task into mining causal knowledge from time-consecutive frames (i.e., images), which are cropped from the video. Thus, we formally define the task as follows. Each image pair $\mathrm{P} \in \mathcal{P}$, where $\mathcal{P}$ is the overall image pair set, consists of two images I_1 and I_2, sampled from the same video, in temporal order (i.e., I_1 appears before I_2). For each P, our goal is to identify all possible causal relations between the contained images. Normally, this task contains two sub-tasks: identifying events in images and identifying causality relation between contained events. As there exists a huge overlap between the event identification task and the scene graph generation task [173] in the computer vision (CV) community, which has been extensively studied [175, 77], in this work, we focus on the second sub-task. We assume that the event set contained in I_1 is denoted as $\mathcal{E}_1$, and the set of all events contained in all images sampled from V_1 is denoted as $\mathcal{E}_v$. For each event $e_1 \in \mathcal{E}_1$, our goal is finding all events $e_2 \in \mathcal{E}_v$ such that e_1 causes e_2.

8.2 The Vis-Causal Dataset

In this section, we introduce the details about the creation of Vis-Causal, which was carried out in four steps: (1) Pre-processing the raw video data into frames for further annotation; (2) Identifying the contained events from the frames; (3) First-round of causality annotation, which requires annotators to write down events that the given event can cause; (4) Second-round of causality annotation, which refines the quality of the annotation from step three via more fine-grained annotation and two settings are included (one is with the visual context and the other one is without any context). We select Amazon MTurk[2] as the annotation platform. We elaborate on each step as follows.

8.2.1 Data Source

To acquire a broad coverage of daily life causality, we choose to use ActivityNet [49], which contains short videos from YouTube, as the video resource. We randomly select 1,000 videos. We take five uniformly sampled screen-shots for each video and take adjoined screen-shots as pairs of time-consecutive images to capture the chained events better.[3] As a result, we collected 4,000 image pairs.

8.2.2 Event Identification

In the first step, we invite annotators to write down any events they can identify in the first image. Clear instructions and examples are given to help annotators understand the definition of events

[2]https://www.mturk.com/

[3]Ideally, we can select any two screen-shots to form the time-consecutive image pairs. However, as we could not afford to annotate all combinations, we only consider the adjoined ones at the current stage.

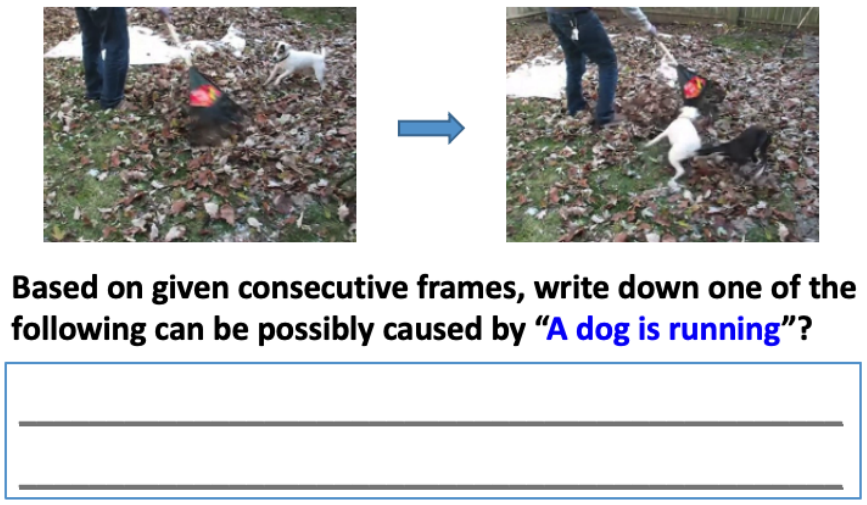

Based on given consecutive frames, write down one of the following can be possibly caused by "A dog is running"?

Figure 8.2: Candidate event pair Preparation.

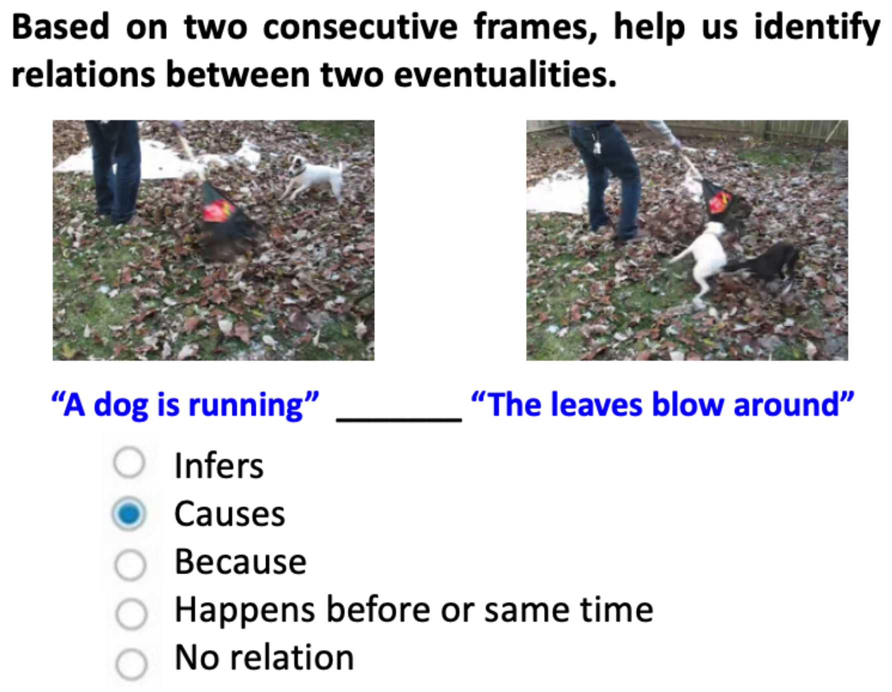
and our task. We invite three annotators for each image pair, resulting in 12,000 events in total for 4,000 image pairs.

8.2.3 First-round Causality Annotation

After identifying events, we invite annotators to identify related event pairs, which are used as candidates for next-step causal relation annotation. For each pair of time-consecutive images, we select all three identified events in the first image, and for each one of them, we ask annotators to describe one event that happens in the second image and is caused by the selected event, which occurs in the first image. If no suitable event is found, which is possible due to causal relations' sparseness, the annotators could choose "None" as the answer. A screenshot of this step is shown in Figure 8.2. For each question, we invite three different annotators to provide annotations. After filtering out answers that contain "None" or have less than two words, we obtain 23,558 event pair candidates. On average, we collect 5.89 candidate event pairs for each image pair.

8.2.4 Second-round Causality Annotation

As crowd-workers, unlike experts, are less likely to effectively distinguish the difference between inference and causality (For example, "A cat is running" infers "An animal is running" but "A cat is running" doesn't cause "an animal is running"), such relations are often mistakenly annotated as causal relation in the first-round annotation based on our observation. To remedy this problem,


Based on two consecutive frames, help us identify relations between two eventualities.

"A dog is running" _______ "The leaves blow around"

- Infers
- [✓ Causes]
- Because
- Happens before or same time
- No relation

Figure 8.3: Causal relation annotation. We only show the survey for causal relation annotation with context as an example. The only difference in the w/o context setting is that we remove the images. Distribution of plausibility scores under different settings and their difference.

in the second round of annotation, we first present a clear definition and several examples of all possible relations (e.g., "Inference" and "Causality") and then ask annotators to select the most plausible relations for all candidate event pairs, which is more fine-grained and thus achieves the better annotation quality. An example is shown in Figure 8.3. Besides that, to investigate the contextual property of causal knowledge, two settings are considered for the annotation (one with the context and one without). For each setting, we invite five annotators for annotating each event pair. Following previous works [128, 52], we employ Inter Annotator Agreement (IAA), which computes the average agreement of an annotator with the average of all other annotators, to evaluate the overall annotation quality. As a result, we achieve 78% and 76% IAA scores for "*with context*" and "*without context*" settings respectively. We achieve slightly lower agreement in the "without context" setting. One possible explanation is that when no context is provided, different people may think about different contexts, and thus their annotations about causal relations could be slightly different.

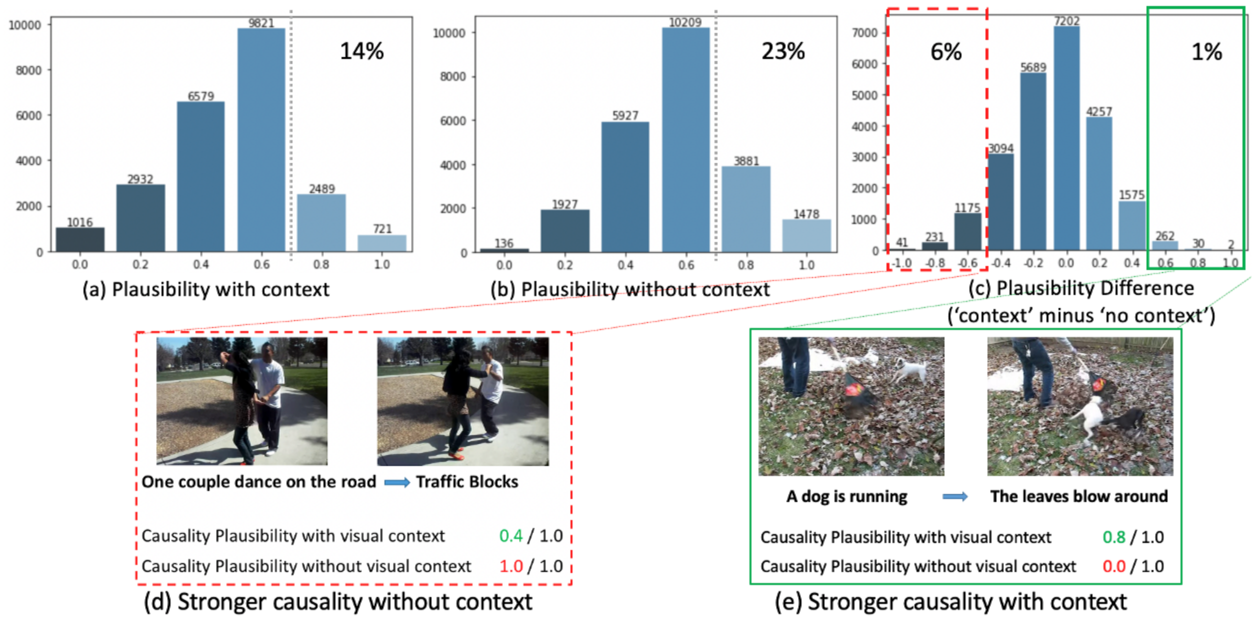


Figure 8.4: Distribution of plausibility scores under different settings and their difference.

8.2.5 Annotation Analysis

The distribution of annotation results for both settings are shown in Figure 8.4(a) and Figure 8.4(b) respectively. For each pair of events, we compute the plausibility based on voting. For example, if four out of five annotators vote "causal," its causal plausibility is 0.8. In general, we can see that the majority of the candidate events pairs have weak causal relations for both settings, and only a small portion of the candidates contain strong causal relations, especially for the "*with context*" setting. One possible explanation is that when no context is provided, humans can think about multiple contexts and find the most suitable one such that the causal relation is plausible in that scenario. However, when the visual context is provided, where the scenario is fixed, humans do not have the freedom to choose a suitable scenario by themselves. As a result, annotators tend to annotate more plausible causal relations in the "*no context*" setting.

To investigate the contextual property of causal relations, we show the distribution of plausibility difference ("*with context*" minus "*without context*") in Figure 8.4(c). From the result, we can observe that about 6% of event pairs, which is indicated with the dashed box, have stronger causal relations without any context, while about 1% of event pairs, which is indicated with the solid box, have a stronger causal relation when the visual context is provided. Two examples of both cases are shown in Figure 8.4(d) and 8.4(e) respectively.

	#Videos	#Images	#Pairs	#Positives	#Cand.
Train	800	4,000	3,200	2,599	31.8
Dev	100	500	400	329	32.1
Test	100	500	400	282	32.2

Table 8.2: Dataset statistics. **#Videos**, **#Images**, and **#Pairs** mean the number of videos, images, and image pairs, respectively. **#Positives** denotes the number of event pairs labeled with causality relation. **#Cand.** denotes the average length of the candidate event list.

8.2.6 Dataset Splitting and Statistics

We split the dataset into the train, dev, and test sets based on the original split of ActivityNet [49] and collect 800, 100, and 100 videos for the train, dev, and test set, respectively. We select positive causal relations based on the annotation under the "*with context*" setting. If at least four of five annotators think there exists a causal relation between a pair of events given the context, we will treat it as a positive example. As a result, we got 2,599, 329, and 282 positive causal pairs for the train, dev, and test set, respectively. On average, each event pair contains 11.41 words, and the total vocabulary size is 10,566. We summarize the detailed dataset statistics in Table 8.2.

8.3 The VCC Model

This section introduces the proposed Vision-Contextual Causal (VCC) Model, which leverages both the visual context and contextual representation of events to predict the causal relations. We show the overall framework in Figure 8.5. In total, we have three major components: event encoding, which encodes the two events into vectors for further prediction; visual context encoding, which encodes the context frame such that the context can be utilized in the model; and cross attention, which aims at finding the best context and event representation via the attention mechanism [164]. The details about these components are introduced as follows.

8.3.1 Textual Event Encoding

As both e_1 and e_2 are represented with natural language, we begin with converting them into vector representations. In this work, we leverage a pre-trained language representation model BERT [25] to encode all events. Assuming that after the tokenization, event e contains n tokens $w_1, w_2, ..., w_n$, we denote their contextualized representations after BERT as $\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_n$.

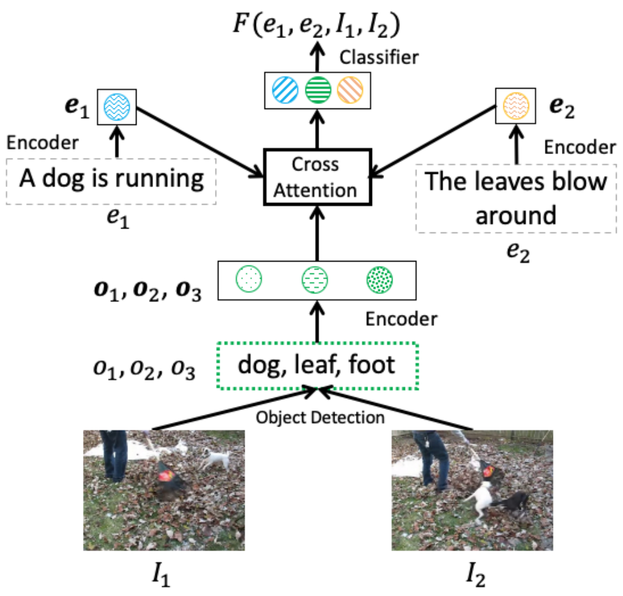


Figure 8.5: Demonstration of the proposed model. We use $\mathbf{e}_1$ and $\mathbf{e}_2$ to denote the vector representation of e_1 and e_2 after encoding respectively. Assuming that three objects (o_1, o_2, and o_3), whose vector representations are denoted as $\mathbf{o}_1$, $\mathbf{o}_2$, and $\mathbf{o}_3$ respectively, are extracted from the two context images, a cross attention module is proposed to jointly leverage the event and context representation to make the final prediction.

8.3.2 Visual Context Encoding

Following the standard approach in multi-modal approaches [173, 22], we first leverage an object detection module to detect objects from images and use all extracted objects to represent the visual context. Assuming that for I_1 and I_2, we extract m_1 and m_2 objects, respectively. After combining all objects from two images together and sorting them based on the confidence score provided by the object detection module, we keep the top m objects and denote them as $o_1, o_2, ..., o_m$. The motivation for that operation is to avoid the influence of noise introduced by the object detection module. As all objects are in the form of words, to align with events, we use the same pre-trained language representation model to extract the vector representation[4] of selected objects and denote them as $\mathbf{o}_1, \mathbf{o}_2, ..., \mathbf{o}_m \in \mathcal{O}$.

8.3.3 Cross-Attention Module

The cross-attention module aims to minimize the influence of noise by selecting important context objects with events and informative tokens in events with the context. Thus, the cross-attention

[4]If an object word is tokenized to multiple tokens, we take their average representation as the token representation.

module contains two sub-steps: (1) context representation; (2) event representation.

Context Representation: For each event e, whose tokens' vector representations are $\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_n$, we first take the average of all tokens and denote the resulted average vector as $\widetilde{\mathbf{w}}$. As the vector representation set of all selected objects is denoted as $\mathcal{O}$, we compute the overall context representation as:

$$\mathbf{o} = \sum_{\mathbf{o}' \in \mathcal{O}} a_{\widetilde{\mathbf{w}},\mathbf{o}'} \cdot \mathbf{o}', \tag{8.1}$$

where $a_{\widetilde{\mathbf{w}},\mathbf{o}'}$ is the attention weight of $\widetilde{\mathbf{w}}$ on object o'. Here we compute the attention weight as:

$$a_{\widetilde{\mathbf{w}},\mathbf{o}'} = NN_a([\widetilde{\mathbf{w}}, \mathbf{o}']), \tag{8.2}$$

where NN_a is a standard two-layer feed forward neural network and $[,]$ indicates the concatenation.

Event Representation: After getting the context representation, the next step is computing the event representation. Assuming that the vector set of e is $\mathcal{W}$, we can get the event representation with a similar attention structure:

$$\mathbf{e} = \sum_{\mathbf{w}' \in \mathcal{W}} b_{\mathbf{o},\mathbf{w}'} \cdot \mathbf{w}',$$

$$b_{\mathbf{o},\mathbf{w}'} = NN_b([\mathbf{o}, \mathbf{w}']),$$

where b is the attention weight we computed with another feed forward neural network NN_b.

8.3.4 Causality Prediction

Assuming that the context representations with e_1 and e_2 as attention signal are denoted as $\mathbf{o}_{e_1}$ and $\mathbf{o}_{e_2}$ respectively and the overall representations of e_1 and e_2 are $\mathbf{e}_1$ and $\mathbf{e}_2$, we can then predict the final causality score as follows:

$$F(e_1, e_2, I_1, I_2) = NN_c([\mathbf{e}_1, \mathbf{e}_2, \mathbf{o}_{e_1}, \mathbf{o}_{e_2}]). \tag{8.3}$$

8.4 The Experiment

This section presents experiments and analysis to show that both the pre-trained textual representation and visual context can help learn causal knowledge from time-consecutive images.

8.4.1 Evaluation Metric

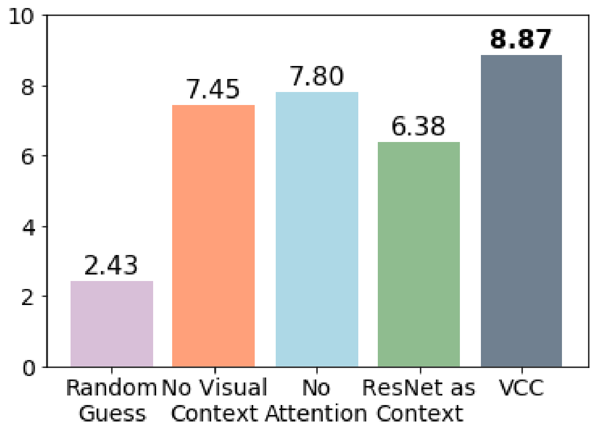
As each event in the first image could cause multiple events in the second image, following previous works [173], we evaluate different causality extraction models with a ranking-based evaluation metric. Given each event e in the first images, models are required to rank all candidate events based on how likely they think these events are caused by e. We then evaluate different models based on whether the correct caused event is covered by the top one, five, or ten ranked events. We denote these evaluation metrics as Recall@1, Recall@5, and Recall@10. In our experiment, to make the task more challenging, we only consider detected events in the same video as negative examples rather than randomly sampling the negative examples.

