



TEXT LOGICAL SCORING BASED ON KNOWLEDGE GRAPH

By

YEWEI SONG

A thesis submitted to
the University of Birmingham
for the degree of
MASTER OF SCIENCE
Supervised by
P.J.HANCOX

School of Computer Science
College of Engineering and Physical Sciences
University of Birmingham
September 2020

ABSTRACT

Computer automatic essay scoring system has been widely used. But most systems do not include a score for logic. The author proposes a method of logical scoring by constructing a knowledge graph on essay using information extraction technology. This paper first discussed the steps of information extraction and knowledge graph construction and related background technologies, compared different information extraction methods. In the experimental chapter, qualitative analysis is used to discuss the feasibility of the method, and then quantitative analysis is used to discuss the effectiveness of the method. The thesis summarizes various issues regarding the accuracy of information extraction, co-reference resolution, entity recognition and scoring standards. Finally proved that the method could be used for scoring but there is still a lot of future work for optimization.

ACKNOWLEDGMENTS

First, my thanks go to the School of Computer Science. Thank you for the improvement given to me by studying here this year. Then, thank my family for supporting me to go to the UK to complete my studies. Finally, I want to thank my supervisor P.J.Hancox for his guidance through my time in Birmingham also inspiration on research methods and discovery of topics.

*Never trust to general impressions
my boy
but concentrate yourself upon details.*

Arthur Conan Doyle - Sherlock Holmes

Contents

	Page
1 Introduction	1
1.1 Problem	1
1.2 Goal	2
1.3 Thesis structure	2
2 Background	4
2.1 Previous Work	4
2.1.1 Text Scoring	4
2.1.2 Text Logical Analysis	5
2.2 Information Extraction	7
2.2.1 Sentence segmentation	7
2.2.2 Tokenization	9
2.2.3 Stemming or Lemmatisation	10
2.2.4 Part-of-speech Tagging	13
2.2.5 Name Entity Recognition	15
2.2.6 Syntactic Parsing	19
2.2.7 Reference Resolution	27
2.3 Knowledge Graph	30
2.3.1 Triples Extraction	30
2.3.2 Graph building	32
2.3.3 Relational Learning	34
3 Experiments	38
3.1 Method Describe	38
3.2 Qualitative analysis	39

3.2.1	Experiment Aim	39
3.2.2	Experiment Design	40
3.2.3	Experiment Procedure	41
3.2.4	Experiment Result & Analysis	48
3.3	Quantitative analysis	49
3.3.1	Experiment Aim	49
3.3.2	Experiment Design	49
3.3.3	Experiment Procedure	50
3.3.4	Experiment Result	52
4	Conclusion	53
4.1	Achievements	53
4.2	Critical analysis	54
4.2.1	Effectiveness of the scoring method	54
4.2.2	The feasibility of a pure end-to-end approach	54
4.2.3	Unreliable NLP technology	54
4.3	Problems analysis	55
4.3.1	Error accumulation problem	55
4.3.2	Entity extraction problem	55
4.3.3	Relationship duplication problem	56
4.3.4	Entity failed to merge	57
4.3.5	High-level writing is difficult to distinguish	57
4.4	Future works	58
4.4.1	Improve the accuracy of each step	58
4.4.2	Predict possible relationships	58
4.4.3	Better scoring standards	58
A	Result Examples	60
A.1	Quantitative analysis result charts	60
A.2	Program code and user guide	61
	References	62

List of Figures

2.1	An illustration of Stanford Open IE’s approach. The left part is the dependency of the long sentence, and the right part is the clause.(Angeli, Premkumar, and C. D. Manning, 2015) . . .	7
2.2	A comparison of three stemming algorithms on a sample text.(Cambridge, 2009)	12
2.3	Words, morphological tags and lemmas.(Malaviya, S. Wu, and Cotterell, 2019)	13
2.4	A Linear Chain Conditional Random Field.	17
2.5	Dilated convolution diagram	18
2.6	Example of syntactic structure parsing and dependency syntactic parsing	19
2.7	Two different PFCG parse trees of He met Lily with flowers.	23
2.8	Projective dependency tree(Jurafsky, 2000)	25
2.9	Non-projective dependency tree(Jurafsky, 2000)	25
2.10	Neural network architecture of Dependency parser by D. Chen and C. D. Manning (2014) . .	27
2.11	Anaphora vs. Coreference Resolution	29
2.12	The architecture of PCNNs(Zeng, Liu, Y. Chen, et al., 2015)	31
2.13	bidirectional sequential and bidirectional tree-structured LSTM-RNNs(Miwa and Bansal, 2016)	32
2.14	Technical architecture of knowledge graph	34
2.15	The framework of PECNN. The red and blue circles represent the word embeddings and tree-based position features of words. The yellow and green circles stand for the feature maps extracted by two kinds of convolution kernels respectively.(Y. Yang et al., 2016)	36
2.16	The ConVE model, by four steps(Dettmers et al., 2018)	36
3.1	Overall system architecture of Stanford CoreNLP toolkit(C. D. Manning et al., 2014)	39
3.2	Lemmatisation running example on Stanford CoreNLP.	42
3.3	Part-of-speech tagging running example on Stanford CoreNLP.	42
3.4	Dependency Parsing running example on Stanford CoreNLP.	43
3.5	Constituent Parsing running example on Stanford CoreNLP.	43
3.6	Open IE running example on Stanford CoreNLP.	44

3.7	An end-to-end model based on BERT.	45
3.8	Coreference Resolution Result Example	46
3.9	Knowledge Graph Example from essay	47
3.10	Model Training of BERT based relation extraction	51
3.11	Visualized spaCy coreference resolution results	51
4.1	Nested entities from "great influence over children", as "influence over children", "great in- fluence" and "influence"	56
4.2	Duplication caused by an inverted relationship	56
A.1	Data result compare, green is relations before merge, red is after mergem, blue is scale of largest sub-graph, yellow is IELTS writing score	60

List of Tables

2.1	PCFG phrase structure analyzer	22
3.1	The relationship number of different essays obtained after Triple extraction, including the duplicate relationship before and after the merger.	52
3.2	The number of sides of the constructed knowledge graph after co-reference resolution.	52

Acronyms

Bi – LSTM Bi-directional long short term memory. 15

HMM Hidden Markov Model. 15

IE Information Extraction. 1

KG Knowledge Graph. 1

LSTM Long Short-Term Memory. 17

NER Name Entity Recognition. 2

NLP Natural Language Processing. 5

NN Neural Network. 12

POS Part-of-Speech. 13

Chapter One

Introduction

1.1 Problem

Essay's automatic scoring system has a long history of development, but the logical scoring function has always been lacking. How to automatically evaluate the coherence of logic and the connectivity of opinions is difficult before the foundation of text understanding has developed to a certain level. Logical relationships can also be considered as dependencies. The classic syllogism reasoning is to make inferences through major premises, minor premises and conclusions. This process can be summarized as the following formula:

$$(a \Rightarrow b) \wedge (b \Rightarrow c) \Rightarrow (a \Rightarrow c) \quad (1.1)$$

So we only need to obtain the dependency relationship between a , b , and c , and we can get an $a \Rightarrow c$ inference relationship. Therefore, we can simplify the problem to how to extract dependencies. According to the practice of the Open Information Extraction (*IE*) system, the relationship is usually summarized as a triple containing subject, object, and relationship.

Because the logical relationship is not isolated, a good essay must have logical coherence. Therefore, investigating logic can't only consider the number of extracted relations, but should connect the relations. Connecting the relationship into a directed graph, we can get the Knowledge Graph (*KG*), the prototype of which was proposed as early as Schneider, 1973. Usually the directed graph we get through a set of relations will not be a single one, but a forest. So the problem of measuring logic can be transformed into finding the largest connected component of the forest.

The problem is summarized in three steps: first understand the text and obtain the triple relationship, then

connect the relationship into a directed graph, and finally calculate the score based on the directed graph forest.

1.2 Goal

As out line above, The author uses KG for the logical scoring of the essay. In order to solve the problems raised, the goal can be divided into the following components which will be address in this work.

- Process and understand the text, try different ways to obtain triples. For example, extract the relationship directly from the sentence, or obtain the dependency syntax tree first, and then obtain the logical relationship.
- The triple relationship is merged, and the referential, identical or highly similar entities are merged using coreference resolution technology.
- The merged relationship is constructed into a knowledge graph, which is a forest containing one or more directed graphs.
- According to the knowledge graph and logical relationship, calculate the score of the essay. Discuss possible scoring criteria and compare them.