8.4.2 Baseline Methods

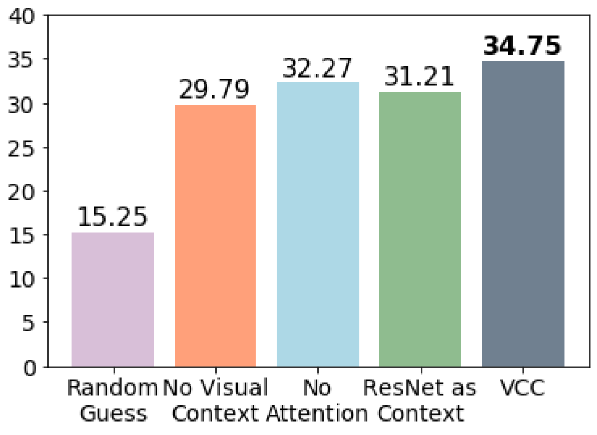
To prove that the visual context is crucial and the proposed cross-attention module is helpful, we compare VCC with the following models:

1. **No Visual Context**: Directly predicts the causal relation between events without considering the visual context. We take the average of word representations as the event representation and concatenate the representations of two events together for the final prediction for each event.
2. **No Attention**: Removes the cross-attention module and uses the average word embeddings of all selected objects to represent the context.
3. **ResNet as Context**: Removes the object detection module and uses the average image representation extracted by ResNet-152 [48] as the context representation. We acquire event representation as same as the "no context" setting and concatenate the representation of context and events together for the final prediction.

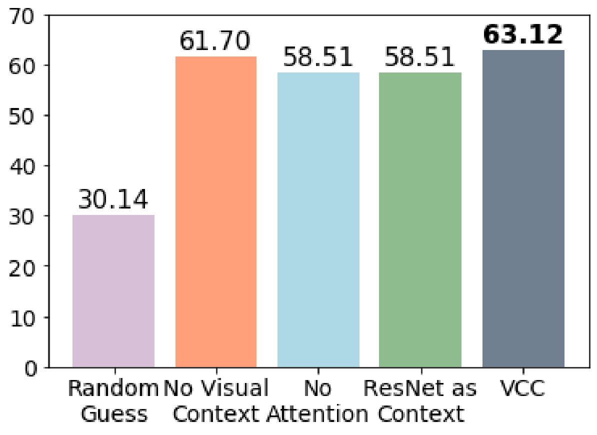
Besides the aforementioned baselines, we also present the performance of a "random guess" baseline, which randomly ranks all candidate events and can be used as a performance lower-bound for all causality extraction models.



(a) Recall@1



(b) Recall@5



(c) Recall@10

Figure 8.6: Experimental results on Recall@1, Recall@5, and Recall@10, respectively.

8.4.3 Implementation Details

Model Details: We use the pre-trained BERT model [25] as the textual representation model to encode both events and objects detected from images. Out-of-vocabulary words are initialized with zero vectors. We follow the previous scene graph generation work [173] to leverage a Faster R-CNN network [129],[5] which is trained on MS-COCO [82], to detect objects from the images. Experimental results on Recall@1, Recall@5, and Recall@10, respectively. We set the hidden state size in the feed-forward neural network to be 200 and the number of selected objects m to be 10. The total number of trainable parameters is 109.9 million (including 109.48 million from BERT-base).

[5]https://github.com/endernewton/tf-faster-rcnn

Training Details: During the training phase, for each positive example, we randomly select one negative example and use cross-entropy as the loss function. We employ stochastic gradient descent (SGD) as the optimizer. All parameters are initialized randomly, and the learning rate is set to be 10^{-4}. All models are trained with up to ten epochs,[6] and the models that perform best on the dev set are evaluated on the test set. The experiments are implemented on Intel(R) Xeon(R) CPU E5-2640v4@2.40GHz and one GTX-1080 GPU. Each training epoch takes 18 minutes on average.

8.4.4 Result Analysis

We report the performance of all models on all categories and show the results in Figure 8.6, from which we can make the following observations:

1. All models significantly outperform the "random guess" baseline, which shows that models can learn to extract meaningful causal knowledge from these time-consecutive images and learning causal knowledge from the visual signal can be a good supplement for the current text-based approach of acquiring causal knowledge in the future.

2. With the help of the context information, VCC outperforms the baseline "No Context" model in most experiment settings, proving the importance of visual context and is consistent with our previous observation that some causality only makes sense in certain contexts.

3. The proposed VCC model outperforms the "No Attention" model significantly, demonstrating the influence of noise introduced by the object detection module and the effectiveness of the proposed cross-attention module.

4. The proposed VCC model, which first extracts objects from images and then use extracted objects as the context, outperforms the "ResNet as Context" baseline, which directly uses the ResNet encoded image representation as the context, even though it does not suffer from the noise introduced by the object detection module at all. One possible explanation is that the event textual representation and the ResNet encoded image representation are vectors in different semantic spaces (one in language and one in vision), which may not perfectly align with each other. As a comparison, in the proposed model, we first extract objects from images and then encode them as text, and thus the alignment issue no longer exists.

[6] All models converge before ten epochs.

	R@1	R@5	R@10
Random Guess	2.13	15.25	30.14
BERT	2.13	22.34	39.00
GPT-2	3.55	17.73	34.40
VCC (BERT)	**8.87**	**34.75**	**63.12**
VCC (GPT-2)	7.80	31.56	56.03

Table 8.3: Performances of different models. Best-performing models are indicated with the **bold** font.

5. In general, even though the proposed model outperforms all baseline methods and can learn to extract meaningful causal knowledge from the training data, the task is still challenging due to the following reasons: (1) Missing information about the visual context: the performance of current object detection module is not good enough and many essential objects related to the two events could be missing; (2) Correct resolution of pronouns: pronouns frequently appear in events and without the correct resolution of those pronouns, it is hard to fully understand the semantic meaning of events; (3) Lack of support of external knowledge, especially commonsense knowledge.

8.4.5 Can Language Representation Models Understand Causality?

As observed in [116], language representation models can preserve rich knowledge, in this subsection, we conduct experiments to investigate whether pre-trained language representation models (i.e., BERT [25] and GPT-2 [121]) can understand causality without any training. For each candidate event pair (e.g., (“A dog is running”, “The leaves blow around”)), we convert it into a natural sentence (e.g., “A dog is running, so the leaves blow around”), and then input it into the language representation model. The overall probability returned by models can be used as their causality predictions. The higher probability indicates higher plausibility prediction. We rank all candidates by their probabilities and evaluate the models in the same way as previous experiments. We conduct experiments on BERT-base[7] and GPT-2 (774 million parameters). with the Hugging face implementation.[8] Besides unsupervised approaches, we also present the performances of replacing the language representation module in VCC with different pre-trained models.

[7]We also tried BERT large, but it does not make a significant improvement over the base model.

[8]https://github.com/huggingface/transformers

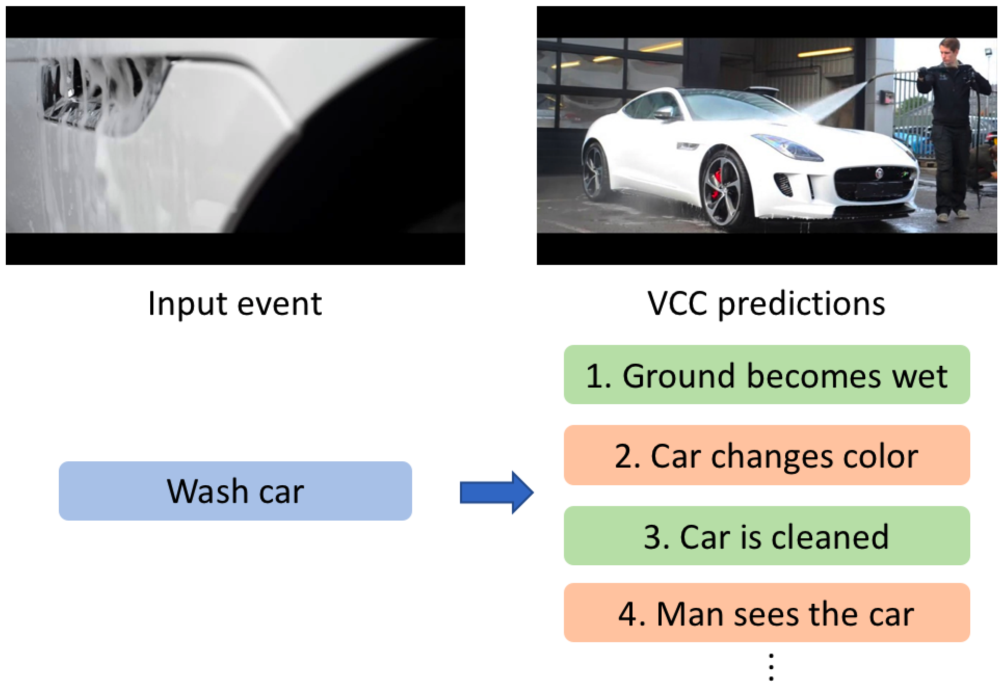


Figure 8.7: Given the event "wash car" in the first image, we show the top four events that VCC predicts to be contained in the second image and caused by "wash car." Correct and wrong predictions are indicated with green and orange backgrounds, respectively.

From the experimental results in Table 8.3, we can see that, compared with the "random guess" baseline, unsupervised BERT and GPT-2 only achieved slightly better performance. The reason behind this is that even though these pre-trained language representation models contain rich semantics about events, they can only distinguish which two events are more relevant rather than identify the causality between them. However, if we incorporate them into the VCC model and further train them, they will achieve much better performance, which shows that if we allow further training, the model will learn how to better use the contained rich semantics and thus achieve better performance. These results show that textual data alone is not enough to cover the rich and complex causal knowledge and further demonstrate the importance of this work, investigating how to mine causal knowledge from the visual data.

8.4.6 Case Study

To further analyze the success and limitation of the proposed VCC model, we present a case study, where predictions are sorted based on the prediction scores, in Figure 8.7, from which we can see that VCC successfully predict that "wash car" can cause "ground becomes wet" and "car is cleaned,” but it also makes mistakes. For example, based on these two images, humans know that the car does not change color, but VCC may mistakenly connect the event “change color” with “wash” from some other training examples. In this case, using a few objects to represent images may not be enough to cover all the visual information. Besides that, VCC also predicts “Man sees the car”, which happens in the second image but is not caused by “wash.” Understanding this might need inference over external knowledge. How to leverage external knowledge to better understand the visual signal and acquire more accurate causal knowledge is left for our future investigation.

PART 3

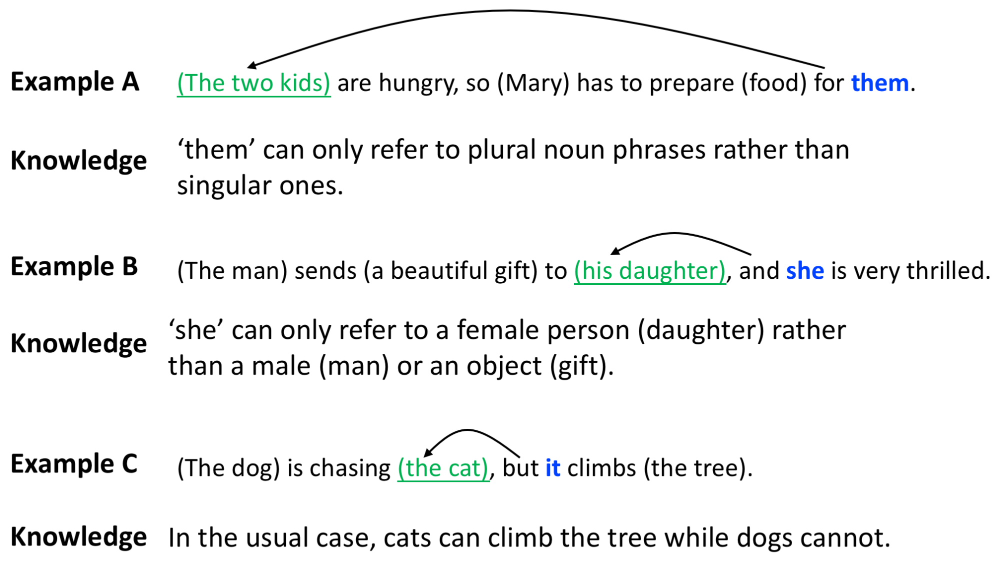
COMMONSENSE KNOWLEDGE APPLICATION

After introducing why higher-order selectional preference is a sound commonsense representation methodology in Part 1 and how can follow this methodology to acquire the commonsense knowledge from different modalities (i.e., text, existing knowledge graphs, and vision) in Part 2, in this part, we explore how to apply the structured knowledge for understanding natural language. We first use pronoun coreference resolution as the downstream task to investigate how can we use the structured knowledge to help current deep models. In Chapter 9, we try to convert structured knowledge into features and append them to local context representation captured by pre-trained language models. After that, to address the limitation that not all structured knowledge can be easily converted into numerical features (higher-order selectional preference), in Chapter 10, we propose a joint attention network to directly conduct inference over the original format of the structured knowledge. In the end of this part, we do not restrict to a specific task and try to explore a general commonsense inference model that can use knowledge from ASER to solve multiple natural language understanding tasks at the same time in Chapter 11.

CHAPTER 9

STRUCTURED KNOWLEDGE AS FEATURE FOR PRONOUN COREFERENCE RESOLUTION

The question of how human beings resolve pronouns has long been of interest to both linguistics and natural language processing (NLP) communities, for the reason that pronoun itself has weak semantic meaning [30] and brings challenges in natural language understanding. To explore solutions for that question, pronoun coreference resolution [53] was proposed. As an important yet vital sub-task of the general coreference resolution task, pronoun coreference resolution is to find the correct reference for a given pronominal anaphor in the context and has been shown to be crucial for a series of downstream tasks [96], including machine translation [97], summarization [155], information extraction [28], and dialog systems [156].



Example A (The two kids) are hungry, so (Mary) has to prepare (food) for **them**.

Knowledge 'them' can only refer to plural noun phrases rather than singular ones.

Example B (The man) sends (a beautiful gift) to (his daughter), and **she** is very thrilled.

Knowledge 'she' can only refer to a female person (daughter) rather than a male (man) or an object (gift).

Example C (The dog) is chasing (the cat), but **it** climbs (the tree).

Knowledge In the usual case, cats can climb the tree while dogs cannot.

Figure 9.1: Pronoun coreference examples, where each example requires different knowledge for its resolution. Blue bold font refers to the target pronoun, where the correct noun reference and other candidates are marked by green underline and brackets, respectively.

Conventionally, people design rules [53, 99, 95] or use features [100, 16, 74] to resolve the pronoun coreferences. These methods heavily rely on the coverage and quality of the manually defined rules and features. Until recently, end-to-end solution [69] was proposed towards solving the general coreference problem, where deep learning models were used to better capture contextual information. However, training such models on annotated corpora can be biased and normally does not consider external knowledge.

Despite the great efforts made in this area in the past few decades [53, 95, 100, 124], pronoun coreference resolution remains challenging. The reason behind is that the correct resolution of pronouns can be influenced by many factors [30]; many resolution decisions require reasoning upon different contextual and external knowledge [125], which is also proved in other NLP tasks [149, 150, 187]. Figure 9.1 demonstrates such requirement with three examples, where Example A depends on the plurality knowledge that "them" refers to plural noun phrases; Example B illustrates the gender requirement of pronouns where "she" can only refer to a female person (girl); Example C requires a more general type of knowledge[1] that "cats can climb trees but a dog normally does not." All of these knowledge are difficult to be learned from training data. Considering the importance of both contextual information and external human knowledge, how to jointly leverage them becomes an important question for pronoun coreference resolution.

In this chapter, we propose a two-layer model to address the question while solving two challenges of incorporating external knowledge into deep models for pronoun coreference resolution, where the challenges include: first, different cases have their knowledge preference, i.e., some knowledge is exclusively important for certain cases, which requires the model to be flexible in selecting appropriate knowledge per case; second, the availability of knowledge resources is limited and such resources normally contain noise, which requires the model to be robust in learning from them.

Consequently, in our model, the first layer predicts the relations between candidate noun phrases and the target pronoun based on the contextual information learned by neural networks. The second layer compares the candidates pair-wisely, in which we propose a knowledge attention module to focus on appropriate knowledge based on the given context. Moreover, a softmax pruning is placed in between the two layers to select high confident candidates. The architecture ensures the model being able to leverage both context and external knowledge. Especially, compared with conventional approaches that simply treat external knowledge as rules or features, our model is not only more flexible and effective but also interpretable as it reflects which knowledge source

[1]This is normally as selectional preference (SP) [53], which is defined as given a predicate (verb), a human has the preference for its argument (subject in this example).

has the higher weight in order to make the decision. Experiments are conducted on a widely used evaluation dataset, where the results prove that the proposed model outperforms all baseline models by a great margin.

9.1 The Task

Following the conventional setting [53], the task of pronoun coreference resolution is defined as: for a pronoun p and a candidate noun phrase set $\mathcal{N}$, the goal is to identify the correct non-pronominal references set[2] $\mathcal{C}$. the objective is to maximize the following objective function:

$$\mathcal{J} = \frac{\sum_{c \in \mathcal{C}} e^{F(c,p)}}{\sum_{n \in \mathcal{N}} e^{F(n,p)}}, \tag{9.1}$$

where c is the correct reference and n the candidate noun phrase. $F(\cdot)$ refers to the overall coreference scoring function for each n regarding p. Following [95], all non-pronominal noun phrases in the recent three sentences of the pronoun p are selected to form N.

Particularly in our setting, we want to leverage both the local contextual information and external knowledge in this task, thus for each n and p, $F(.)$ is decomposed into two components:

$$F(n, p) = F_c(n, p) + F_k(n, p), \tag{9.2}$$

where $F_c(n, p)$ is the scoring function that predicts the relation between n and p based on the contextual information; $F_k(n, p)$ is the scoring function that predicts the relation between n and p based on the external knowledge. There could be multiple ways to compute F_c and F_k, where a solution proposed in this chapter is described as follows.

9.2 The Model

The architecture of our model is shown in Figure 9.2, where we use two layers to incorporate contextual information and external knowledge. Specifically, the first layer takes the representations of different n and the p as input and predict the relationship between each pair of n and p, so as to compute F_c. The second layer leverages the external knowledge to compute F_k, which consists

[2]It is possible that a pronoun has multiple references.

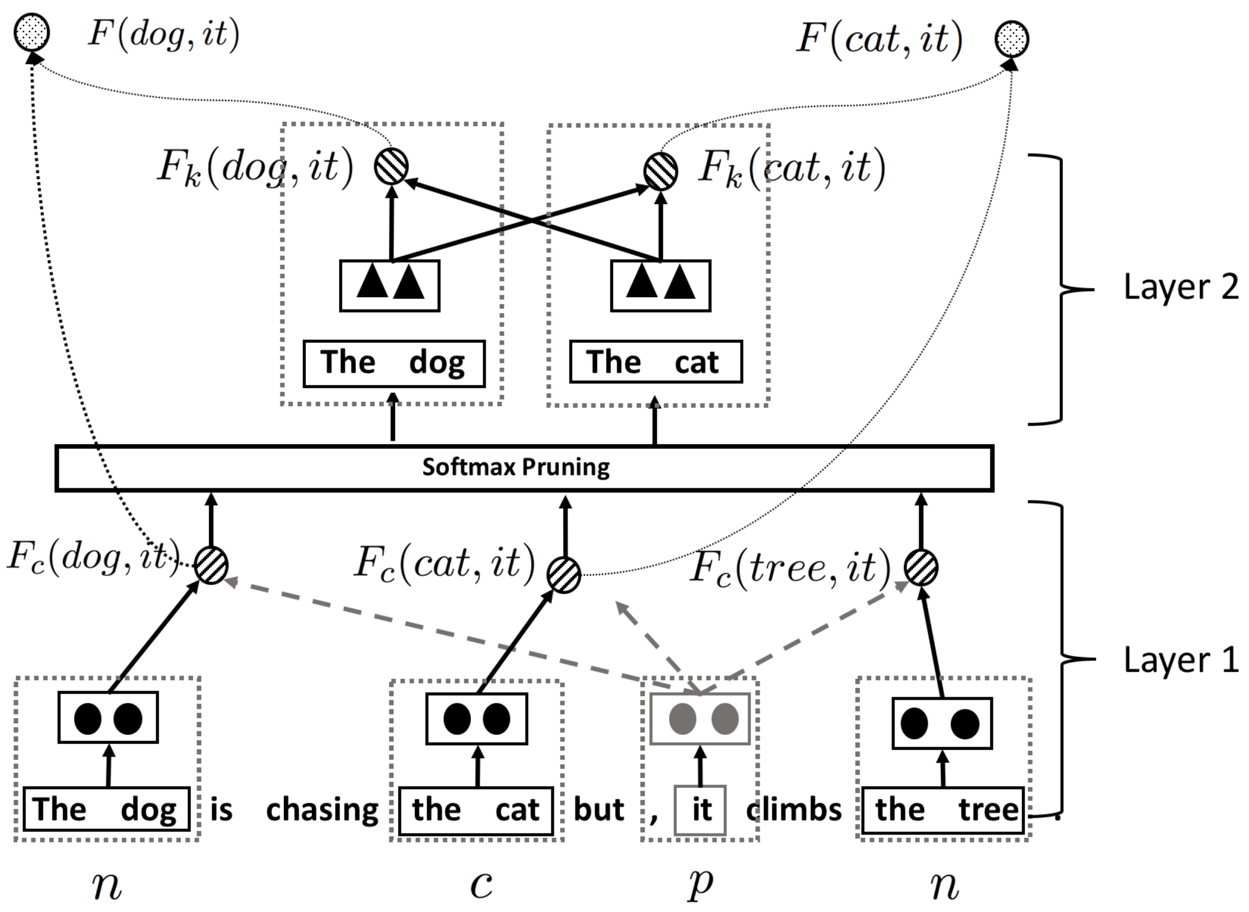


Figure 9.2: The architecture of the two-layer model for pronoun coreference resolution. The first layer encodes the contextual information for computing $\mathsf{F_c}$. The second layer leverages external knowledge to score $\mathsf{F_k}$. A pruning layer is applied in between the two layers to control computational complexity. The dashed boxes in the first and second layer refer to span representation and knowledge scoring, respectively.

of pair-wise knowledge score $\mathsf{f_k}$ among all candidate n. To enhance the efficiency of the model, a softmax pruning module is applied to select high confident candidates into the second layer. The details of the aforementioned components are described in the following subsections.