1.3 Thesis structure

The first part of the paper discusses the previous system, including the technology used and its advantages and disadvantages. Then discuss how to do logical analysis of the text. This part mainly obtains relevant information from the literature.

The next part is about background technology. It is mainly divided into relationship extraction and knowledge map construction. The part of relation extraction includes all the technologies needed for text understanding and relation extraction, from the preprocessing part of Sentence segmentation, Tokenization, Lemmatisation and Part-of-speech Tagging, to Name Entity Recognition (*NER*), Parsing and Reference Resolution. Among them, co-reference resolution is also used in relationship deduplication. Knowledge graph construction is mainly divided into the generation of directed graphs and relational reasoning. The above techniques are used in experiments, or as part of black box tools.

The experiment part mainly includes qualitative analysis experiment and quantitative analysis experiment. Qualitative analysis experiments focus on the feasibility of obtaining triple relationships and knowledge graphs from essay texts as a whole. Use examples to analyze the execution of each step and discuss how the results can be used in the next step. It also compared the pros and cons of using different technologies for each step. The quantitative analysis experiment extracts relationships according to the selected optimal relationship extraction method, and discusses which scoring standard is more suitable for logical analysis based on the analysis results of 15 essays.

Chapter Two

Background

2.1 Previous Work

2.1.1 Text Scoring

Using automatic essay scoring system to assist teachers is an important research direction. However, due to education fairness considerations, it is very important to design a reasonable scoring algorithm that is consistent in the way it scores essays. In the early 21st century, several practical essay scoring systems were widely used, such as Project Essay Grade (PEG)(Page, 2003), Intelligent Essay Assessor (IEA)(Foltz, Laham, and Landauer, 1999), Electronic Essay Rater (E-Rater)(J. Burstein and Chodorow, 1999). These systems have different basis for scoring, and the evaluation algorithms used are also different(Shermis and J. C. Burstein, 2003).

PEG was designed by Ellis Page at the request of the American University Committee in 1966, and its research purpose was to make the grading of batch essays more efficient. PEG totally uses the analysis of the shallow-level linguistic features to score the essay without considering the deep content of the essay. Because the fluency of the writing, the complexity of the sentence, and the level of words are difficult to measure directly by the computer, the author used the proxy measures method to calculate various indicators. For example, the length of essay represents fluency, and the number of pronouns and prepositions represents the complexity of the sentence. Because PEG ignores semantics, it cannot give effective feedback to users, and it is very easy to be deceived by special skills, such as deliberately using long spelled words to obtain high vocabulary scores.

IEA was developed by Pearson in 1998 and is based on latent semantic analysis technology. Latent semantic analysis is a text indexing and information extraction method based on statistics. It is a statistical model of word usage that allows comparison of the semantic similarity between the information contained in the segment text. The term frequency vector is calculated by term frequency–inverse document frequency (TF-IDF), and the higher the similarity of the vectors, the more similar the meaning. Therefore, the IEA system compares the student’s essay with the reference text of known writing quality, obtains the vector cosine similarity and obtains the score.

E-rater was launched in 1999 by J. Burstein and Chodorow (1999). of the Educational Testing Service (ETS). The E-rater system includes three components: a lexical analyzer, a syntactic analyzer, and a text analyzer. The scoring process consists of 5 independent modules. Three are used to extract scoring features, including: syntax module, chapter module and topic analysis module. These three modules are used to extract the feature values of 67 text features in essay’s syntactic diversity, topic organization and vocabulary usage. The fourth module, the model building module, uses the data extracted from the first three modules as independent variables, and the artificial scores as dependent variables for linear regression. It screens among 67 variables and establishes an essay scoring model. The fifth module is used to calculate the final score of the article to be scored, that is, extract the essay feature value, and substitute it into the regression model to calculate the score.

In addition to the above three systems, there are IntelliMetric™, BETSY, etc. Most of these systems rely on the extraction of linguistic features as the basis for scoring, and model the scoring model through statistical or machine learning methods. Some challengers have proposed the idea of using hidden Markov models or neural networks to extract linguistic features, but high-level content usually be ignored.

2.1.2 Text Logical Analysis

Before we discuss the logical scoring of essay, we should first discuss how to analyze the logical structure of the document. Since the 1990s, with the development of information extraction technology, how to represent the logical relationship between entity objects is an important Natural Language Processing (*NLP*) topic. In late 1970’s, Yale’s artificial intelligence team developed FRUMP (Fast Reading Understanding and Memory Program)(DeJong, 1982). Its design goal is to skim and summarize international news from the Associated Press. Use sketchy scripts to extract keywords in more than 60 special scenes to achieve logical relationship extraction. This system is not particularly intelligent, but inspired the later SCISOR system of

the 1990s(Rau, 1987). SCISOR combines full syntactic parsing and conceptual expectation-driven parsing. It shows the advantages and potential of natural language processing technology in the field of information extraction.

Riloff (1996) described the use of supervised learning techniques in information extraction. As Hidden Markov Model technology is applied to serialized data recognition, such as speech recognition technology, more machine learning IE methods are proposed. McCallum, Freitag, and Pereira (2000) proposed MEMM (Maximum Entropy Markov Models), and Lafferty, McCallum, and Pereira (2001) proposed CRF (Conditional Random Fields). These techniques are used to process the labeling and identification of serialized data.

F. Wu and Weld (2010) chose Wikipedia as the data source, and proposed a prototype implementation of a semi-supervised machine learning information extraction system. This method, called the distant supervision algorithm, has proved that its accuracy is not weaker than that of human readers in certain scenarios. In 2010, they proposed *WOE^{parse}*, which significantly improved the recall rate and accuracy rate. TEXTRUNNER is another famous s Open Information Extraction(OIE) system supposed by Banko et al. (2007). Their main idea is to use the classifier to evaluate the confidence of candidate tuples and to strengthen the confidence of the tuples that appear multiple times.

While the development of the IE system based on machine learning, the technological improvement of Rule-based Systems has also been carried out. REVERB(Fader, Soderland, and Etzioni, 2011) uses some syntactically constrained and lexically constrained extractors to effectively improve three common problems in Open IE systems: uninformative extractions, incoherent extractions, and overly-specific relations. Stanovsky et al., 2016 suggested a more abstract way by building a directed graph from dependency parse tree. PredPatt(White et al., 2016) uses Universal Dependency (UD)(De Marneffe et al., 2014) to extract the predicate parameter structure and construct a directed graph based on the dependency relationship between the predicate and its arguments. Because the UD structure uses language-independent extraction modes, PredPatt can work across different languages differ from other Open IE tools.

One way to reduce IE errors in recent years is based on clause analysis. Use a verb to reconstruct a sentence into multiple different types of clauses. These simple clauses can be subdivided using Open IE tuples. ClausIE(Del Corro and Gemulla, 2013) uses this idea to map the dependency of the input sentence to the clause components. The system can extract a part of the effective information from each clause.

Angeli's team proposed Stanford Open IE(Angeli, Premkumar, and C. D. Manning, 2015). It analyzes the component dependencies in the sentence first, then traverses the dependency tree and uses a classifier at each step to predict whether an independent clause should be generated. After splitting a sentence into

many independent clauses, use natural logic inference to shorten them. Finally, match these clauses to useful information. The figure 2.1 describes this method. This tool is also the main technical tool used in this article.