9.2.1 Encoding Contextual Information

Before $\mathsf{F_c}$ is computed, the contextual information is encoded through a span[3] representation (SR) module in the first layer of the model. Following Lee et al. [69], we adopt the standard bidirectional LSTM (biLSTM) [54] and the attention mechanism [4] to generate the span representation, as

[3]Both noun phrases and the pronoun are treated as spans.

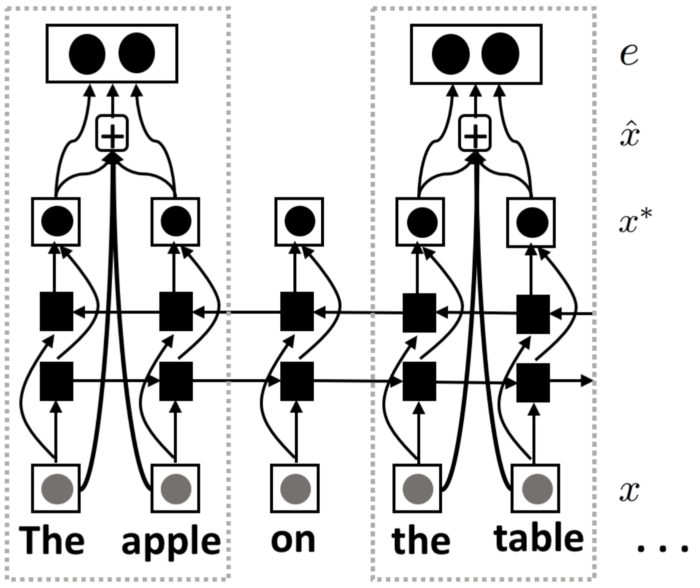


Figure 9.3: The structure of span representation. Bidirectional LSTM and inner-span attention mechanism are employed to capture the contextual information.

shown in Figure 9.3. Given that the initial word representations in a span n_i are $\mathbf{x}_1, ..., \mathbf{x}_T$,

we denote their representations $\mathbf{x}^*_1, ..., \mathbf{x}^*_T$ after encoded by the biLSTM. Then we obtain the inner-span attention by

$$a_t = \frac{e^{\alpha_t}}{\sum_{k=1}^{T} e^{\alpha_k}}, \tag{9.3}$$

where α_t is computed via a standard feed-forward neural network[4] $\alpha_t = \mathrm{NN}_\alpha(\mathbf{x}^*_t)$. Thus, we have the weighted embedding of each span $\hat{x}_i$ through

$$\hat{\mathbf{x}}_i = \sum_{k=1}^{T} a_k \cdot \mathbf{x}_k. \tag{9.4}$$

Afterwards, we concatenate the starting ($\mathbf{x}^*_{start}$) and ending ($\mathbf{x}^*_{end}$) embedding of each span, as well as its weighted embedding ($\hat{\mathbf{x}}_i$) and the length feature ($\phi(i)$) to form its final representation e:

$$\mathbf{e}_i = [\mathbf{x}^*_{start}, \mathbf{x}^*_{end}, \hat{\mathbf{x}}_i, \phi(i)]. \tag{9.5}$$

Once the span representation of $n \in \mathcal{N}$ and p are obtained, we compute F_c for each n with a

[4]We use NN to present feed-forward neural networks throughout this chapter.

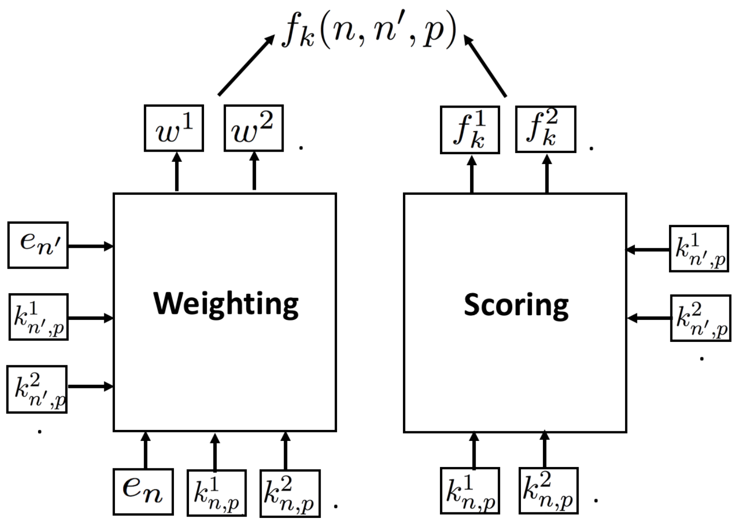


Figure 9.4: The structure of the knowledge attention module. For each feature k_i from knowledge source i, the the weighting component predict its weight w^i and the scoring component computes its knowledge score f_k^i. Then a weighted sum is obtained for f_k.

standard feed-forward neural network:

$$F_c(n, p) = NN_c([\mathbf{e}_n, \mathbf{e}_p, \mathbf{e}_n \odot \mathbf{e}_p]), \tag{9.6}$$

where $\odot$ is the element-wise multiplication.

9.2.2 Processing External Knowledge

In the second layer of our model, external knowledge is leveraged to evaluate all candidate n so as to give them reasonable F_k scores. In doing so, each candidate is represented as a group of features from different knowledge sources, e.g., "the cat" can be represented as a singular noun, unknown gender creature, and a regular subject of the predicate verb "climb." For each candidate, we conduct a series of pair-wise comparisons between it and all other ones to result in its F_k score. An attention mechanism is proposed to perform the comparison and selectively use the knowledge features. Consider there exists noise in external knowledge, especially when it is automatically generated, such attention mechanism ensures that, for each candidate, reliable and useful knowledge is utilized rather than ineffective ones. The details of the knowledge attention module and the overall scoring are described as follows.

Knowledge Attention Figure 9.4 demonstrates the structure of the knowledge attention module, where there are two components: (1) weighting: assigning weights to different knowledge features regarding their importance in the comparison; (2) scoring: valuing a candidate against another one based on their features from different knowledge sources. Assuming that there are m knowledge sources input to our model, each candidate can be represented by m different features, which are encoded as embeddings. Therefore, two candidates n and n' regarding p have their knowledge feature embeddings $\mathbf{k}^1_{n,p}, \mathbf{k}^2_{n,p}, ..., \mathbf{k}^m_{n,p}$ and $\mathbf{k}^1_{n',p}, \mathbf{k}^2_{n',p}, ..., \mathbf{k}^m_{n',p}$, respectively. The weighting component receives all features $\mathbf{k}$ for n and n', and the span representations $\mathbf{e}_n$ and $\mathbf{e}_{n'}$ as input, where $\mathbf{e}_n$ and $\mathbf{e}_{n'}$ help selecting appropriate knowledge based on the context. As a result, for a candidate pair (n, n') and a knowledge source i, its knowledge attention score is computed via

$$\beta_i(n, n', p) = NN_{ka}([\mathbf{o}^i_{n,p}, \mathbf{o}^i_{n',p}, \mathbf{o}^i_{n,p} \odot \mathbf{o}^i_{n',p}]), \tag{9.7}$$

where $\mathbf{o}^i_{n,p} = [\mathbf{e}_n, \mathbf{k}^i_{n,p}]$ and $\mathbf{o}^i_{n',p} = [\mathbf{e}_{n'}, \mathbf{k}^i_{n',p}]$ are the concatenation of span representation and external knowledge embedding for candidate n and n' respectively. The weight for features from different knowledge sources is thus computed via

$$w_i = \frac{e^{\beta_i}}{\sum_{j=1}^{m} e^{\beta_j}}. \tag{9.8}$$

Similar to the weighting component, for each feature i, we compute its score $f^i_k(n, n', p)$ for n against n' in the scoring component through

$$f^i_k(n, n', p) = NN_{ks}([\mathbf{k}^i_{n,p}, \mathbf{k}^i_{n',p}, \mathbf{k}^i_{n,p} \odot \mathbf{k}^i_{n',p}]). \tag{9.9}$$

where it is worth noting that we exclude $\mathbf{e}$ in this component for the reason that, in practice, the dimension of $\mathbf{e}$ is normally much higher than $\mathbf{k}$. As a result, it could dominate the computation if $\mathbf{e}$ and $\mathbf{k}$ is concatenated.[5]

Once the weights and scores are obtained, we have a weighted knowledge score for n against n':

$$f_k(n, n', p) = \sum_{i=1}^{m} w_i \cdot f^i_k(n, n', p). \tag{9.10}$$

Overall Knowledge Score After all pairs of n and n' are processed by the attention module, the overall knowledge score for n is computed through the averaged $f_k(n, n', p)$ over all n':

$$F_k(n, p) = \frac{\sum_{n' \in \mathcal{N}_o} f_k(n, n', p)}{|\mathcal{N}_o|}, \tag{9.11}$$

[5]We do not have this concern for the weighting component because the softmax (c.f. Eq. 9.8) actually amplifies the difference of β even if they are not much differentiated.

type	train	dev	test	all
Third Personal	21,828	2,518	3,530	27,876
Possessive	7,749	1,007	1,037	9,793
All	29,577	3,525	4,567	37,669

Table 9.1: Statistics of the evaluation dataset. Number of selected pronouns are reported.

where $\mathcal{N}_o = \mathcal{N} - n$ for each n.

9.2.3 Softmax Pruning

Normally, there could be many noun phrases that serve as the candidates for the target pronoun. One potential obstacle in the pair-wise comparison of candidate noun phrases in our model is the squared complexity $O(|\mathcal{N}|^2)$ with respect to the size of $\mathcal{N}$. To filter out low confident candidates so as to make the model more efficient, we use a softmax-pruning module between the two layers in our model to select candidates for the next step. The module takes F_c as input for each n, uses a softmax computation:

$$\hat{F}_c(n, p) = \frac{e^{F_c(n,p)}}{\sum_{n_i \in \mathcal{N}} e^{F_c(n_i,p)}}. \tag{9.12}$$

where candidates with higher $\hat{F}_c$ are kept, based on a threshold t predefined as the pruning standard. Therefore, if candidates have similar F_c scores, the module allow more of them to proceed to the second layer. Compared with other conventional pruning methods [69, 70] that generally keep a fixed number of candidates, our pruning strategy is more efficient and flexible.

9.3 Experiment Settings

9.3.1 Data

The CoNLL-2012 shared task [117] corpus is used as the evaluation dataset, which is selected from the Ontonotes 5.0.[6] Following conventional approaches [100, 74], for each pronoun in the document, we consider candidate n from the previous two sentences and the current sentence. For pronouns, we consider two types of them following Ng [100], i.e., third personal pronoun (*she*, *her*,

[6]https://catalog.ldc.upenn.edu/LDC2013T19

he, *him*, *them*, *they*, *it*) and possessive pronoun (*his*, *hers*, *its*, *their*, *theirs*). Table 9.1 reports the number of the two type pronouns and the overall statistics for the experimental dataset. According to our selection range of candidate n, on average, each pronoun has 4.6 candidates and 1.3 correct references.

9.3.2 Knowledge Types

In this study, we use two types of knowledge in our experiments. The first type is linguistic features, i.e., plurality and animacy & gender. We employ the Stanford parser,[7] which generates plurality, animacy, and gender markups for all the noun phrases, to annotate our data. Specifically, the plurality feature denotes each n and p to be singular or plural. For each candidate n, if its plurality status is the same as the target pronoun, we label it 1, otherwise 0. The animacy & gender (AG) feature denotes whether a n or p is a living object, and being male, female, or neutral if it is alive. For each candidate n, if its AG feature matches the target pronoun's, we label it 1, otherwise 0.

The second type is the selectional preference (SP) knowledge. For this knowledge, we create a knowledge base by counting how many times a predicate-argument tuple appears in a corpus and use the resulted number to represent the preference strength. Specifically, we use the English Wikipedia[8] as the base corpus for such counting. Then we parse the entire corpus through the Stanford parser and record all dependency edges in the format of *(predicate, argument, relation, number)*, where predicate is the governor and argument the dependent in the original parsed dependency edge.[9] Later for sentences in the training and test data, we firstly parse each sentence and find out the dependency edge linking p and its corresponding predicate. Then for each candidate[10] n in a sentence, we check the previously created SP knowledge base and find out how many times it appears as the argument of different predicates with the same dependency relation (i.e., *nsubj* and *dobj*). The resulted frequency is grouped into the following buckets [1, 2, 3, 4, 5-7, 8-15, 16-31, 32-63, 64+] and we use the bucket id as the final SP knowledge. Thus in the previous example:

[7]https://stanfordnlp.github.io/CoreNLP/

[8]https://dumps.wikimedia.org/enwiki/

[9]In Stanford parser results, when a verb is a linking verb (e.g., am, is), an 'nsubj' edge is created between its predicative and subject. Thus for this case the predicative is treated as the predicate for the subject (argument) in our study.

[10]If a noun phrase contains multiple words, we use the parsed result to locate its keyword and use it to represent the entire noun phrase.

The dog is chasing *the cat* but **it** climbs *the tree*.

Its parsing result indicates that "**it**" is the subject of the verb "climb." Then for "*the dog*," "*the cat*," and "*the tree*," we check their associations with "climb" in the knowledge base and group them in the buckets to form the SP knowledge features.

9.3.3 Baselines

Several baselines are compared in this work. The first two are conventional unsupervised ones:

- **Recent Candidate**, which simply selects the most recent noun phrase that appears in front of the target pronoun.
- **Deterministic** model [123], which proposes one multi-pass seive model with human designed rules for the coreference resolution task.

Besides the unsupervised models, we also compare with three representative supervised ones:

- **Statistical** model, proposed by Clark and Manning [18], uses human-designed entity-level features between clusters and mentions for coreference resolution.
- **Deep-RL** model, proposed by Clark and Manning [19], a reinforcement learning method to directly optimize the coreference matrix instead of the traditional loss function.
- **End2end** is the current state-of-the-art coreference model [70], which performs in an end-to-end manner and leverages both the contextual information and a pre-trained language model [114].

Note that the Deterministic, Statistical, and Deep-RL models are included in the Stanford CoreNLP toolkit,[11] and experiments are conducted with their provided code. For End2end, we use their released code[12] and replace its mention detection component with gold mentions for the fair comparison.

To clearly show the effectiveness of the proposed model, we also present a variation of our model as an extra baseline to illustrate the effect of different knowledge incorporation manner:

[11] https://stanfordnlp.github.io/CoreNLP/coref.html

[12] https://github.com/kentonl/e2e-coref

Model	Third Personal Pronoun			Possessive Pronoun			All		
	P	R	F1	P	R	F1	P	R	F1
Recent Candidate	50.7	40.0	44.7	64.1	45.5	53.2	54.4	41.6	47.2
Deterministic [123]	68.7	59.4	63.7	51.8	64.8	57.6	62.3	61.0	61.7
Statistical [18]	69.1	62.6	65.7	58.0	65.3	61.5	65.3	63.4	64.3
Deep-RL [19]	72.1	68.5	70.3	62.9	74.5	68.2	68.9	70.3	69.6
End2end [70]	75.1	83.7	79.2	73.9	82.1	77.8	74.8	83.2	78.8
Feature Concatenation	73.5	88.3	80.2	72.5	87.3	79.2	73.2	87.9	79.9
The Complete Model	75.4	87.9	**81.2**	74.9	87.2	**80.6**	75.2	87.7	**81.0**

Table 9.2: Pronoun coreference resolution performance of different models on the evaluation dataset. Precision (P), recall (R), and F1 score are reported, with the best one in each F1 column marked bold.

- **Feature Concatenation**, a simplified version of the complete model that removes the second knowledge processing layer, but directly treats all external knowledge embeddings as features and concatenates them to span representations.

9.3.4 Implementation

Following previous work [70], we use the concatenation of the 300d GloVe embeddings [113] and the ELMo [114] embeddings as the initial word representations. Out-of-vocabulary words are initialized with zero vectors. Hyper-parameters are set as follows. The hidden state of the LSTM module is set to 200, and all the feed-forward networks in our model have two 150-dimension hidden layers. The default pruning threshold t for softmax pruning is set to 10^{-7}. All linguistic features (plurality and AG) and external knowledge (SP) are encoded as 20-dimension embeddings.

For model training, we use cross-entropy as the loss function and Adam [64] as the optimizer. All the aforementioned hyper-parameters are initialized randomly, and we apply dropout rate 0.2 to all hidden layers in the model. Our model treats a candidate as the correct reference if its predicted overall score $F(n, p)$ is larger than 0. The model training is performed with up to 100 epochs, and the best one is selected based on its performance on the development set.

	F1	ΔF1
The Complete Model	81.0	-
–Plurality knowledge	80.7	-0.3
–AG knowledge	80.5	-0.5
–SP knowledge	80.4	-0.6
–Knowledge Attention	80.1	-0.9

Table 9.3: Performance of our model with removing different knowledge sources and knowledge attention.

9.4 Experimental Results

Table 9.2 compares the performance of our model with all baselines. Overall, our model performs the best with respect to all evaluation metrics. Several findings are also observed from the results. First, manually defined knowledge and features are not enough to cover rich contextual information. Deep learning models (e.g., End2end and our proposed models), which leverage text representations for context, outperform other approaches by a great margin, especially on the recall. Second, external knowledge is highly helpful in this task, which is supported by that our model outperforms the End2end model significantly.

Moreover, the comparison between the two variants of our models is also interesting, where the final two-layer model outperforms the Feature Concatenation model. It proves that simply treating external knowledge as the feature, even though they are from the same sources, is not as effective as learning them in a joint framework. The reason behind this result is mainly from the noise in the knowledge source, e.g., parsing error, incorrectly identified relations, etc. For example, the plurality of 17% noun phrases are wrongly labeled in the test data. As a comparison, our knowledge attention might contribute to alleviate such noise when incorporating all knowledge sources.