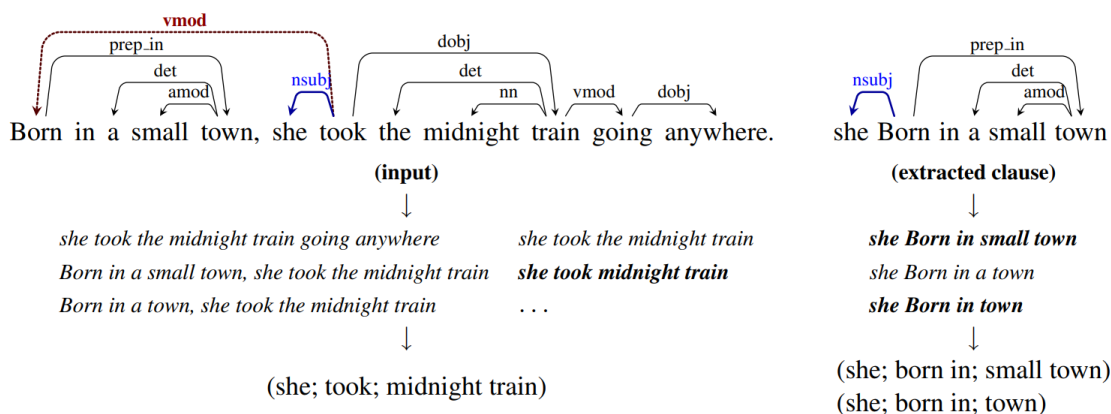


Figure 2.1: An illustration of Stanford Open IE's approach. The left part is the dependency of the long sentence, and the right part is the clause. (Angeli, Premkumar, and C. D. Manning, 2015)

2.2 Information Extraction

2.2.1 Sentence segmentation

Text segmentation is the process of cutting text into meaningful items. This process exists not only in the computer understanding of the text, but also in the psychological process of human reading text. Although this process is very simple for humans with reading ability, it is not the case for computers. Although some written languages have clear word boundary markers, such as spaces between words in written English and Beginning of sentence in Arabic, these signals are sometimes ambiguous and not always present in all written languages.

Cutting problems can be divided into the following categories:

- **Word segmentation:** Word segmentation is the problem of dividing text into its constituent words. In many European languages including English and French, the space is the word separator, and the words

are not continuous. But even in these languages, such as "-" conjunctions, but also need to consider changes in part of speech and collocation. In other languages, there is no obvious word separator in written languages, such as Chinese. Chinese word segmentation is a difficult problem.

- Intent segmentation: Intent segmentation differs from word segmentation in that the goal is to cut into key phrases (2 or more word groups).
- Sentence segmentation: Sentence segmentation is the problem of dividing a string of text into sentences. Punctuation marks are used in many languages, especially the period as a marker for dividing sentences, such as English. Not all written languages contain punctuation marks, but most languages include similar rules. An exception is classical Chinese, which has no punctuation.
- Topic segmentation: Splitting the text into paragraphs with different topics plays a significant role in information retrieval or speech recognition.

A statistical method of automatically dividing text into coherent segments was proposed by Beeferman, Berger, and Lafferty (1999). A statistical method for automatically dividing text into coherent segments was proposed by John Lafferty in 1999. This method trains an exponential model to extract features related to boundaries in labeled training text. The features are divided into two categories: topicality features and cue-word features. Topicality features perceive broad changes of topic, and cue-word features detect occurrences of specific words. The author has verified both in the news articles of the Wall Street Journal and the news reports of TV broadcasts, and proved the effectiveness in both areas.

Word segmentation is one of the most basic problems of text segmentation. Although the processing of English text does not require word segmentation in most cases, this article also briefly introduces it here so as to promote it to more languages. Word segmentation is one of the most basic problems of text segmentation. Although the processing of English text does not require word segmentation in most cases, this article also briefly introduces it here so as to promote it to more languages. The word segmentation method based on word tagging learning started from Xue (2003), using relative position tags to express the boundary information carried by the word.

Ng and Low (2004) applied strict string labeling learning to word segmentation for the first time, using the Maximum Entropy Markov model. Subsequently, conditional random field (CRF) was widely used for word segmentation, and Sun et al. (2009) used semi-CRF with hidden variables for word segmentation. CRF is less likely to fall into a local optimum, but the cost is a higher amount of calculation. After the practical application of word embedding technology, deep learning has been widely used in various tasks of natural language processing. L. Zhang et al. (2013) proposed a neural network Chinese word

segmentation method, which for the first time verified the feasibility of applying deep learning methods to Chinese word segmentation tasks. M. Zhang, Y. Zhang, and Fu (2016) proposed a transfer-based model for word segmentation, and tried the fusion method of features automatically extracted by neural network and traditional discrete features. Cai et al. (2017) designed a fast word segmentation system based on greedy search by simplifying the network structure, mixing word input, and using early update and other training strategies with better convergence.

Text segmentation is the first step in extracting information in many languages. At the same time, topic segmentation also plays an important role in some special essay scoring tasks, such as evaluating each part of a multi-topic essay. In order to improve the text segmentation technology, one aspect is to use a larger corpus, etc. On the other hand, because the cost of labeling data is very expensive, the development of unsupervised algorithms or semi-supervised algorithms is also a research hotspot.

2.2.2 Tokenization

A simple definition of Tokenization is the unstructured data, such as text and audio, referred to as a token. In the processing of text data, it specifically refers to segmenting the text and giving each word a token as an index. Tokenization not only deals with words, but also punctuation marks, hyphens, etc. Even in the recognition of English text, we cannot rely solely on spaces as the basis for word segmentation. For example, the following text.

When I got to the Alconburys' and rang their entire-tune-of-town-hall-clock-style doorbell I was still in a strange world of my own—nauseous, vile-headed, acidic

Another example is "New York-based", when we only rely on spaces, it will become "New|York-based", but actually the division is "New York|-|based". Some special words like "C++" or "01/07/2020", they are one word but might be segment into many meaningless units by simple rule. Indo-European languages is quite easier in this task than Sino-Tibetan or Japonic languages, but for higher accuracy, this is an endless challenge for each language.

After the word segmentation is completed, each item is classified and transformed into structured data. An example of converting text into symbolic expressions is as follows:

In the text string:

The quick brown fox jumps over the lazy dog

To the s-expressions:

(sentence
(word The)
(word quick)
(word brown)
(word fox)
(word jumps)
(word over)
(word the)
(word lazy)
(word dog))

The lexical analyzer usually retains enough information to reproduce the original morpheme in order to use it for parsing. The parser usually retrieves this information from the lexical analyzer and stores it in the abstract syntax tree. However, Tokenization is only the first step in lexical processing, and usually we still need to further analyze the words to support the grammar parser.

2.2.3 Stemming or Lemmatisation

A typical common NLP text processing problem is how to deal with the inflection or prefix and suffix of the synthetic language. A simple example is to group "fisher", "fishing" and "fished" into "fish". In most cases, this problem only exists in fusional language and some agglutinative languages. This process can also be referred to as word normalization. But the ideas of Stemming and Lemmatisation are different. Stemming reducing inflected words to their word stem, base or root form. And lemmatisation is to convert inflected words into Lemma, or dictionary form. See the specific difference below:

Stemming extracts the stem or root form of a word, and the result obtained may not be able to express the complete semantics. The purpose of Stemming is to map related words to the same stem, even if the related words themselves are not valid roots. For example, "wolves" may be processed as "wolv", and the stem is not a complete word.

Lovins, 1968 first published a stemmer with longest-match stemming algorithm for English. This article provides guidelines for subsequent research. The algorithm involves 3 aspects:

- ending, a total of 294 word suffixes
- condition, a total of 29 conditions, suffixes that meet the conditions can be removed
- transformations, 35 in total, They are the ways to transform endings.

The procedure of the algorithm can be divided into two parts: Part A, for English words, according to the ending list, scan from long to short according to the ending, to find the first ending that meets the condition; Part B, for the remaining stem after cutting, use the transformation correspond to ending, transform into the appropriate form.

A later research by Porter et al. (1980), was widely used became the de facto standard of English stemming with suffix stripping. This algorithm removes and transforms the affixes at the end of the word in 5 steps, and can handle most regular English word stemming tasks. Some examples of the rules include:

- if the word ends in 's', remove the 's'
- if the word ends in 'ing', remove the 'ing'
- if the word ends in 'est', remove the 'est'

Afterwards, many improvements to the Porter algorithm were proposed, especially some rules. A major improved version was proposed by Porter (2001) himself, and this algorithm is called Snowball. This version is about 5% different from the original algorithm, but the author believes that this version is better than the one 20 years ago.

Another algorithm with more radical rules is Paice-Husk Stemmer proposed by Paice (1990) Lancaster University. It is a iterative stemmer, with customizable rules for removal or replacement of an ending. Its replacement technology avoids the another stage of recode or provide partial matching but might get some strange stems shorter than previous two.