Effect of Different Knowledge To illustrate the importance of different knowledge sources and the knowledge attention mechanism, we ablate various components of our model and report the corresponding F1 scores on the test data. The results are shown in Table 9.3, which clearly show the necessity of the knowledge. Interestingly, AG contributes the most among all knowledge types, which indicates that potentially more cases in the evaluation dataset demand on the AG knowledge than others. More importantly, the results also prove the effectiveness of the knowledge attention module, which contributes to the performance gap between our model and the Feature Concatena-

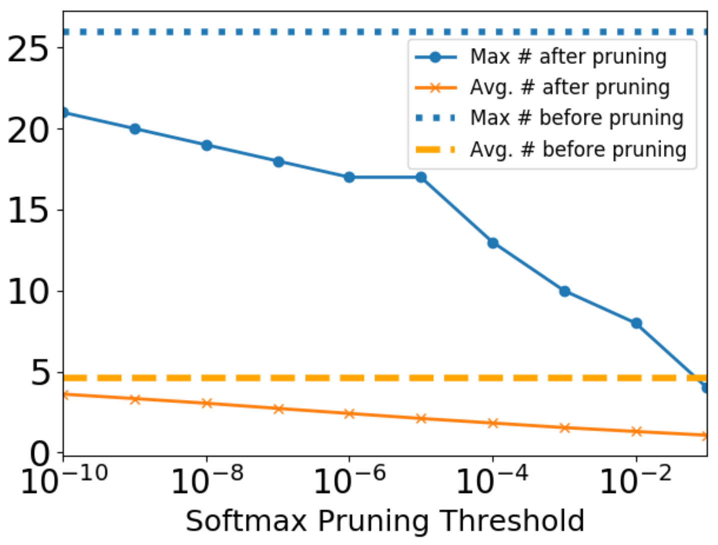


Figure 9.5: Effect of different thresholds on candidate numbers. Max and Average number of candidates after pruning are represented with solid lines in blue and orange, respectively. Two dashed lines indicate the max and the average number of candidates before pruning.

tion one.

Effect of Different Pruning Thresholds We try different thresholds t for the softmax pruning in selecting reliable candidates. The effects of different thresholds on reducing candidates and overall performance are shown in Figure 9.5 and 9.6 respectively. Along with the increase of t, both the max and the average number of pruned candidates drop quickly, so that the space complexity of the model can be reduced accordingly. Particularly, there are as much as 80% candidates can be filtered out when $t = 10^{-1}$. Meanwhile, when referring to Figure 9.6, it is observed that the model performs stable with the decreasing of candidate numbers. Not surprisingly, the precision rises when reducing candidate numbers, yet the recall drops dramatically, eventually results in the drop of F1. With the above observations, the reason we set $t = 10^{-7}$ as the default threshold is straightforward: on this value, one-third candidates are pruned with almost no influence on the model performance in terms of precision, recall, and the F1 score.

9.5 Case Study

To further demonstrate the effectiveness of incorporating knowledge into pronoun coreference resolution, two examples are provided for detailed analysis. The prediction results of the End2end model and our complete model are shown in Table 9.4. There are different challenges in both ex-

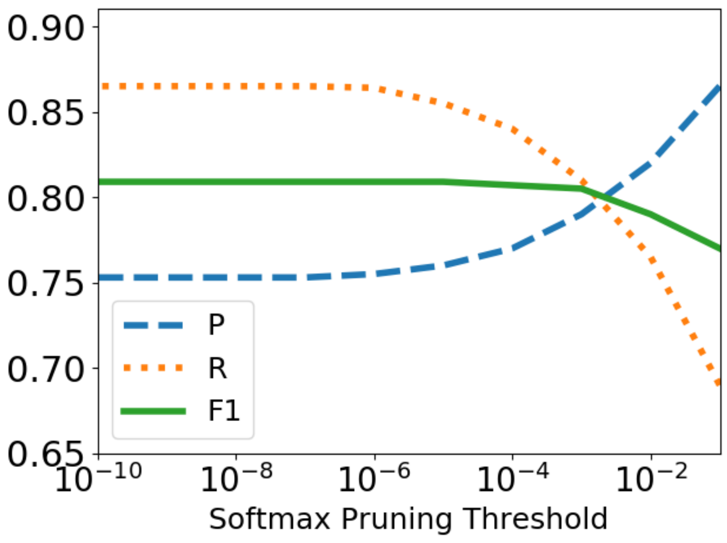


Figure 9.6: Effect of different pruning thresholds on model performance. With the threshold increasing, the precision increases while the recall and F1 drop.

amples. In Example A, "Jesus," "man," and "my son" are all similar (male) noun phrases matching the target pronoun "*He*." The End2end model predicts all of them to be correct references because their context provides limited help in distinguishing them. In Example B, the distance between "an accident" and the pronoun "it" is too far. As a result, the "None" result from the End2end model indicates that the contextual information is not enough to make the decision. As a comparison, in our model, integrating external knowledge can help to solve such challenges, e.g., for Example A, SP knowledge helps when Plurality and AG cannot distinguish all candidates.

To clearly illustrate how our model leverages the external knowledge, we visualize the knowledge attention of the correct reference against other candidates[13] via heatmaps in Figure 9.7. Two interesting observations are drawn from the visualization. First, given two candidates, if they are significantly different in one feature, our model tends to pay more attention to that feature. Take AG as an example, in Example A, the AG features of all candidates consistently match the pronoun "he" (all male/neutral). Thus the comparison between "my son" and all candidates pay no attention to the AG feature. While in Example B, the target pronoun "it" cannot describe human, thus "father" and "friend" are 0 on the AG feature while "hospital" and "accident" are 1. As a result, the attention module emphasizes AG more than other knowledge types. Second, The importance of SP is clearly shown in these examples. In example A, Plurality and AG features cannot help,

[13]Only candidates entered the second layer are considered.

	Example A	Example B
Sentences	... (A large group of people) met (Jesus). (A man in the group) shouted to him: "(Teacher), please come and look at (my son). **He** is the only child I have" ...	... (My neighbor) told me that there was (an accident), and everyone else was intact, except (his father), who was in (hospital) for fractures. I comforted him first and asked (my friend) to rush me to (the hospital). (My neighbor) showed me the police report at (the hospital), which indicated **it** was all my neighbor's fault. ...
Pronoun Candidate NPs	**He** A large group of people, Jesus, A man in the group, Teacher, my son.	**it** My friend, an accident, his father, hospital, my friend, the hospital, My neighbor, the hospital.
End2end	Jesus, A man in the group, my son	None
Our Model	my son	an accident

Table 9.4: The comparison of End2end and our model on two examples drawn from the test data. Pronouns are marked as blue bold font. Correct references are indicated in green underline font and other candidates are indicated with brackets. "None" refers to that none of the candidates is predicated as the correct reference.

the attention module weights higher on SP because "son" appears 100 times as the argument of the parsed predicate "child" in the SP knowledge base, while other candidates appear much less at that position. In example B, as mentioned above, once AG helps filtering "hospital" and "accident," SP plays an important role in distinguishing them because "accident" appears 26 times in the SP knowledge base as the argument of the "fault" from the results of the parser, while "hospital" never appears at that position.

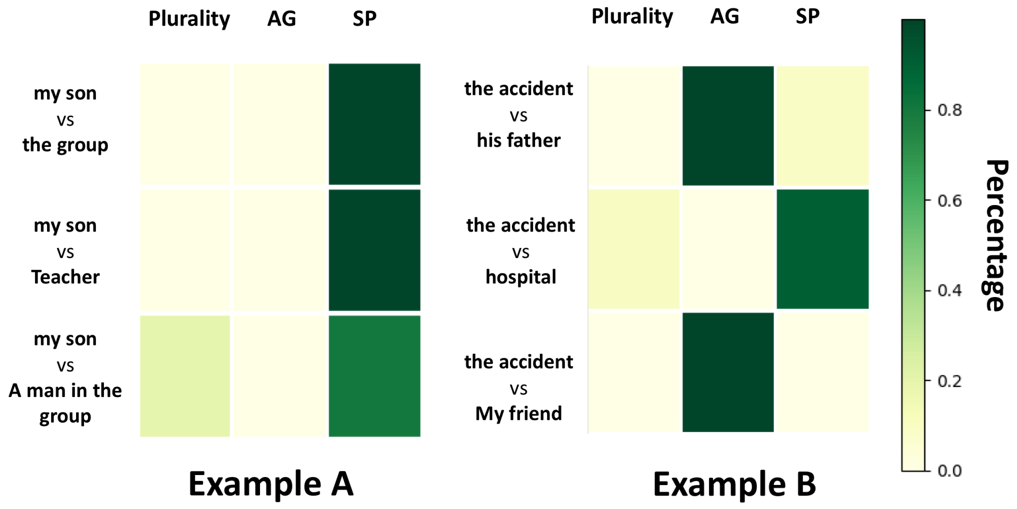


Figure 9.7: Heatmaps of knowledge attention for two examples, where in each example the knowledge attention weights of the correct references against other candidates are illustrated. Darker color refers to higher weight on the corresponding knowledge type.

CHAPTER 10

AN END-TO-END STRUCTURED KNOWLEDGE ENHANCED PCR SYSTEM

A significant limitation of the aforementioned feature-based approach is that it cannot handle complex knowledge because we cannot easily convert them into features. Moreover, the feature computation phase still requires the support of either gold mentions or other mention detection modules, which could bring error propagation.

To address these limitations, we propose a novel end-to-end model that learns to resolve pronoun coreferences with general knowledge graphs (KGs). Different from conventional approaches, our model does not require to use featurized knowledge. Instead, we directly encode knowledge triplets, the most common format of modern knowledge graphs, into our model. In doing so, the learned model can be easily applied across different knowledge types as well as domains with adopted KG. Moreover, to address the knowledge matching issue, we propose a knowledge attention module in our model, which learns to select the most related and helpful knowledge triplets according to different contexts. Experiments conducted on general (news) and in-domain (medical) cases shows that the proposed model outperforms all baseline models by a great margin. Additional experiments with the cross-domain setting further illustrate the validity and effectiveness of our model in leveraging knowledge smartly rather than fitting with limited training data.

10.1 The Task

Different from the setting in Chapter 9, we do not require the gold mentions. Thus we can formally define the task as follows. Given a text D, which contains a pronoun p, the goal is to identify all the mentions that p refers to. We denote the correct mentions p refers to as $c \in \mathcal{C}$, where $\mathcal{C}$ is the correct mention set. Similarly, each candidate span is denoted as $s \in \mathcal{S}$, where $\mathcal{S}$ is the set of all candidate spans. Note that in the case where no golden mentions are annotated, all possible spans in D are used to form $\mathcal{S}$. To exploit knowledge, we denote the knowledge set as $\mathcal{G}$, instantiated by

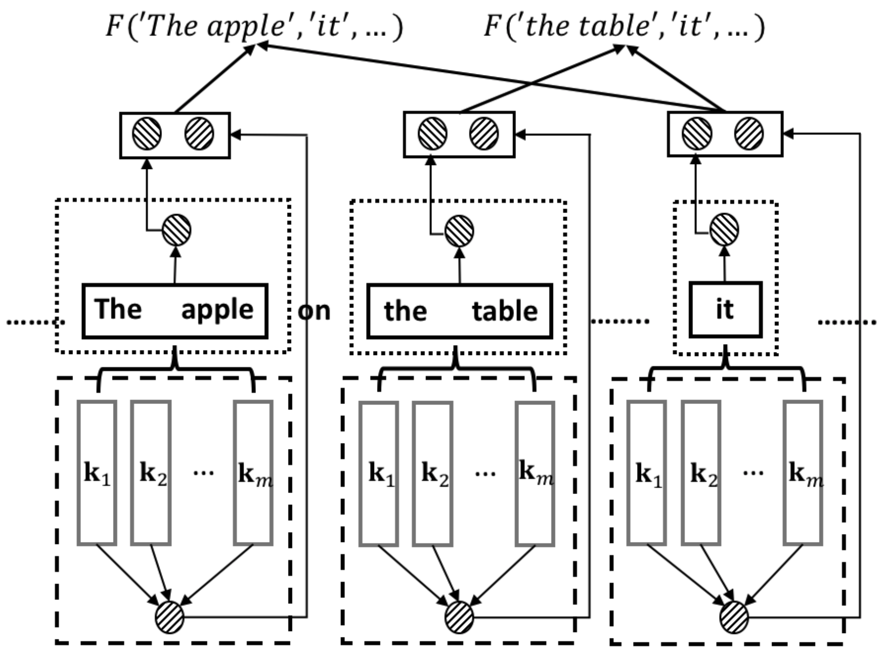


Figure 10.1: The overall framework of our approach to pronoun corference resolution with KGs. $\mathbf{k}_1$,...,$\mathbf{k}_m$ represent the retrieved knowledge for each span in the black boxes. Dotted box represents the span representation module, which generates a contextual representation for each span. Dashed box represents the knowledge selection module, which selects appropriate knowledge based on the context and generates an overall knowledge representation for each span. $\mathsf{F}(\cdot)$ is the overall coreference scoring function.

multiple knowledge triplets.[1] The task is thus to identify $\mathcal{C}$ out of $\mathcal{S}$ with the support of $\mathcal{G}$. Formally, it optimizes

$$\mathcal{J} = \frac{\sum_{c \in \mathcal{C}} e^{\mathsf{F}(c,p,\mathcal{G},D)}}{\sum_{s \in \mathcal{S}} e^{\mathsf{F}(s,p,\mathcal{G},D)}}, \tag{10.1}$$

where $\mathsf{F}(\cdot)$ is the overall scoring function[2] of p referring to s in D with $\mathcal{G}$. The details of F are illustrated in the following section.

10.2 Model

The overall framework of our model is shown in Figure 10.1. There are several layers in it. At the bottom, we encode all mention spans (s) and pronouns (p) into embeddings so as to incorporate

[1]Each triplet contains a head, a tail, and a relation from the head to the tail.

[2]We omit $\mathcal{G}$ and D in the rest of this chapter for simplicity.

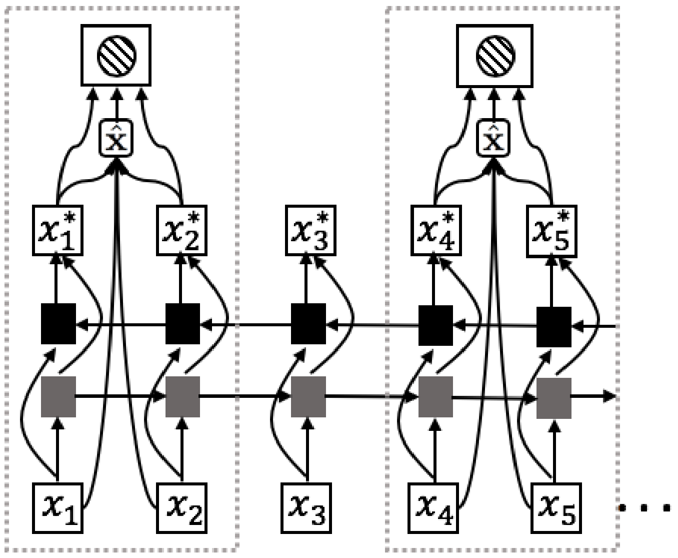


Figure 10.2: The structure of the span representation module. BiLSTM and attention are employed to encode the contextual information.

contextual information. In the middle layer, for each pair of (s, p), we use their embeddings to select the most helpful knowledge triplets from $\mathcal{G}$ and generate the knowledge representation of s and p. At the top layer, we concatenate the textual and knowledge representation as the final representation of each s and p, and then use this representation to predict whether there exists the coreference relation between them.

10.2.1 Span Representation

Contextual information is crucial to distinguish the semantics of a word or phrase, especially for text representation learning [150, 148]. In this work, a standard bidirectional LSTM (BiLSTM) [54] model is used to encode each span with attentions [4], which is similar to the one used in Lee et al. [69]. The structure is shown in Figure 10.2. Let initial word embeddings in a span s_i be denoted as $\mathbf{x}_1, ..., \mathbf{x}_T$ and their encoded representation be $\mathbf{x}_1^*, ..., \mathbf{x}_T^*$. The weighted embeddings of each span $\hat{\mathbf{x}}_i$ is obtained by

$$\hat{\mathbf{x}}_i = \sum_{t=1}^{T} a_t \cdot \mathbf{x}_t, \tag{10.2}$$

where a_t is the inner-span attention computed by

$$a_t = \frac{e^{\alpha_t}}{\sum_{k=1}^{T} e^{\alpha_k}}, \tag{10.3}$$

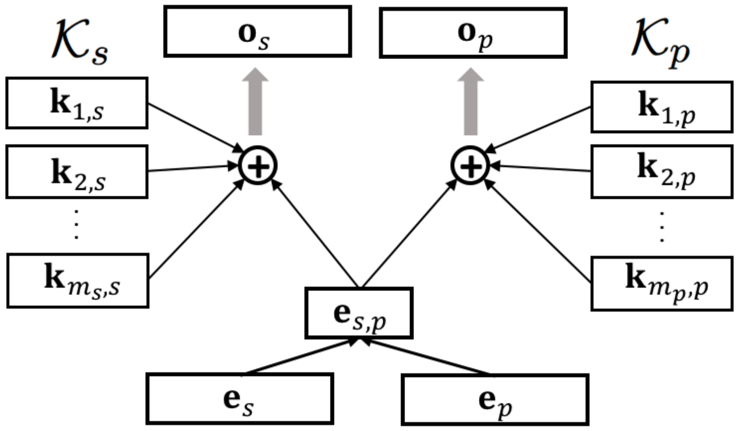


Figure 10.3: The structure of the knowledge attention module. The joint representation of the candidate span and pronoun is used to select knowledge for s and p.

where α_t is a standard feed-forward neural network[3] $\alpha_t = \mathrm{NN}_\alpha(\mathbf{x}_t^*)$.

Finally, the starting ($\mathbf{x}_{start}^*$) and ending ($\mathbf{x}_{end}^*$) embedding of each span is concatenated with the weighted embedding ($\hat{\mathbf{x}}_i$) and the length feature ($\phi(i)$) to form its final representation $\mathbf{e}$:

$$\mathbf{e}_i = [\mathbf{x}_{start}^*, \mathbf{x}_{end}^*, \hat{\mathbf{x}}_i, \phi(i)]. \tag{10.4}$$

Thus the span representation of s and p are marked as $\mathbf{e}_s$ and $\mathbf{e}_p$, respectively.

10.2.2 Knowledge Representation

For each candidate span s and the target pronoun p, different knowledge from a KG can be extracted with various methods. For simplicity and generalization consideration, we use the string match in our model for knowledge extraction. Specifically, for each triplet $t \in \mathcal{G}$ where the head and tail of t are both lists of words, if its head is the same as the string of s, we consider it to be a related triplet. Therefore, we encode the information of t by the averaging embeddings of all words in its tail. For example, if s is "the apple" and the knowledge triplet ("the apple," *IsA*, "healthy food") is found by searching the KG, we represent this relation from the averaged embeddings of "healthy" and "food." Consequently, for s and p, we denote their retrieved knowledge set as $\mathcal{K}_s$ and $\mathcal{K}_p$ respectively, where $\mathcal{K}_s$ contains m_s related knowledge embeddings $\mathbf{k}_{1,s}$, $\mathbf{k}_{2,s}$, ..., $\mathbf{k}_{m_s,s}$ and $\mathcal{K}_p$ contains m_p of them $\mathbf{k}_{1,p}$, $\mathbf{k}_{2,p}$, ..., $\mathbf{k}_{m_p,p}$.

[3]We use NN to present feed-forward neural networks.

To incorporate the aforementioned knowledge embeddings into our model, we face a challenge that there are a huge number of such embeddings while most of them are useless in certain contexts. To solve it, a knowledge attention module is proposed to select the appropriate knowledge.

For each pair of (s, p), as shown in Figure 10.3, we first concatenate $\mathbf{e}_s$ and $\mathbf{e}_p$ to get the overall (span, pronoun) representation $\mathbf{e}_{s,p}$, which is used to select knowledge for both s and p. Taking that for s as example, we compute the weight of each $\mathbf{k}_i \in \mathcal{K}_s$ by

$$w_i = \frac{e^{\beta_{\mathbf{k}_i}}}{\sum_{\mathbf{k}_j \in \mathcal{K}_s} e^{\beta_{\mathbf{k}_j}}}, \tag{10.5}$$

where $\beta_{\mathbf{k}} = \mathrm{NN}_\beta([\mathbf{e}_{s,p}, \mathbf{k}])$. As a result, the knowledge of s is summed by

$$\mathbf{o}_s = \sum_{\mathbf{k}_i \in \mathcal{K}_s} w_i \cdot \mathbf{k}_i. \tag{10.6}$$

to represent the overall knowledge for s. A similar process is also conducted for p with its knowledge representation $\mathbf{o}_p$.