Sample text: Such an analysis can reveal features that are not easily visible from the variations in the individual genes and can lead to a picture of expression that is more biologically transparent and accessible to interpretation

Lovins stemmer: such an analys can reve featur that ar not eas vis from th vari in th individu gen and can lead to a pictur of expres that is mor biolog transpar and acces to interpres

Porter stemmer: such an analysi can reveal featur that ar not easili visibl from the variat in the individu gene and can lead to a pictur of express that is more biolog transpar and access to interpret

Paice stemmer: such an analys can rev feat that are not easy vis from the vary in the individ gen and can lead to a pict of express that is mor biolog transp and access to interpret

Figure 2.2: A comparison of three stemming algorithms on a sample text.(Cambridge, 2009)

With the development of search engines, stemming has a greater demand, that is, the extraction of search keywords. For example, Context sensitive stemming proposed by Peng et al. (2007) provides a method to predict that morphological changes are beneficial to search results. Other challenging stemming appears in other languages with complex morphology, orthography or character encoding. Examples include Italian (greater number of verb inflections), Hebrew (nonconcatenative morphology), and Russian (more noun declensions). **Lemmatisation** is the process to group different inflected forms of a word into the same group, and use lemma or dictionary form as identification. Unlike stemming mechanical processing of word strings, lemmatization needs to understand the intend part of speech and meaning of the target word. Lemmatization and stemming are closely related, in the following example:

- "best" has "good" as its lemma. Stemming can not retrieve it.
- "eating" has "eat" as its lemma. This is matched in both stemming and lemmatisation.
- "meeting" can be a noun or a verb. Stemming will handle it all as verb and ignore the using of noun.

However, stemmer is easier to design and implement, and calculation is faster. And in the information retrieval system, using stemming can improve the recall rate, or true positive rate. But stemming reduces precision, or true negative rate in some systems.(Cambridge, 2009).

Lemmatization is modelled as a classification problem by Chrupała (2006). The algorithm generate "edit trees" have leaf nodes which can be key element of inflection. Bergmanis and Goldwater (2018) use encoder-decoder Neural Network (*NN*) design a lemmatizer they name as Lematus. Both encoder and decoder are 2-layer Gated Recurrent Unit (*GRU*). They pointed out that if the words in a language do not have

many ambiguities, but has many unseen words, then a context-free system is feasible. Malaviya, S. Wu, and Cotterell (2019) use a joint NN mode for morphological tagging and lemmatization, especially helpful in low-resource lemmatization. They use 2-layer BiLSTM encoder and 1-layer LSTM decoder.

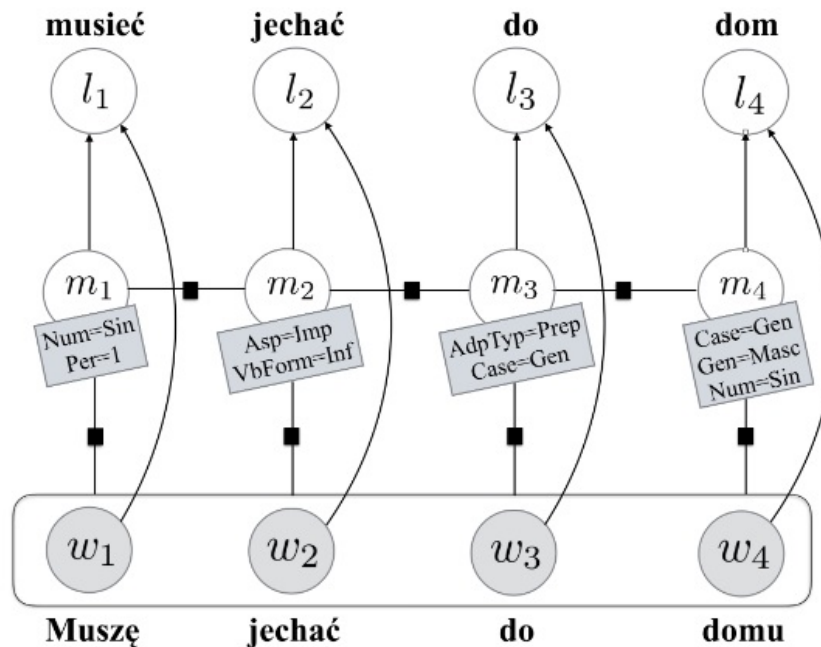


Figure 2.3: Words, morphological tags and lemmas.(Malaviya, S. Wu, and Cotterell, 2019)

2.2.4 Part-of-speech Tagging

Part-of-Speech (*POS*) tagging is the process of tagging words in text to specific parts of speech associated with context. This label indicates usage and its syntactic function, such as nouns, verbs, adjectives, etc. In the process of computer simulation to understand language, we first need to understand the rules of natural language. The most basic rule is lexical, especially part of speech. Of course, in addition to part of speech, various inflectional forms of verbs such as tense, person, and singular and plural are all objects that need POS tagging analysis. This process can also be applied to the lemmatization above, as a pre-step to obtain the inflection, and then use the lexicon to restore the word to lemma.