10.2.3 Scoring

The final score of each pair (s, p) is computed by

$$F(s,p) = f_m(s) + f_c(s,p), \tag{10.7}$$

where $f_m(s) = \mathrm{NN}_m([\mathbf{e}_s, \mathbf{o}_s])$ is the scoring function for s to be a valid mention and $f_c(s,p) = \mathrm{NN}_c([\mathbf{e}_n, \mathbf{o}_n, \mathbf{e}_p, \mathbf{o}_p, \mathbf{e}_n \odot \mathbf{e}_p, \mathbf{o}_n \odot \mathbf{o}_p])$ is the scoring function to identify whether there exists a coreference relation from p to s, with $\odot$ denoting element-wise multiplication.

After getting the coreference score for all mention spans, we adopt a softmax selection on the most confident candidates for the final prediction, which is formulated as

$$\hat{F}(s,p) = \frac{e^{F(s,p)}}{\sum_{s_i \in \mathcal{S}} e^{F(s_i,p)}}. \tag{10.8}$$

where candidates with score $\hat{F}$ higher than a threshold t are selected.

10.3 Experiments

Experiments are illustrated in this section.

10.3.1 Datasets

Two datasets are used in our experiments, where they are from two different domains:

- **CoNLL:** The CoNLL-2012 shared task [117] corpus, which is a widely used dataset selected from the Ontonotes 5.0.[4]
- **i2b2:** The i2b2 shared task dataset [163], consisting of electronic medical records from two different organizations, namely, Partners HealthCare (Part) and Beth Israel Deaconess medical center (Beth). All records have been fully de-identified and manually annotated with coreferences.

We split the datasets into different proportions based on their original settings. Three types of pronouns are considered in this chapter following Ng [100], i.e., third personal pronoun (e.g., *she*, *her*, *he*, *him*, *them*, *they*, *it*), possessive pronoun (e.g., *his*, *hers*, *its*, *their*, *theirs*), and demonstrative pronoun (e.g., *this*, *that*, *these*, *those*). Table 10.1 reports the number of the three types of pronouns and the overall statistics of the experiment datasets with proportion splittings. Following conventional approaches [100, 74], for each pronoun, we consider its candidate mentions from the previous two sentences and the current sentence it belongs to. According to our selection range of the candidate mentions, each pronoun in the CoNLL data and i2b2 data has averagely 1.3 and 1.4 correct references, respectively.

10.3.2 Knowledge Resources

As mentioned in previous sections, our model is designed to leverage general KGs, where it takes triplets as the input of knowledge representations. For all knowledge resources, we format them as triplets and merge them together to obtain the final knowledge set. Different knowledge resources are introduced as follows.

[4]https://catalog.ldc.upenn.edu/LDC2013T19

Dataset		TP	Poss	Dem	All
CoNLL	train	21,828	7,749	2,229	31,806
	dev	2,518	1,007	222	3,747
	test	2720	1,037	321	4,078
i2b2	train	2,024	685	270	2,979
	test	1,244	367	166	1,777
Overall		30,334	10,845	3,208	44,387

Table 10.1: Statistics of the two datasets. "TP," "Poss," and "Dem" refer to third personal, possessive, and demonstrative pronouns, respectively.

Commonsense knowledge graph (OMCS). We use the largest commonsense knowledge base, the open mind common sense (OMCS) [145]. OMCS contains 600K crowd-sourced commonsense triplets such as *(food, UsedFor, eat)* and *(wind, CapableOf, blow to east)*. All relations in OMCS are human-defined and we select those highly-confident ones (confidence score larger than 2) to form the OMCS KG, with 62,730 triplets.

Medical concepts (Medical-KG). Being part of the i2b2 contest, the related knowledge about medical concepts such as *(the CT scan, is, test)* and *(intravenous fluids, is, treatment)* are provided. The annotated triplets are used as the medical concept KG, which contains 22,234 triplets.

Linguist features (Ling). In addition to manually annotated KGs, we also consider linguist features, i.e., plurality and animacy & gender (AG), as one important knowledge resources. Stanford parser[5] is employed to generate plurality, animacy, and gender markups for all the noun phrases, so as to automatically generate linguistic knowledge (in the form of triplets) for our data. Specifically, the plurality feature denotes each s and p to be singular or plural. The animacy & gender (AG) feature denotes whether the n or p is a living object, and being male, female, or neutral if it is alive. For example, a mention "the girls" is labeled as *plural* and *female*; we use triplets *("the girls," plurality, Plural)* and *("the girls," AG, female)* to represent them. As a result, we have 40,149 and 40,462 triplets for plurality and AG, respectively.

Selectional Preference (SP). Selectional preference [53] knowledge is employed as the last knowledge resource, which is the semantic constraint for word usage. SP generally refers to that, given a predicate (e.g., verb), people have the preference for the argument (e.g., its object or subject) con-

[5]https://stanfordnlp.github.io/CoreNLP/

nected. To collect SP knowledge, we first parse the English Wikipedia[6] with the Stanford parser and extract all dependency edges in the format of *(predicate, argument, relation, number)*, where predicate is the governor and argument the dependent in each dependency edge.[7] Following [130], each potential SP pair is measured by a posterior probability

$$P_r(a|p) = \frac{\mathrm{Count}_r(p, a)}{\mathrm{Count}_r(p)}, \tag{10.9}$$

where $\mathrm{Count}_r(p)$ and $\mathrm{Count}_r(p, a)$ refer to how many times p and the predicate-argument pair (p, a) appear in the relation r, respectively. In our experiment, if $P_r(a|p) > 0.1$ and $\mathrm{Count}_r(p, a) > 10$, we consider the triplet (p, r, a) (e.g., *("dog," nsubj, "barks")*) a valid SP relation. Finally, we select two SP relations, *nsubj* and *dobj*, to form the SP knowledge graph, including 17,074 and 4,536 frequent predicate-argument pairs for *nsubj* and *dobj*, respectively.

10.3.3 Baselines

We compare with the following baselines, including three widely used pre-trained models:

- **Deterministic** model [123], which is an unsupervised model and leverages manual rules to detect coreferences.
- **Statistical** model [18], which is a supervised model and trained on manually crafted entity-level features between clusters and mentions.
- **Deep-RL** model [19], which uses reinforcement learning to directly optimize the coreference matrix instead of the loss function of supervised learning.

The above models are included in the Stanford CoreNLP toolkit.[8] We also include a state-of-the-art end-to-end neural model as one of our baselines:

- **End2end** [70], which is the current state-of-the-art model performing in an end-to-end manner and leverages both contextual information and a pre-trained language model [114].

[6]https://dumps.wikimedia.org/enwiki/

[7]In the Stanford parser, an "nsubj" edge is created between its predictive and subject when a verb is a linking verb (e.g., am, is); the predicative is thus treated as the predicate for the subject (argument).

[8]https://stanfordnlp.github.io/CoreNLP/coref.html

We use their released code.[9] In addition, to show the importance of incorporating knowledge, we also experiment with two variations of our model:

- **Without KG** removes the KG component and keeps all other components in the same setting as that in our complete model.
- **Without Attention** removes the knowledge attention module and concatenates all the knowledge embeddings. All other components are identical as our complete model.

10.3.4 Implementation

Following the previous work [70], we use the concatenation of the 300d GloVe embeddings [113] and the ELMo [114] embeddings as the initial word representations for computing span representations. For knowledge triplets, we use the GloVe embeddings to encode tail words in them. Out-of-vocabulary words are initialized with zero vectors. The hidden state of the LSTM module is set to 200, and all the feed-forward networks have two 150-dimension hidden layers. The selection thresholds are set to 10^{-2} and 10^{-8} for the CoNLL and i2b2 dataset, respectively.

For model training, we use cross-entropy as the loss function and Adam [64] as the optimizer. All the aforementioned hyper-parameters are initialized randomly, and we apply dropout rate 0.2 to all hidden layers in the model. For the CoNLL dataset, the model training is performed with up to 100 epochs, and the best one is selected based on its performance on the development set. For the i2b2 dataset, because no dev set is provided, we train the model up to 100 epochs and use the final converged one.

10.3.5 Results

Table 10.2 reports the performance of all models, with the results for CoNLL and i2b2 in (a) and (b), respectively. Overall, our model outperforms all baselines on two datasets with respect to all pronoun types. There are several interesting observations. In general, the i2b2 dataset seems simpler than the CoNLL dataset, which might because that i2b2 only involves clinical narratives and its training data is highly similar to the test data. As a result, all neural models perform dramatically good, especially on the third personal and possessive pronouns. In addition, we also notice that

[9]https://github.com/kentonl/e2e-coref

(a) CoNLL

Model	Third Personal			Possessive			Demonstrative			All		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Deterministic	25.5	58.9	35.6	22.9	64.3	33.8	3.4	5.7	4.2	23.4	57.0	33.4
Statistical	25.8	62.1	36.5	28.9	64.9	40.0	9.8	6.3	7.6	25.4	59.3	36.5
Deep-RL	78.6	63.9	70.5	73.3	68.9	71.0	3.7	2.9	5.5	76.4	61.2	68.0
End2end	70.6	75.7	73.1	73.0	76.2	74.6	58.4	17.6	27.0	71.1	72.1	71.6
Without KG	78.2	72.4	75.2	80.0	66.4	72.6	46.7	62.5	53.4	75.7	70.1	72.8
Without Attention	76.6	77.9	77.2	79.0	73.5	76.2	42.4	72.6	53.5	73.6	76.4	74.9
Our Complete Model	78.8	77.8	**78.1**	80.7	72.5	**76.4**	45.3	66.7	**53.9**	75.9	75.6	**75.7**

(b) i2b2

Model	Third Personal			Possessive			Demonstrative			All		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Deterministic	25.7	57.4	35.5	25.2	61.6	35.7	6.6	4.0	5.0	25.1	54.0	34.3
Statistical	19.3	35.9	25.1	25.7	50.5	34.0	6.7	4.5	5.4	20.5	36.6	26.3
Deep-RL	78.2	48.0	59.5	78.6	57.7	66.5	9.1	5.1	9.6	77.8	46.3	58.1
End2end	95.0	93.4	94.2	95.3	96.0	95.7	74.8	52.5	61.7	93.9	90.7	92.3
Without KG	96.8	95.9	96.3	97.1	97.5	97.3	66.5	68.2	67.3	94.3	94.0	94.2
Without Attention	96.1	97.2	96.6	96.3	98.2	97.2	66.7	77.8	71.8	93.4	95.9	94.6
Our Complete Model	97.5	96.3	**96.9**	98.5	97.8	**98.2**	71.9	72.2	**72.0**	95.6	94.7	**95.2**

Table 10.2: The performance of pronoun coreference resolution with different models on two evaluation datasets. Precision (P), recall (R), and the F1 score are reported, with the best one in each F1 column marked as bold.

it is more challenging for all models to resolve demonstrative pronouns (e.g., *this*, *that*) on both datasets, because such pronouns may refer to complex things and occur with low frequency.

Moreover, there are significant gaps in the performance of different models, with the following observations. First, models with manually defined rules or features, which cannot cover rich contextual information, perform poorly. In contrast, deep learning models (e.g., End2end and our proposed models), which leverage text representations for context, outperform other approaches by a great margin, especially on the recall. Second, adding knowledge in an appropriate manner within neural models is helpful, which is supported by that our model outperforms the End2end model and the Without KG one on both datasets, especially CoNLL, where the external knowledge plays a more important role. Third, the knowledge attention module ensures our model to predict more precisely, which also results in the overall improvement on F1. To summarize, the results

	CoNLL		i2b2	
	F1	Δ F1	F1	Δ F1
The Complete Model	75.7	-	95.2	-
–OMCS	74.8	-0.9	95.1	-0.1
–Medical-KG	74.5	-1.2	94.6	-0.6
–Ling	73.8	-1.9	94.9	-0.3
–SP	74.0	-1.7	94.7	-0.5

Table 10.3: The performance of our model with removing different knowledge resources. The F1 of each case and the difference of F1 between each case and the complete model are reported.

suggest that external knowledge is important for effectively resolving pronoun coreference, where rich contextual information determines the appropriate knowledge with a well-designed module.

10.4 Analysis

Further analysis is conducted in this section regarding the effect of different knowledge resources, model components, and settings. Details are illustrated as follows.

10.4.1 Ablation Study

We ablate different knowledge for their contributions in our model, with the results reported in Table 10.3. It is observed that all knowledge resources contribute to the final success of our model, where different knowledge types play their unique roles in different datasets. For example, the Ling knowledge contributes the most to the CoNLL dataset while the medical knowledge is the most important one for the medical data.

10.4.2 Effect of the Selection Threshold

We experiment with different thresholds t for the softmax selection. The effects of t against overall performance are shown in Figure 10.4. In general, with the increase of t, fewer candidates are selected. Therefore, the overall precision increases and the recall drops. Consider that both the precision and recall are important for resolving pronoun coreference, we select different thresholds for different datasets to ensure the balance between precision and recall. In detail, for the CoNLL

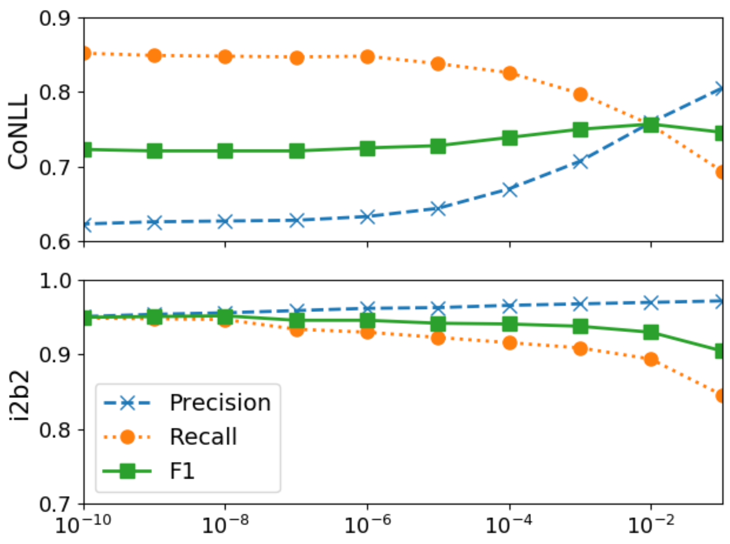


Figure 10.4: Effect of different softmax selection thresholds with respect to our model performance on two datasets. In general, with the threshold becoming larger, less candidates are selected, the precision thus increases while the recall drops.

Model	Setting	CoNLL	i2b2
End2end	Original	71.6	92.3
	+ Gold mention	77.8	94.4
Our Model	Original	75.7	95.2
	+ Gold mention	**80.7**	**96.0**
[183]	+ Gold mention	79.9	-

Table 10.4: Influence of gold mentions. F1 scores on different test sets are reported. Adding human-annotated gold mentions help both the End2end and our model. Best performed model are indicated with the bold font.

dataset, we set $r = 10^{-2}$ to select the most confident predictions; and for the i2b2 dataset, we set $r = 10^{-8}$ so as to keep more predictions.

10.4.3 Effect of Gold Mentions

The effect of adding gold mentions is shown in Table 10.4. Providing gold mentions to the End2end model can significantly boost its performance by 6.2 F1 and 2.1 F1 on the CoNLL and i2b2 dataset, respectively. Yet, the performance gain from gold mentions is less for our model. Such results clearly illustrate that our model is able to benefit the mention detection with the help of KG incorporation. Besides that, with the help of gold mentions, our model achieves the comparable (slightly

Model	Training data	Test data CoNLL	Test data i2b2
End2end	CoNLL	71.6	75.2
	i2b2	20.0	92.3
Our Model	CoNLL	75.7	80.9
	i2b2	42.7	95.2

Table 10.5: Cross-domain performance of different models. F1 on the target domain test sets are reported.

better) performance with the context-and-knowledge model [183]. As their features are originally designed for CoNLL, we only report the performance on CoNLL in Table 10.4. As we also included one new challenging pronoun type, the demonstrative pronoun, the overall performance of their model is lower than the one reported in the original paper. The reason of our model being better is that more knowledge resources (e.g., OMCS) can be incorporated into our model due to its generalizable design. Moreover, it is more difficult for their method [183] to incorporate mention detection into the model, because in this case we need to enumerate all mention spans and generate corresponding features for all spans. which is expensive and difficult to acquire.

10.4.4 Cross-domain Evaluation

Considering that neural models are intensive data-driven and normally restricted by data nature, they are not easily applied in a cross-domain setting. However, if a model is required to perform in real applications, it has to show promising performance on those cases out of the training data. Herein we investigate the performance of different models with training and testing in different domains, with the results reported in Table 10.5. Overall, all models perform significantly worse if they are used cross domains. Specifically, if we train the End2end model on the CoNLL dataset and test it on the i2b2 dataset, it only achieves 75.2 F1. As a comparison, our model can achieve 80.9 F1 in the same case. This observation confirms the capability of knowledge where our model is able to handle. A similar observation is also drawn for the reversed case. However, even though our model outperforms the End2end model by 22.7 F1 from i2b2 to CoNLL, its overall performance is still poor, which might be explained by that the i2b2 is an in-domain dataset and the knowledge contained in its training data is rarely useful for the general (news) domain dataset. Nevertheless, this experiment shows that the generalization ability of deep models is still crucial for building a

	Example A	Example B
Sentence	He walks into the room with one magazine and drops **it** on the couch.	... A small area of erythema around his arm ... **This** will be treated empirically.
Prediction	magazine	erythema
Knowledge	("magazine," *dobj*, "drop")	("erythema," *IsA*, "disease")

Table 10.6: The case study on two examples from the test data, i.e., A: from the CoNLL and B: from the i2b2. Pronouns and correct mentions are marked by red bold and blue underline font respectively. Knowledge triplets used for them are listed in the bottom row.

successful coreference model, and learns to use knowledge is a promising solution to it.

10.5 Case Study

To better illustrate the effectiveness of incorporating different knowledge in this task, two examples are provided for the case study in Table 10.6. In example A, our model correctly predicts that "*it*" refers to the "*magazine*" rather than the "*room*," because we successfully retrieve the knowledge that compared with the "*room*," the "*magazine*" is more likely to be the object of *drop*. In example B, even though the distance between "erythema" and "This" is relatively far,[10] our model is able to determine the coreference relation between them because it successfully finds out that "erythema" is a kind of disease, while a lot of diseases appear as the context of "be treated" in the training data.

[10]We omit the intermediate part of the long sentence in the table for a clear presentation.

CHAPTER 11

EXPLORING A GENERAL COMMONSENSE INFERENCE FRAMEWORK

In previous chapters, we have discussed potential solutions for applying structured knowledge for a fundamental natural language understanding task that requires commonsense reasoning, pronoun coreference resolution. Besides the pronoun coreference resolution, many commonsense reasoning datasets [133, 159, 179, 80] have been proposed to encourage research on commonsense reasoning. Even though they may target different formats, knowledge types, or modalities, they often follow a standard supervised learning setting, and aim at helping machines to solve a specific commonsense task with the training data. Recently, with the help of pre-trained language models (e.g., BERT [25]) and large-scale annotation, the community has made impressive progress on these tasks. However, models do not generalize well across tasks [61].

In this chapter, considering that there could be enormous kinds of commonsense reasoning tasks and it is infeasible to annotate a huge dataset for each of them, we are interested in exploring a universal solution for all of them at the same time. Specifically, based on the assumption that even though different tasks may have different formats, they share the limited kinds of inference over the commonsense knowledge, we propose to convert different commonsense tasks into a unified format (i.e., question answering). For each of the question-answer pair, we equip it with a small supporting commonsense knowledge graph. On top of that, we can learn the correct inference over commonsense rather than acquire the new commonsense. Specifically, for each question-answer pair, we plan to first extract the key eventualities out of it and then find a sub-graph of an external commonsense knowledge base that covers the relevant knowledge.[1] After that, we will conduct human annotation to verify whether the auto-extracted knowledge can be used to answer this question or not. We name this task as commonsense knowledge based question answering (CKBQA). An example is shown in Figure 11.1, in which we denote key eventualities from the question with purple and orange boxes and the correct inference path over the graph with the yellow

[1]Currently, as no existing complete commonsense knowledge graph that has the perfect coverage, we only keep questions that we can find the relevant commonsense knowledge.

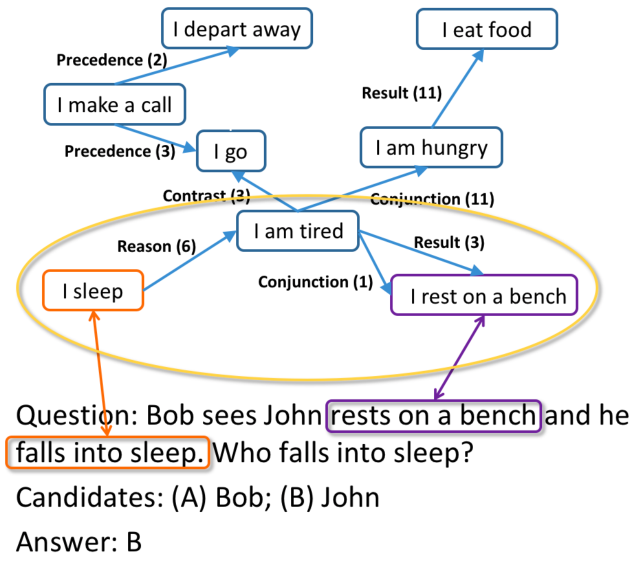


Figure 11.1: CKBQA demonstration.

oval. So, in CKBQA, we expect the model to learn to resolve "he" to "John" rather than "Bob" by knowing that "sleep" and "rest on a bench" can both be caused by "being tired" and thus it is more likely for the subject of these two events to be the same person. Experiments on CKBQA show that even though inference over commonsense knowledge is challenging, models can learn to conduct simple inference after training with a few examples. Another observation is that the learned model demonstrates the generalization ability across tasks, which was not observed in previous commonsense reasoning models.

11.1 Problem Overview

In CKBQA, given a question Q, a supporting knowledge graph G, and two candidate answers A_1 and A_2, we are interested in two tasks: (1) Whether the model can predict the correct answer? (2) Whether the model can predict "I do not know (IDK)" when G cannot help solve Q correctly?

11.2 CKBQA

In this section, we introduce the preparation details of CKBQA, including the commonsense knowledge base (KB) selection, the commonsense task selection, format unification, and supporting knowledge graph extraction.

	ConceptNet	ATOMIC	ASER
Scale	middle	middle	large
Quality	high	high	middle
Relation Richness	middle	small	large
Relation Weight	binary	binary	weighted

Table 11.1: Comparison of different commonsense knowledge bases from four perspectives: Scale, Quality, Relation Richness, and Relation Weight.

11.2.1 Commonsense KB Selection

As aforementioned, different commonsense KBs have different properties and strengths. For example, ConceptNet [84] leverages the human annotation to collect 600 thousand pieces of commonsense about 34 human-defined relations (e.g., "Cat"-HasA-"Tail" and "Eat"-HasPrerequisite-"Being hungry"). Similar to ConceptNet, ATOMIC [139] also leverages human annotation to collect commonsense knowledge about daily events. In total, ATOMIC contains 310 thousand triplets about 9 different event relations (e.g., "Be caused by" and "Result in"). As a comparison, ASER is automatically extracted and contains commonsense knowledge about 194 million eventualities. The nodes in ASER are eventualities, and edges are weighted discourse relations (e.g., "I am tired"→*Result (3)*→"I rest on a bench"). The weights in brackets reflect humans' selectional preference over daily eventualities.

A detailed comparison of aforementioned commonsense KBs are presented in Table 11.1. We choose ASER as the supporting knowledge graph for the following reasons. First, commonsense is more complex than either the 34 human-defined relations in ConceptNet or the 9 relations in ATOMIC. As a comparison, ASER uses a linguistic relation based representation format, which can cover richer commonsense relations. As discussed in Chapter 5, the selectional preference over linguistic relations stored in ASER can effectively cover knowledge in other commonsense KBs (i.e., ConceptNet). Second, commonsense is a kind of preference rather than fixed factorial knowledge [60, 168, 131]. For example, compared with "being tasty," "being hungry" is more likely to be the cause of eating, but not every time one feels hungry, one will eat. The weighted representation format of ASER can naturally be used to reflect such preference.

11.2.2 Commonsense Task Selection

We select the following four commonsense reasoning tasks based on two consideration: (1) We want to cover more diverse formats; (2) The selected dataset should be of high quality.

1. HardPCR: The hard pronoun coreference resolution (HardPCR) task is one of the most famous commonsense reasoning tasks. For each question, a target pronoun and two candidate mentions are provided and the task is to select the correct mention that the pronoun refers to. Careful expert annotations were conducted to get rid of the influence of all simple linguistic rules and ask the models to solve the problem with commonsense reasoning. In CKBQA, we include instances from WSC [72], DPR [126], and WinoGrande [136].

2. CommonsenseQA [159]: CommonsenseQA is a commonsense question answering dataset. For each question-answer pair, four relevant but wrong concepts are used as the other candidates, and the models are required to select the correct one out of five candidates. In CKBQA, we randomly sample a negative answer to make it a binary choice task, which is consistent with other datasets.

3. COPA [133]: COPA focuses on evaluating whether models can understand the causality between events or not. For each head event, two candidate tail events are provided, and models are asked to predict the one caused by or the reason of the head event.

4. ATOMIC [139]: The last one is the commonsense knowledge base completion, which aims at predicting the tail node given head and relations to populate it. In CKBQA, We focus on predicting edges of ATOMIC.

11.2.3 Format Unification

After selecting the base commonsense datasets, motivated by recent works [63, 20, 171, 27], we then unify all selected questions into the QA format. For the hard pronoun coreference resolution task (i.e., WSC, DPR, and Winogrande), as the original dataset has already provided two candidates, all we need to do is automatically creating a question regarding the target pronoun. Specifically, we first extract the eventuality that contains the target pronoun and then determine whether the pronoun refers to a person or an object. If it is a person, we will ask who participates

Task Name	Original Assertion	Transformed Question	Answer
Hard PCR	The fish ate the worm. It was hungry.	The fish ate the worm. It was hungry. What was hungry?	(A) Fish; (B) Worm
CommonsenesQA	What is a place that someone can go buy a teddy bear?	What is a place that someone can go buy a teddy bear?	(A) Toy store; (B) Shelf
COPA	I drank from the water fountain.	I drank from the water fountain. What was the cause of this?	(A) I was thirsty.; (B) I felt nauseous.
ATOMIC	PersonX buys the bike.	Before PersonX buys the bike, what did PersonX want?	(A) To be social.; (B) To have transportation.

Table 11.2: Demonstration of the original assertion, transformed questions, and answers. Correct and wrong answers are indicated with blue and red, respectively. More examples are in the Appendix.

in that event. Otherwise, we will ask what participates in that event. For CommonsenseQA, as it is already in the question-answering format, we do not need to change its format. For COPA and ATOMIC, as they are essentially predicting the relations between two eventualities (e.g., "PersonX eats"-*Causes*-"PersonX is full"), we randomly sample a tail eventuality (e.g., "PersonX is hungry") as the negative sample and ask the model to select the correct one. For each type of relation, we write a simple pattern to generate the question. For example, for the "Causes" relation, we will ask "What can be caused by 'PersonX is hungry'?" Demonstrations of instances in original datasets and their transformed questions and candidate answers are presented in Table 11.2.

11.2.4 Supporting Graph Extraction

For each question, we need to obtain a sub-graph from ASER such that it contains the relevant commonsense knowledge about this question. The sub-graph extraction process includes the following three steps: Data Pre-processing, Eventuality Matching, and Graph Extraction.

Data Pre-processing: For each question and the associated candidate answers, to find the relevant knowledge about both the questions and answers, we first try to replace the question words (e.g., "What") with the two candidate answers such that it becomes two declarative sentences. For instance, if the question is "The fish ate the worm. It was hungry. Who is hungry?" and the candidates are "Fish" and "Worm", we will convert the question into two declarative sentences: "Fish is hungry" and "Worm is hungry." After that, to better align with ASER, we leverage the eventuality extraction tool released by ASER [185] to extract eventualities out of those converted declarative sentences. After the pre-processing, we will get a set of relevant eventualities for each QA pair (i.e., "Fish ate the worm," "Fish is hungry," and "Worm is hungry").

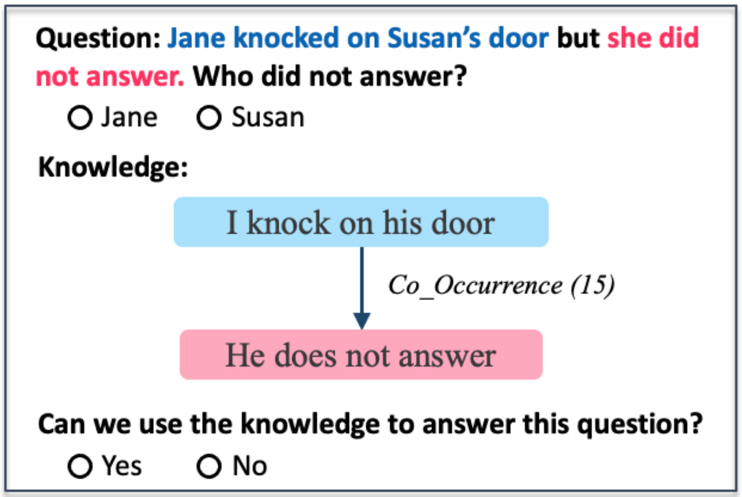
Eventualities Matching: After extracting key eventualities out of the questions, we then map them to nodes in ASER. Considering that each eventuality may have multiple words and it is often hard to find an exact match, we adopt a soft matching technique. For each extracted eventuality and node in ASER, we treat them as a sentence and get the corresponding representations with Sentence-BERT [127]. For each input sentence, Sentence-BERT will return a vector to represent its overall semantics in the Euclidean space. Closer distance between two vectors indicates that the two sentences are more similar to each other. In our current pipeline, we use the cosine similarity of the Sentence-BERT representations to indicate the similarity between two eventualities.[2] Since there are 194 million eventualities in ASER, it is infeasible to compute the cosine similarity between eventualities pair by pair. Thus for each extracted eventuality, we first apply Faiss [58], which is a large-scale similarity-based matching algorithm that first clusters all ASER eventualities in the vector space to increase the matching efficiency, to find the top 60 eventualities in ASER. After that, we can then sort the found 60 eventualities based on the cosine similarity to find the top K similar eventualities. We set K to be 1 for most datasets while 5 for COPA. This is mainly because of the small size of COPA. If we set K to be 1 for COPA, we will get very few examples left. On average, it takes 25 seconds to retrieve relevant eventualities for each question.

Graph Extraction: In the last step, we construct the sub-graph. We denote the extracted m eventualities as $e_1, e_2, ..., e_m$, and for each of them we find K similar eventualities from ASER. The resulting matched eventuality sets are denoted as $\mathcal{E}_1, \mathcal{E}_2, ..., \mathcal{E}_m$. For any pair of eventualities $e \in \mathcal{E}_i$ and $e' \in \mathcal{E}_j$, if there exist a one-hop or two-hop connection in ASER between e and e' and $i \neq j$, we will keep that path. After merging all paths together, we will get the final sub-graph. On average, it takes less than two seconds for us to construct such a graph for each question.

11.3 Annotation

In this section, we introduce the annotation details.

[2]We also tried other matching techniques such as string match, ROUGE [81], and BLEURT [144]. They are either not accurate or too slow for our scale.



Question: Jane knocked on Susan's door but she did not answer. **Who did not answer?**

○ Jane ○ Susan

Knowledge:

Can we use the knowledge to answer this question?

○ Yes ○ No

Figure 11.2: An example of the used survey.

11.3.1 Survey Design

The annotation goal is to determine whether the supporting graph can help answer the question or not. Thus, for each QA pair, we present the question, candidate answers, and the supporting sub-graph to annotators, and then ask them two questions: (1) What is the correct answer for this question; (2) Whether the provided commonsense knowledge contains all the essential commonsense for answering this question. The purpose of the first question is to assess the annotation quality. A survey example is shown in Figure 11.2. In each survey, we also provide detailed instructions and examples to help annotators understand our task.

11.3.2 Annotation Details

We first employ annotators from Amazon Mechanical Turk to provide annotations. To improve the annotation quality, we require the annotators to be English native speaker and to have an overall acceptance rate above 90%. For each question, we invite five annotators to provide the annotations. Considering that our task is relatively challenging for laymen because it does not only involve multiple kinds of commonsense tasks but also involves finding the essential inference path over a graph, we also invite domain experts to annotate the same survey. Specifically, we invited 30 students to provide the annotation. After the careful instruction, they can clearly understand the task and requirements. As a result, they can provide higher quality annotation. For each QA pair, we randomly assign it to three different students. To compare these annotation approaches, we show

	Crowd-sourcing	Experts
Accuracy on Q1	0.65	0.90
IAA on Q1	0.56	0.91
IAA on Q2	0.63	0.87

Table 11.3: Annotation Quality. Q1 and Q2 indicate the original question and the question asking whether the knowledge in the sub-graph is enough or not. IAA is the inter-annotator agreement.

Task Name	# Instances	Average Sub-graph Size (# Edges)	K	# Helpful Instances
Hard PCR	2,030	3.03	1	185
CommonsenseQA	530	3.30	1	47
COPA	103	2.73	5	27
ATOMIC	5,655	2.77	1	914
Total	8,318	2.87	-	1,173

Table 11.4: CKBQA statistics.

the average accuracy on the first question as well as the inter-annotator agreement on both questions in Table 11.3. The results show that experts have significantly higher accuracy and agreement than crowd-sourcing workers. This makes sense because unlike an ordinary crowd-sourcing task, our task requires complex thinking and is more challenging for ordinary people. Considering this, we choose to use the expert annotation as the final label.

11.3.3 CKBQA Statistics

We report the number of total instances, average supporting graph size, and the number of helpful instances of CKBQA in Table 11.4. In total, we collect 8,318 instances, and among which Hard PCR and ATOMIC provide the most questions because their original datasets are much larger than others. According to the annotation, 14.10% of the supporting knowledge graphs can be used to answer the questions. Based on our analysis, annotators hold a very strict standard for whether the supporting knowledge is helpful or not. More detailed statistics and case studies are in the appendix. For each task, we randomly split the dataset into training, development, and testing set with a standard 8:1:1 splitting. As a result, we get 6,656 training, 831 development, and 831 testing instances.

11.4 Experiments

In this section, we conduct experiments with the language model (LM) based multiple-choice (MC) model. For each candidate answer, we concatenate it with the question and then feed it to a pre-trained language model. After getting the sentence representation, a single layer neural network is applied to predict the plausibility score. Candidates with the higher score will be selected. Specifically, we are interested in the following four settings:

1. **Without Knowledge**: For the baseline model, we follow the simplest multiple-choice setting and use the question and answer as the input.

2. **With Knowledge**: To use the knowledge, we propose to convert the supporting KG into paragraphs and use them as the additional context. As all edges in the graph are discourse relations, we can convert each edge into a sentence (e.g., "I eat food because I am hungry (11)") by replacing the discourse relation with the most representative connectives[3] (e.g., "because") and putting the edge weight into brackets. And then we can randomly concatenate sentences together to be a paragraph. In this setting, we concatenate the knowledge, question, and candidates as the input.

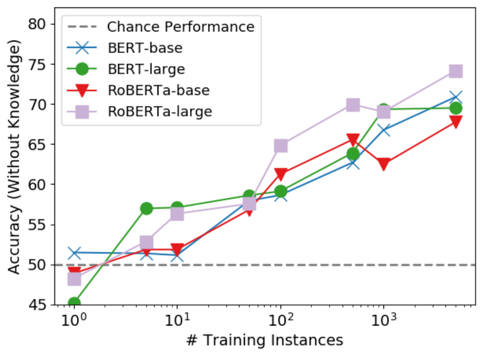
3. **Answer Only**: To study the artifact in CKBQA, we add an "Answer Only" setting, where only candidates are fed into models. Ideally, models should get the chance performance.

4. **IDK**: To study if the model can learn to distinguish whether the provided knowledge is helpful or not, we add an "IDK" experiment. For each question, we add an additional candidate "I do not know (IDK)" and ask the model to predict if the knowledge is labeled as "not helpful." We only report the performance whether the model could know the knowledge is helpful or not, rather than the original task. In this setting, we randomly select the same number of "Not Helpful" examples as the "Helpful" ones to make the dataset balanced.

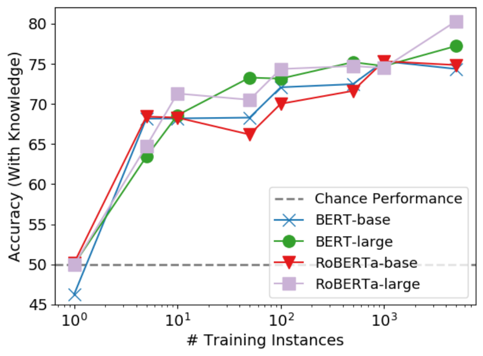
We implement the experiments with Huggingface [170] and select BERT [25] and RoBERTa [85] as the language models. We use Cross-Entropy as the loss function and Adam [64] as the optimization method. All models are trained for 10,000 steps[4] and the best-performing checkpoints on the dev set are evaluated. We tried training with different training data scales to show the learning curves of different settings. All other hyper-parameters are the same as the base LMs.

[3]All selected connectives are in the appendix.

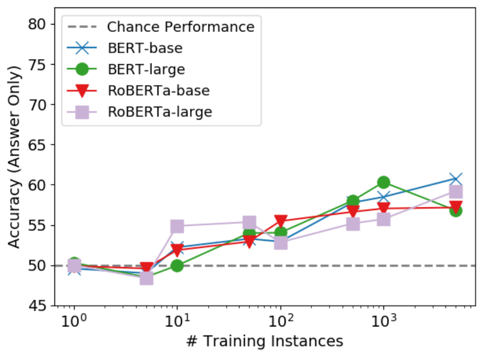
[4]All models converge at 10,000 steps.



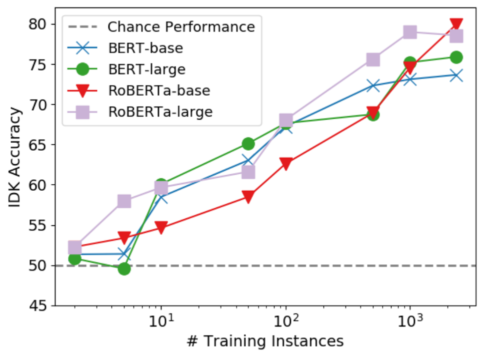
(a) Without Knowledge (Questions, Answers)



(b) With Knowledge (Questions, Knowledge, Answers)



(c) Answer Only (Answers)



(d) IDK (Questions, Knowledge, Answers)

Figure 11.3: Learning curves of different experiment settings. The model input under different settings are in brackets.

11.5 Results Analysis

Experimental results are shown in Figure 11.3, from which we can make the following observations:

1. Comparing Figure 11.3 (a) and (b), we can find out that even though the auto-extracted knowledge may be annotated as "not helpful," they are still helpful. This is probably because some knowledge graphs are annotated as "not helpful" because the annotators think that they are not sufficient enough. As a result, it is possible that some knowledge graphs is not sufficient enough to solve the problem by itself. But if we combine it with the knowledge from the pre-trained LMs, we can solve the problem.

2. The "With Knowledge" model can achieve good performance after seeing a small number of

(a) Without Knowledge

Training Task	Testing Task			
	Hard PCR (151 / 1,624)	CommonsenseQA (40 / 424)	COPA (22 / 83)	ATOMIC (712 / 4,525)
Hard PCR (21 / 203)	56.29 / 50.31	40.00 / 52.83	54.55 / 45.78	39.75 / 47.89
CommonsenseQA (4 / 53)	56.95 / 48.03	70.00 / 52.35	36.36 / 50.60	52.95 / 46.34
COPA (4 / 10)	45.03 / 49.82	45.00 / 50.47	59.09 / 46.99	55.62 / 54.43
ATOMIC (90 / 565)	40.40 / 49.94	65.00 / 56.72	59.09 / 56.63	72.75 / 75.99

(b) With Knowledge

Training Task	Testing Task			
	Hard PCR (151 / 1,624)	CommonsenseQA (40 / 424)	COPA (22 / 83)	ATOMIC (712 / 4,525)
Hard PCR (21 / 203)	55.63 / 50.98	57.50 / 40.57	59.09 / 49.40	48.74 / 48.62
CommonsenseQA (4 / 53)	51.66 / 49.38	95.00 / 42.92	68.18 / 59.04	95.36 / 69.70
COPA (4 / 10)	45.69 / 48.28	72.50 / 60.61	86.36 / 67.47	87.64 / 74.01
ATOMIC (90 / 565)	45.03 / 51.23	92.50 / 69.34	95.45 / 78.31	98.03 / 83.14

Table 11.5: Generalization ability demonstration. We report the performance on both the clean dataset (i.e., only questions with helpful knowledge are selected for training and testing) and full dataset to show the generalization ability before and after the slash, respectively. Number of training and testing instances are in brackets.

examples. This suggests that if we only want to learn to do the inference over commonsense, we may only need a few training examples.

3. As shown by the "Answer Only" experiment, CKBQA also inherits the artifact from previous datasets. As a common phenomenon in human-crafted commonsense reasoning datasets, such artifact significantly influences the reliability of the learned models. Based on our experiment results, even though it is hard to fully get rid of the artifact, we can minimize its influence by controlling the training data size, which suggests that the few-shot learning setting might be the more fair setting for both CKBQA and other commonsense reasoning datasets.

4. Answering "I do not know" is a reasonable task if enough training data is provided. However, it is still challenging for our current naive model to distinguish whether the knowledge is helpful or not under the few-shot learning setting.

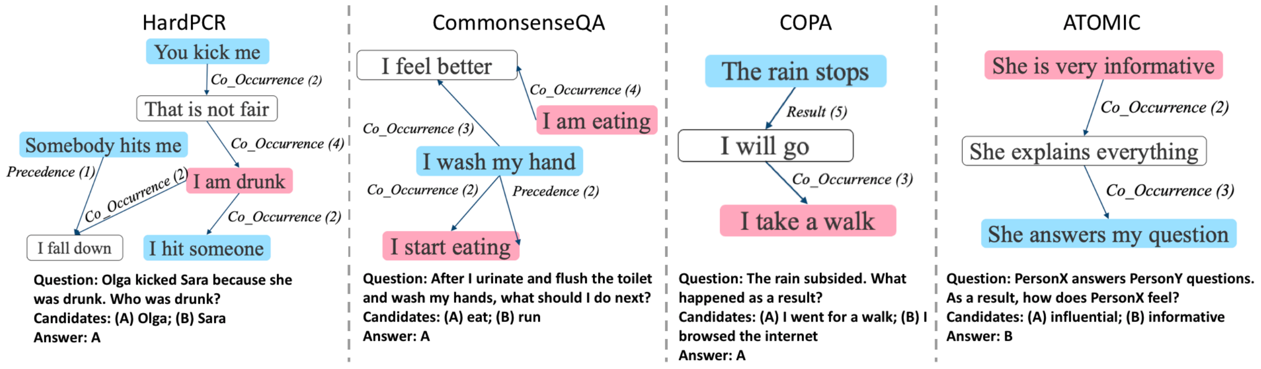


Figure 11.4: CKBQA Case Study. Mapped eventualities for the question and answers are indicated with blue and pink. Other eventualities are white. Edge weights are in brackets. We only show the relevant part of the graph for the clear representation. All extracted eventualities are lemmatized, we recover them for the ease of understanding.

11.6 Generalization Ability

An important assumption and motivation behind CKBQA is that even though the commonsense could be enormous, the inference rules over commonsense knowledge should be limited. As a result, even though we could not learn all the commonsense from limited training data, we can learn how to conduct inference with several tasks and then generalize to others. In this section, we conduct experiments with both the "Without Knowledge" and "With Knowledge" models to show that with our unified formulation, we can gain such generalization ability across different tasks. Specifically, to minimize the influence of the dataset artifacts, we follow the previous observation and conduct our experiments with a few-shot learning setting (i.e., we randomly selected 10% of the data for training, 10% for development, and 80% for testing). Also, to clearly show the effect of the supporting commonsense KB, we conduct experiments on two settings: (1) Helpful Subset: We only train and test the model on questions, where the supporting graph is annotated as helpful; (2) Full Set: We train and test the model with the whole dataset. We try to train the model with questions from a specific task and test it on all tasks. The results are presented in Table 11.5. Strong and moderate generalization settings are indicated with the green and orange background, respectively.

From the results, we can see that the knowledge can help models to generalize well among CommonsenseQA, COPA, and ATOMIC. The only exception is HardPCR. This is mainly because the inference needed for solving HardPCR is more complex than the other tasks. As shown in Figure 11.4. For the HardPCR example, two paths can be found relevant to question: (1) "I am drunk”→*Co_Occurrence*→“I hit someone”; (2) “I am drunk”→*Co_Occurrence*→“That is not fair”→*Co_Occurrence*→“You kick me”. For the correct inference, we need to know when there is a conflict, we should trust the one-hop inference more because the additional node in the two-hop path may introduce extra noise. As a comparison, for other tasks, the main inference we need is to find the relevant paths, which is relatively easy. In general, the observed generalization ability is encouraging because if we can learn a good model on CKBQA, based on the assumption that there exists limited types of inference, potentially we can solve any commonsense reasoning tasks as long as the needed inference type is covered by CKBQA . At the same time, we also notice that current models still cannot learn complex inference (i.e., compare multiple paths) with few examples, and we leave how to solve that problem as the future work.

CHAPTER 12

CONCLUSION AND REMAINING CHALLENGES

In this thesis, we focus on the commonsense reasoning task. To propose a robust and explainable commonsense reasoning system, we first leverage the Winograd Schema Challenge [72] to conduct an in-depth diagnosis of current data-driven commonsense reasoning models. Experimental results show that even though current deep models can achieve high performance on standard benchmarks, they still cannot fully understand the reason behind the predictions. An important reason behind this is that we do not have a clear understanding of commonsense and how we should represent them. As a result, we can only treat the commonsense as a black box.

To shed some light on this challenging question, we propose to understand commonsense from the angle of selectional preference and represent commonsense knowledge with higher-order selectional preference over eventualities. Following this methodology, we develop a large-scale eventuality-centric commonsense knowledge graph ASER. Extensive evaluations are conducted to demonstrate the quantity and quality of ASER. After that, to demonstrate the collected knowledge in ASER is indeed commonsense, we conduct experiments to show the transferability from the selectional preference knowledge in ASER to human-defined commonsense knowledge.

As it is infeasible for a single modality like text to cover all the commonsense, we also explored how we can leverage other modalities to better acquire the selectional preference-based commonsense knowledge. Specifically, we first propose a multiplex word embedding model to capture selectional preference knowledge. After that, we introduce how to utilize the analogous property of eventualities and the support of other knowledge bases (i.e., WordNet) to generalize the knowledge about observed eventualities to unseen ones. Last but not least, we investigate the possibility of acquiring causal knowledge about daily eventualities from the time-consecutive images.

In the last part of this thesis, we also try to apply the structured commonsense for downstream natural language understanding tasks. We first investigate how to jointly use the structured knowledge and language representation models for better pronoun coreference resolution. After that, we introduce a new commonsense inference learning paradigm: Commonsense knowledge base question answering (CKBQA). Experiments on CKBQA show that with our formulation, even though inference over commonsense knowledge is challenging, models can learn to conduct simple inference after training with a few examples. Besides that, the learned model also demonstrates the encouraging generalization ability across tasks, which was not observed in previous commonsense reasoning models.

Even though this thesis has answered several questions about how we should represent, acquire, and apply the commonsense knowledge, there is still a long way to go for us to fully solve the commonsense reasoning problem, which requires us to solve the following challenges:

1. **Evaluation**: The first challenge we are still facing is the lack of a good evaluation system. Unlike other tasks, most current commonsense reasoning tasks (e.g., Winograd Schema Challenge [72]) are not directly evaluating models' commonsense reasoning abilities. Instead, they are a kind of approximation. Take WSC as an example, many research has discovered that current models can bypass the essential commonsense reasoning and solve the questions with other information [34]. It is quite often that we are just solving a "dataset" without solving the underlining "task" we truly want to solve.

2. **Storage and Computation Efficiency**: As aforementioned, ASER has 438 million eventualities and 648 million edges. Such a large scale guarantees the coverage of ASER, but it also brings a huge burden for storage and computation. Our current hardware architecture and inference algorithms still cannot support fast inference and response. We can try to address this issue from two angles: (1) Better hardware architecture; (2) Better Knowledge graph organization.

3. **Scope of Reasoning**: The last critical challenge we are facing is the unclear scope of commonsense reasoning. For example, is knowledge retrieval counted as part of the reasoning? The lack of a formal definition of the scope of commonsense reasoning could bring huge ambiguity for both the evaluation and model design.

To conclude this thesis, even though commonsense reasoning is still very challenging, it is a task that we must solve to achieve the strong artificial intelligence. I hope that the methodology and techniques we proposed in this thesis could bring us one step closer to the final goal.

References

[1] Jacqueline Aguilar, Charley Beller, Paul McNamee, Benjamin Van Durme, Stephanie Strassel, Zhiyi Song, and Joe Ellis. A comparison of the events and relations across ace, ere, tac-kbp, and framenet annotation standards. In Proceedings of the Second Workshop on EVENTS: Definition, Detection, Coreference, and Representation, EVENTS@ACL 2014, pages 45–53, 2014.

[2] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. DBpedia: A nucleus for a web of open data. Springer, 2007.

[3] Emmon Bach. The algebra of events. Linguistics and philosophy, 9(1):5–16, 1986.

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In Proceedings of ICLR 2015, 2015.

[5] Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The Berkeley FrameNet Project. In Proceedings of COLING-ACL 1998, pages 86–90, 1998.

[6] Michele Banko, Michael J. Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. Open information extraction from the web. In Proceedings of IJCAI 2007, Hyderabad, India, pages 2670–2676, 2007.

[7] Elias Bareinboim and Judea Pearl. Causal inference and the data-fusion problem. Proc. Natl. Acad. Sci. USA, 113(27):7345–7352, 2016.

[8] Jonathan Berant, Vivek Srikumar, Pei-Chun Chen, Abby Vander Linden, Brittany Harding, Brad Huang, Peter Clark, and Christopher D. Manning. Modeling biological processes for reading comprehension. In Proceedings of EMNLP 2014, 2014.

[9] James C Bezdek and Richard J Hathaway. Convergence of alternating optimization. Neural, Parallel & Scientific Computations, 11(4):351–368, 2003.

[10] Kurt D. Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In Proceedings of SIGMOD 2008, pages 1247–1250, 2008.

[11] Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli **c**Celikyilmaz, and Yejin Choi. COMET: commonsense transformers for automatic knowledge graph construction. In Proceedings of the ACL 2019, pages 4762–4779, 2019.

[12] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In Proceedings of NeurIPS 2020, 2020.

[13] Alexander Budanitsky and Graeme Hirst. Evaluating wordnet-based measures of lexical semantic relatedness. Comput. Linguistics, 32(1):13–47, 2006.

[14] Mario Bunge. Causality and modern science. Routledge, 2017.

[15] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell. Toward an architecture for never-ending language learning. In Proceedings of AAAI, pages 1306–1313, 2010.

[16] Eugene Charniak and Micha Elsner. Em works for pronoun anaphora resolution. In Proceedings of EACL 2009, pages 148–156, 2009.

[17] Danqi Chen and Christopher Manning. A fast and accurate dependency parser using neural networks. In Proceedings of EMNLP 2014, pages 740–750, 2014.

[18] Kevin Clark and Christopher D Manning. Entity-centric coreference resolution with model stacking. In Proceedings of ACL-IJCNLP 2015, volume 1, pages 1405–1415, 2015.

[19] Kevin Clark and Christopher D. Manning. Deep reinforcement learning for mention-ranking coreference models. In Proceedings of EMNLP 2016, pages 2256–2262, 2016.

[20] Amir D. N. Cohen, Shachar Rosenman, and Yoav Goldberg. Relation extraction as two-way span-prediction. CoRR, abs/2010.04829, 2020.

[21] Bhavana Dalvi, Lifu Huang, Niket Tandon, Wen tau Yih, and Peter Clark. Tracking state changes in procedural text: A challenge dataset and models for process paragraph comprehension. Proceedings of NAACL-HLT 2018, pages 1595–1604, 2018.

[22] Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José M. F. Moura, Devi Parikh, and Dhruv Batra. Visual dialog. In Proceedings of CVPR 2017, pages 1080–1089, 2017.

[23] Joe Davison, Joshua Feldman, and Alexander M. Rush. Commonsense knowledge mining from pretrained models. In Proceedings of EMNLP-IJCNLP 2019, pages 1173–1178, 2019.

[24] Tim Van de Cruys. A neural network approach to selectional preference acquisition. In Proceedings of EMNLP 2014, pages 26–35, 2014.

[25] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Proceedings of NAACL-HLT 2019, pages 4171–4186, 2019.

[26] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In Proceedings of KDD 2014, pages 601–610, 2014.

[27] Xinya Du and Claire Cardie. Event extraction by answering (almost) natural questions. In Proceedings of EMNLP 2020, pages 671–683, 2020.

[28] Richard J Edens, Helen L Gaylard, Gareth JF Jones, and Adenike M Lam-Adesina. An investigation of broad coverage automatic pronoun resolution for information retrieval. In Proceedings of SIGIR 2003, pages 381–382. ACM, 2003.

[29] Mark Edmonds, Xiaojian Ma, Siyuan Qi, Yixin Zhu, Hongjing Lu, and Song-Chun Zhu. Theory-based causal transfer: Integrating instance-level induction and abstract-level structure learning. In Proceedings of AAAI 2020, pages 1283–1291, 2020.

[30] Kate Ehrlich. Search and inference strategies in pronoun resolution: An experimental study. In Proceedings of ACL 1981, pages 89–93, 1981.

[31] Lisa Ehrlinger and Wolfram Wöß. Towards a definition of knowledge graphs. In Joint Proceedings of the Posters and Demos Track of SEMANTiCS2016 and SuCCESS'16, volume 1695, 2016.

[32] Kiana Ehsani, Hessam Bagherinezhad, Joseph Redmon, Roozbeh Mottaghi, and Ali Farhadi. Who let the dogs out? modeling dog behavior from visual data. In Proceedings of CVPR 2018, pages 4051–4060, 2018.

[33] Yanai Elazar, Abhijit Mahabal, Deepak Ramachandran, Tania Bedrax-Weiss, and Dan Roth. How large are lions? inducing distributions over quantitative attributes. In Proceedings of ACL 2019, pages 3973–3983, 2019.

[34] Yanai Elazar, Hongming Zhang, Yoav Goldberg, and Dan Roth. Back to square one: Bias detection, training and commonsense disentanglement in the winograd schema. CoRR, abs/2104.08161, 2021.

[35] Ali Emami, Noelia De La Cruz, Adam Trischler, Kaheer Suleman, and Jackie Chi Kit Cheung. A knowledge hunting framework for common sense reasoning. In Proceedings of EMNLP 2018, pages 1949–1958, 2018.

[36] Katrin Erk, Sebastian Padó, and Ulrike Padó. A flexible, corpus-driven model of regular and inverse selectional preferences. Computational Linguistics, 36(4):723–763, 2010.

[37] Oren Etzioni, Michael Cafarella, and Doug Downey. Webscale information extraction in knowitall (preliminary results). In Proceedings of WWW 2004, pages 100–110, 2004.

[38] Christiane Fellbaum, editor. WordNet: an electronic lexical database. MIT Press, 1998.

[39] Amy Sue Fire and Song-Chun Zhu. Learning perceptual causality from video. ACM TIST, 7(2):23:1–23:22, 2016.

[40] Goran Glavas, Jan Snajder, Marie-Francine Moens, and Parisa Kordjamshidi. Hieve: A corpus for extracting event hierarchies from news stories. In Proceedings of LREC 2014, pages 3678–3683, 2014.

[41] Alison Gopnik, Clark Glymour, David M Sobel, Laura E Schulz, Tamar Kushnir, and David Danks. A theory of causal learning in children: causal maps and bayes nets. Psychological review, 111(1):3, 2004.

[42] Jonathan Gordon and Lenhart K. Schubert. Quantificational sharpening of commonsense knowledge. In Proceesings of AAAI 2010 Fall Symposium, Arlington, Virginia, USA, 2010.

[43] Jonathan Gordon, Benjamin Van Durme, and Lenhart K. Schubert. Learning from the web: Extracting general world knowledge from noisy text. In Collaboratively-Built Knowledge Sources and Artificial Intelligence, Papers from the 2010 AAAI Workshop, Atlanta, Georgia, USA, 2010.

[44] Mark Granroth-Wilding and Stephen Clark. What happens next? event prediction using a compositional neural network model. In Proceedings of AAAI 2016, pages 2727–2733, 2016.

[45] H Paul Grice. Further notes on logic and conversation. In Pragmatics, pages 113–127. Brill, 1978.

[46] Thomas L Griffiths and Joshua B Tenenbaum. Theory-based causal induction. Psychological review, 116(4):661, 2009.

[47] Chikara Hashimoto, Kentaro Torisawa, Julien Kloetzer, Motoki Sano, István Varga, Jong-Hoon Oh, and Yutaka Kidawara. Toward future scenario generation: Extracting event causality exploiting semantic relation, context, and association features. In Proceedings of ACL 2014, pages 987–997, 2014.

[48] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of CVPR 2016, pages 770–778, 2016.

[49] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In Proceedings of CVPR 2015, pages 961–970, 2015.

[50] Benjamin Heinzerling, Nafise Sadat Moosavi, and Michael Strube. Revisiting selectional preferences for coreference resolution. In Proceedings of EMNLP 2017, pages 1332–1339, 2017.

[51] Christopher Hidey and Kathy McKeown. Identifying causal relations using parallel wikipedia articles. In Proceedings of ACL 2016, 2016.

[52] Felix Hill, Roi Reichart, and Anna Korhonen. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. Computational Linguistics, 41(4):665–695, 2015.

[53] Jerry R Hobbs. Resolving pronoun references. Lingua, 44(4):311–338, 1978.

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

[55] Jena D. Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. COMET-ATOMIC 2020: On symbolic and neural commonsense knowledge graphs. In Proceedings of AAAI 2021, 2021.

[56] Naoya Inoue, Yuichiroh Matsubayashi, Masayuki Ono, Naoaki Okazaki, and Kentaro Inui. Modeling context-sensitive selectional preference with distributed representations. In Proceedings of COLING 2016, pages 2829–2838, 2016.

[57] Ray Jackendoff, editor. Semantic Structures. Cambridge, Massachusetts: MIT Press, 1990.

[58] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. CoRR, abs/1702.08734, 2017.

[59] Richard M. Karp. Reducibility among combinatorial problems. In Proceedings of a symposium on the Complexity of Computer Computations 1972, pages 85–103, 1972.

[60] Jerrold Katz and Jerry Fodor. The structure of a semantic theory. Language, 39:170–210, 1963.

[61] Mayank Kejriwal and Ke Shen. Do fine-tuned commonsense language models really generalize? CoRR, abs/2011.09159, 2020.

[62] Frank Keller and Mirella Lapata. Using the web to obtain frequencies for unseen bigrams. Computational Linguistics, 29(3):459–484, 2003.

[63] Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. Unifiedqa: Crossing format boundaries with a single QA system. In Proceedings of EMNLP 2020 Findings, pages 1896–1907, 2020.

[64] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Proceedings of ICLR 2015, 2015.

[65] Vid Kocijan, Ana-Maria Cretu, Oana-Maria Camburu, Yordan Yordanov, and Thomas Lukasiewicz. A surprisingly robust trick for the winograd schema challenge. In Proceedings of ACL 2019, pages 4837–4842, 2019.

[66] Mahnaz Koupaee and William Yang Wang. Wikihow: A large scale text summarization dataset. CoRR, abs/1810.09305, 2018.

[67] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. Int. J. Comput. Vis., 123(1):32–73, 2017.

[68] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. In Proceedings or ICLR 2020, 2020.

[69] Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. End-to-end neural coreference resolution. In Proceedings of EMNLP 2017, pages 188–197, 2017.

[70] Kenton Lee, Luheng He, and Luke Zettlemoyer. Higher-order coreference resolution with coarse-to-fine inference. In Proceedings of NAACL-HLT 2018, pages 687–692, 2018.

[71] Douglas B Lenat and Ramanathan V Guha. Building large knowledge-based systems; representation and inference in the Cyc project. Addison-Wesley Longman Publishing Co., Inc., 1989.

[72] Hector J Levesque, Ernest Davis, and Leora Morgenstern. The winograd schema challenge. In AAAI Spring Symposium: Logical formalizations of commonsense reasoning, volume 46, page 47, 2011.

[73] Omer Levy and Yoav Goldberg. Dependency-based word embeddings. In Proceedings of ACL 2014, pages 302–308, 2014.

[74] Dingcheng Li, Tim Miller, and William Schuler. A pronoun anaphora resolution system based on factorial hidden markov models. In Proceedings of ACL 2011, pages 1169–1178, 2011.

[75] Xiang Li, Aynaz Taheri, Lifu Tu, and Kevin Gimpel. Commonsense knowledge base completion. In Proceedings of ACL 2016, volume 1, pages 1445–1455, 2016.

[76] Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. Dailydialog: A manually labelled multi-turn dialogue dataset. In Proceedings of IJCNLP 2017, pages 986–995, 2017.

[77] Yikang Li, Wanli Ouyang, Bolei Zhou, Jianping Shi, Chao Zhang, and Xiaogang Wang. Factorizable net: An efficient subgraph-based framework for scene graph generation. In Proceedings of ECCV 2018, pages 346–363, 2018.

[78] Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. Kagnet: Knowledge-aware graph networks for commonsense reasoning. In Proceedings of EMNLP-IJCNLP 2019, pages 2829–2839, 2019.

[79] Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. Kagnet: Knowledge-aware graph networks for commonsense reasoning. In Proceedings of EMNLP-IJCNLP 2019, pages 2829–2839, 2019.

[80] Bill Yuchen Lin, Seyeon Lee, Rahul Khanna, and Xiang Ren. Birds have four legs?! numersense: Probing numerical commonsense knowledge of pre-trained language models. In Proceedings of EMNLP 2020, pages 6862–6868, 2020.

[81] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In Proceedings of Text Summarization Branches Out 2004, pages 74–81, 2004.

[82] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In Proceedings of ECCV 2014, pages 740–755. Springer, 2014.

[83] Pierre Lison and Jörg Tiedemann. Opensubtitles2016: Extracting large parallel corpora from movie and TV subtitles. In Proceedings of LREC 2016, 2016.

[84] Hugo Liu and Push Singh. Conceptnet: a practical commonsense reasoning tool-kit. BT technology journal, 22(4):211–226, 2004.

[85] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. CoRR, abs/1907.11692, 2019.

[86] Songjian Lu and Xinghua Lu. An exact algorithm for the weighed mutually exclusive maximum set cover problem. CoRR, abs/1401.6385, 2014.

[87] Zhiyi Luo, Yuchen Sha, Kenny Q. Zhu, Seung-won Hwang, and Zhongyuan Wang. Commonsense causal reasoning between short texts. In Proceedings of KR 2016, pages 421–431, 2016.

[88] Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In Proceedings of EMNLP 2015, pages 1412–1421, 2015.

[89] Shangwen Lv, Daya Guo, Jingjing Xu, Duyu Tang, Nan Duan, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, and Songlin Hu. Graph-based reasoning over heterogeneous external knowledge for commonsense question answering. In Proceedings of AAAI 2020, pages 8449–8456, 2020.

[90] John McCarthy et al. Programs with common sense. RLE and MIT computation center, 1960.

[91] Ken McRae, Michael J Spivey-Knowlton, and Michael K Tanenhaus. Modeling the influence of thematic fit (and other constraints) in on-line sentence comprehension. Journal of Memory and Language, 38(3):283–312, 1998.

[92] Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. The nombank project: An interim report. In Proceedings of the Workshop Frontiers in Corpus Annotation@HLT-NAACL 2004, 2004.

[93] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In Proceedings of NIPS, pages 3111–3119, 2013.

[94] George A Miller. WordNet: An electronic lexical database. MIT press, 1998.

[95] Ruslan Mitkov. Robust pronoun resolution with limited knowledge. In Proceedings of ACL 1998, pages 869–875, 1998.

[96] Ruslan Mitkov. Anaphora resolution. Routledge, 2014.

[97] Ruslan Mitkov et al. Anaphora resolution in machine translation. In Proceedings of the Sixth International conference on Theoretical and Methodological issues in Machine Translation. Citeseer, 1995.

[98] Nasrin Mostafazadeh, Aditya Kalyanpur, Lori Moon, David W. Buchanan, Lauren Berkowitz, Or Biran, and Jennifer Chu-Carroll. GLUCOSE: generalized and contextualized story explanations. In Proceedings of EMNLP 2020, pages 4569–4586, 2020.

[99] Testuya Nasukawa. Robust method of pronoun resolution using full-text information. In Proceedings of CCL 1994, pages 1157–1163, 1994.

[100] Vincent Ng. Supervised ranking for pronoun resolution: Some recent improvements. In Proceedings of EMNLP 2005, volume 20, page 1081, 2005.

[101] Qiang Ning, Zhili Feng, Hao Wu, and Dan Roth. Joint reasoning for temporal and causal relations. In Proceedings of ACL 2018, pages 2278–2288, 2018.

[102] NIST. The ACE evaluation plan., 2005.

[103] Jong-Hoon Oh, Kentaro Torisawa, Chikara Hashimoto, Motoki Sano, Stijn De Saeger, and Kiyonori Ohtake. Why-question answering using intra- and inter-sentential causal relations. In Proceedings of ACL 2013, pages 1733–1743, 2013.

[104] Simon Ostermann, Michael Roth, Ashutosh Modi, Stefan Thater, and Manfred Pinkal. Semeval-2018 task 11: Machine comprehension using commonsense knowledge. In Proceedings of SemEval-2018, pages 747–757, 2018.

[105] Alexander P. D. Mourelatos. Events, processes, and states. Linguistics and Philosophy, 2: 415–434, 01 1978.

[106] Ulrike Padó, Frank Keller, and Matthew W Crocker. Combining syntax and thematic fit in a probabilistic model of sentence processing. In Proceedings of CogSci 2006, pages 657–662, 2006.

[107] Martha Palmer, Daniel Gildea, and Paul Kingsbury. The proposition bank: An annotated corpus of semantic roles. Computational linguistics, 31(1):71–106, 2005.

[108] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In Proceedings of ACL 2002, pages 311–318, 2002.

[109] Debjit Paul and Anette Frank. Social commonsense reasoning with multi-head knowledge attention. In Proceedings of the EMNLP 2020, Findings, pages 2969–2980, 2020.

[110] Judea Pearl. The seven tools of causal inference, with reflections on machine learning. Commun. ACM, 62(3):54–60, 2019.

[111] Judea Pearl and Dana Mackenzie. The book of why: the new science of cause and effect. Basic Books, 2018.

[112] Eyal Peer, Laura Brandimarte, Sonam Samat, and Alessandro Acquisti. Beyond the turk: Alternative platforms for crowdsourcing behavioral research. Journal of Experimental Social Psychology, 70:153–163, 2017.

[113] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In Proceedings of EMNLP 2014, pages 1532–1543, 2014.

[114] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In Proceedings of NAACL-HLT 2018, pages 2227–2237, 2018.

[115] Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. Language models as knowledge bases? arXiv preprint arXiv:1909.01066, 2019.

[116] Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. Language models as knowledge bases? In Proceedings of EMNLP-IJCNLP 2019, pages 2463–2473, 2019.

[117] Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In Proceedings of EMNLP 2012, pages 1–40, 2012.

[118] Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind K Joshi, and Bonnie L Webber. The penn discourse treebank 2.0. In Proceedings of LREC 2008. Citeseer, 2008.

[119] James Pustejovsky, Patrick Hanks, Roser Sauri, Andrew See, Robert Gaizauskas, Andrea Setzer, Dragomir Radev, Beth Sundheim, David Day, Lisa Ferro, et al. The timebank corpus. In Corpus linguistics, page 40, 2003.

[120] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. Amazon, 2018.

[121] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. OpenAI Blog, 1(8):9, 2019.

[122] Kira Radinsky, Sagie Davidovich, and Shaul Markovitch. Learning causality for news events prediction. In Proceedings WWW 2012, pages 909–918, 2012.

[123] Karthik Raghunathan, Heeyoung Lee, Sudarshan Rangarajan, Nathanael Chambers, Mihai Surdeanu, Dan Jurafsky, and Christopher Manning. A multi-pass sieve for coreference resolution. In EMNLP, pages 492–501, 2010.

[124] Altaf Rahman and Vincent Ng. Supervised models for coreference resolution. In Proceedings of EMNLP 2009, pages 968–977, 2009.

[125] Altaf Rahman and Vincent Ng. Coreference resolution with world knowledge. In Proceedings of ACL 2011, pages 814–824, 2011.

[126] Altaf Rahman and Vincent Ng. Resolving complex cases of definite pronouns: The winograd schema challenge. In Proceedings of EMNLP-CoNLL 2012, pages 777–789, 2012.

[127] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In Proceedings of EMNLP 2019, pages 3980–3990, 2019.

[128] Joseph Reisinger and Raymond J. Mooney. A mixture model with sharing for lexical semantics. In Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP 2010, 9-11 October 2010, MIT Stata Center, Massachusetts, USA, A meeting of SIGDAT, a Special Interest Group of the ACL, pages 1173–1182, 2010.

[129] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. IEEE Trans. Pattern Anal. Mach. Intell., 39(6):1137–1149, 2017.

[130] Philip Resnik. Selectional preference and sense disambiguation. Tagging Text with Lexical Semantics: Why, What, and How?, 1997.

[131] Philip Stuart Resnik. Selection and information: A class-based approach to lexical relationships. IRCS Technical Reports Series, page 200, 1993.

[132] Alan Ritter, Mausam, and Oren Etzioni. A latent dirichlet allocation method for selectional preferences. In Proceedings of ACL 2010, pages 424–434, 2010.

[133] Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S. Gordon. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In Proceedings of AAAI 2011 Spring Symposium, pages 90–95, 2011.

[134] DE Rumelhart. Notes on a schema for stories language, thought, and culture. Representation and understanding, pages 211–236, 1975.

[135] Itsumi Saito, Kyosuke Nishida, Hisako Asano, and Junji Tomita. Commonsense knowledge base completion and generation. In Proceedings of CoNLL 2018, pages 141–150, 2018.

[136] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. In Proceedings of AAAI 2020, 2019.

[137] Sandhaus and Evan. The New York Times Annotated Corpus LDC2008T19. LDC, 2008.

[138] Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A. Smith, and Yejin Choi. ATOMIC: an atlas of machine commonsense for if-then reasoning. In Proceedings of AAAI 2019, pages 3027–3035, 2019.

[139] Maarten Sap, Ronan LeBras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A. Smith, and Yejin Choi. ATOMIC: an atlas of machine commonsense for if-then reasoning. In Proceedings of AAAI 2019, pages 3027–3035, 2019.

[140] Roger C Schank and Robert P Abelson. Scripts, plans, goals, and understanding: An inquiry into human knowledge structures. Psychology Press, 2013.

[141] Lenhart K Schubert. What kinds of knowledge are needed for genuine understanding. In Proceedings of Cognitum 2015, 2015.

[142] Sebastian Schuster and Christopher D. Manning. Enhanced english universal dependencies: An improved representation for natural language understanding tasks. In Proceedings of LREC 2016, 2016.

[143] Bernhard Schölkopf. Causality for machine learning, 2019.

[144] Thibault Sellam, Dipanjan Das, and Ankur P. Parikh. BLEURT: learning robust metrics for text generation. In Proceedings of ACL 2020, pages 7881–7892, 2020.

[145] Push Singh, Thomas Lin, Erik T Mueller, Grace Lim, Travell Perkins, and Wan Li Zhu. Open mind common sense: Knowledge acquisition from the general public. In OTM Confederated International Conferences on the Move to Meaningful Internet Systems, pages 1223–1237, 2002.

[146] Noah A. Smith, Yejin Choi, Maarten Sap, Hannah Rashkin, and Emily Allaway. Event2Mind: Commonsense inference on events, intents, and reactions. In Proceedings of ACL 2018, pages 463–473, 2018.

[147] Valery D Solovyev, Vladimir V Bochkarev, and Anna V Shevlyakova. Dynamics of core of language vocabulary. arXiv preprint:1705.10112, 2017.

[148] Yan Song and Shuming Shi. Complementary learning of word embeddings. In Proceedings of IJCAI 2018, pages 4368–4374, 7 2018.

[149] Yan Song, Chia-Jung Lee, and Fei Xia. Learning word representations with regularization from prior knowledge. In Proceedings of CoNLL 2017, pages 143–152, 2017.

[150] Yan Song, Shuming Shi, Jing Li, and Haisong Zhang. Directional skip-gram: Explicitly distinguishing left and right context for word embeddings. In Proceedings of NAACL-HLT 2018, pages 175–180, 2018.

[151] Yangqiu Song, Haixun Wang, Zhongyuan Wang, Hongsong Li, and Weizhu Chen. Short text conceptualization using a probabilistic knowledgebase. In Proceedings of IJCAI 2011, pages 2330–2336, 2011.

[152] Yangqiu Song, Shusen Wang, and Haixun Wang. Open domain short text conceptualization: A generative + descriptive modeling approach. In Proceedings of IJCAI 2015, pages 3820–3826, 2015.

[153] Robert Speer and Catherine Havasi. Conceptnet 5: A large semantic network for relational knowledge. In The People's Web Meets NLP, pages 161–176. Springer, 2013.

[154] Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. In Proceedings of AAAI 2017, pages 4444–4451, 2017.

[155] Josef Steinberger, Massimo Poesio, Mijail A Kabadjov, and Karel Jevzek. Two uses of anaphora resolution in summarization. Information Processing & Management, 43(6):1663–1680, 2007.

[156] Michael Strube and Christoph Müller. A machine learning approach to pronoun resolution in spoken dialogue. In Proceedings of ACL 2003, pages 168–175, 2003.

[157] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: A core of semantic knowledge unifying wordnet and wikipedia. In Proceedings of WWW 2007, pages 697–706, 2007.

[158] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In Proceedings of NeurIPS 2014, pages 3104–3112, 2014.

[159] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In Proceedings of NAACL 2019, pages 4149–4158, 2019.

[160] Niket Tandon, Gerard de Melo, Abir De, and Gerhard Weikum. Knowlywood: Mining activity knowledge from hollywood narratives. In Proceedings of CIKM 2015, pages 223–232, 2015.

[161] Niket Tandon, Gerard de Melo, and Gerhard Weikum. Webchild 2.0 : Fine-grained commonsense knowledge distillation. In Proceedings of ACL 2017, pages 115–120, 2017.

[162] Haiqing Tang, Deyi Xiong, Min Zhang, and Zhengxian Gong. Improving statistical machine translation with selectional preferences. In Proceedings of COLING 2016, pages 2154–2163, 2016.

[163] Ozlem Uzuner, Andreea Bodnari, Shuying Shen, Tyler Forbush, John Pestian, and Brett R South. Evaluating the state of the art in coreference resolution for electronic medical records. Journal of the American Medical Informatics Association, 19(5):786–791, 2012.

[164] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Proceedings of NIPS 2017, pages 5998–6008, 2017.

[165] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In Proceedings of ICLR 2018, 2018.

[166] Jianxiang Wang and Man Lan. A refined end-to-end discourse parser. In Proceedings CoNLL 2015, pages 17–24, 2015.

[167] Liang Wang, Meng Sun, Wei Zhao, Kewei Shen, and Jingming Liu. Yuanfudao at semeval-2018 task 11: Three-way attention and relational knowledge for commonsense machine comprehension. arXiv preprint arXiv:1803.00191, 2018.

[168] Yorick Wilks. An intelligent analyzer and understander of english. Commun. ACM, 18(5): 264–274, May 1975.

[169] Yorick Wilks. A preferential, pattern-seeking, semantics for natural language inference. Artificial intelligence, 6(1):53–74, 1975.

[170] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. Huggingface's transformers: State-of-the-art natural language processing. CoRR, abs/1910.03771, 2019.

[171] Wei Wu, Fei Wang, Arianna Yuan, Fei Wu, and Jiwei Li. Corefqa: Coreference resolution as query-based span prediction. In Proceedings of ACL 2020, pages 6953–6963, 2020.

[172] Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Q Zhu. Probase: A probabilistic taxonomy for text understanding. In Proceedings of SIGMOD 2012, pages 481–492. ACM, 2012.

[173] Danfei Xu, Yuke Zhu, Christopher B. Choy, and Li Fei-Fei. Scene graph generation by iterative message passing. In Proceedings of CVPR 2017, pages 3097–3106, 2017.

[174] Nianwen Xue, Hwee Tou Ng, Sameer Pradhan, Rashmi Prasad, Christopher Bryant, and Attapol Rutherford. The conll-2015 shared task on shallow discourse parsing. In Proceedings of CoNLL 2015, pages 1–16, 2015.

[175] Jianwei Yang, Jiasen Lu, Stefan Lee, Dhruv Batra, and Devi Parikh. Graph R-CNN for scene graph generation. In Proceedings of ECCV 2018, pages 690–706, 2018.

[176] Changlong Yu, Hongming Zhang, Yangqiu Song, and Wilfred Ng. Cocolm: Complex commonsense enhanced language model. CoRR, abs/2012.15643, 2020. URL `https://arxiv.org/abs/2012.15643`.

[177] Jeffrey M Zacks and Barbara Tversky. Event structure in perception and conception. Psychological bulletin, 127(1):3, 2001.

[178] Beñat Zapirain, Eneko Agirre, Lluís Màrquez, and Mihai Surdeanu. Selectional preferences for semantic role classification. Computational Linguistics, 39(3):631–663, 2013.

[179] Rowan Zellers, Yonatan Bisk, Ali Farhadi, and Yejin Choi. From recognition to cognition: Visual commonsense reasoning. In Proceedings of CVPR 2019, pages 6720–6731, 2019.

[180] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In Proceedings of ACL 2019, pages 4791–4800, 2019.

[181] Hongming Zhang, Liwei Qiu, Lingling Yi, and Yangqiu Song. Scalable multiplex network embedding. In Proceedings of IJCAI 2019, pages 3082–3088, 2018.

[182] Hongming Zhang, Hantian Ding, and Yangqiu Song. SP-10K: A large-scale evaluation set for selectional preference acquisition. In Proceedings of ACL 2019, pages 722–731, 2019.

[183] Hongming Zhang, Yan Song, and Yangqiu Song. Incorporating context and external knowledge for pronoun coreference resolution. In Proceedings of NAACL-HLT 2019, 2019.

[184] Hongming Zhang, Daniel Khashabi, Yangqiu Song, and Dan Roth. Transomcs: From linguistic graphs to commonsense knowledge. In Proceedings of IJCAI 2020, pages 4004–4010, 2020.

[185] Hongming Zhang, Xin Liu, Haojie Pan, Yangqiu Song, and Cane Wing-Ki Leung. ASER: A large-scale eventuality knowledge graph. In Proceedings of WWW 2020, pages 201–211, 2020.

[186] Hongming Zhang, Xinran Zhao, and Yangqiu Song. Winowhy: A deep diagnosis of essential commonsense knowledge for answering winograd schema challenge. In Proceedings of ACL 2020, pages 5736–5745, 2020.

[187] Yingyi Zhang, Jing Li, Yan Song, and Chengzhi Zhang. Encoding conversation context for neural keyphrase extraction from microblog posts. In Proceedings of NAACL-HLT 2018, pages 1676–1686, 2018.

[188] Ben Zhou, Qiang Ning, Daniel Khashabi, and Dan Roth. Temporal common sense acquisition with minimal supervision. In Proceedings of ACL 2020, pages 7579–7589, 2020.

[189] Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. Commonsense knowledge aware conversation generation with graph attention. In Proceedings of IJCAI 2018, pages 4623–4629, 2018.

[190] Will Y Zou, Richard Socher, Daniel Cer, and Christopher D Manning. Bilingual word embeddings for phrase-based machine translation. In Proceedings of EMNLP 2013, pages 1393–1398, 2013.