A simple POS tagging case is as follows:

Sam bought a book.

The tag list is:

Sam -Proper noun
bought -Verb past tense
a -Determiner
book -Noun

This example shows some common problems that may be encountered. The word "bought" can also be used as an adjective, and "book" can also be a verb or noun. Language is an extremely variable active entity, so we cannot achieve POS tagging through simple matching. An extreme example is as follows:

Will Will will the will to Will?

Will can be a person's name, can represent testament (verb or noun), can represent what will be done. Solving this problem usually consists of two ideas, symbolic and statistical. The symbolic method consists of a set of rules constructed for different language phenomena. These rules are usually written manually, but sometimes they are learned automatically. Statistical methods usually use machine learning algorithms to learn language phenomena, and treat part-of-speech tagging as a sequence tagging problem. The basic idea is: given a sequence of annotated words, we can predict the most likely part of speech of the next word. The usage of Part-of-speech Tagging can be summarized into following categories.

- A. Word sense disambiguation: Some words have many meanings based on usage. For example, in "book a hotel", "book" is used as a verb, and in "buy a book", it is used as a noun. Appearing in different contexts, the part-of-speech tags of the same word may be different.
- B. Improve vocabulary-based features: When the feature is connected with part of speech, words of different parts of speech in the context will not be summarized into the same word, so stronger features are obtained.
- C. Normalization and Lemmatization: Part-of-speech tags are the basis for translating words into their dictionary form (lemma)
- D. Remove stop words efficiently: Part-of-speech tags are also very useful in removing stop words.

The research of part-of-speech tagging is closely related to the development of corpus linguistics. In 1987, UCREL developed CLAWS(Garside, 1987), which could be the earliest POS tagging system. Their method comes from the earlier work of using the Hidden Markov Model (*HMM*) on Lancaster-Oslo-Bergen Corpus to eliminate part-of-speech ambiguity(Johansson et al., 1986). DeRose (1988) and Church (1989) have independently developed methods of using dynamic programming for part-of-speech tagging. DeRose uses a table of pairs, and Church uses a table of triples, and they both achieved an accuracy of more than 95%. This method of purely using dynamic programming simplifies the theory and practice of computerized language analysis, thus encouraging researchers to separate the various parts of natural language processing tasks. Since the 2000s, NLP based on statistical methods has become very popular. Toutanova and C. Manning (2000) proposed a part-of-speech tagger based on Maximum Entropy to achieve excellent performance by enriching the information source for tagging. Compared with the previous system, it can better handle the capital letters of unknown words, remove the tense form of verbs, and remove ambiguities from prepositions and adverbs. In 2003, Toutanova, Klein, et al. (2003) published some improvements. The first is to use the two labels before and after the target word through the dependency network representation. Then the model can effectively use a priori and richer vocabulary features. It can also perform fine-grained modeling of unknown word features.

With the rise of deep learning, neural networks are also used for part-of-speech tagging tasks. Plank, Søgaard, and Goldberg (2016) proposed a new Bi-directional long short term memory (*Bi-LSTM*) model for POS tagging. The loss function used by this model can more consider the vocabulary that rarely appears. They compared their method with the traditional POS tagging method, and the tests in 22 languages achieved the best performance at that time. NAACL 2018 Best Paper by Peters et al. (2018) proposed a new type of deep contextualized word representations which can model the complex features of word use (syntax and semantics) and polysemous words use in the language context.

Deep learning technology improves the accuracy of part-of-speech tagging, but it also has the disadvantage of requiring a large amount of tagging data. Different languages have different tagging difficulties. For example, the accuracy of English part-of-speech tagging can reach about 97%, while Chinese part-of-speech tagging is much more difficult.

2.2.5 Name Entity Recognition

Named entity recognition(NER) is one of the important subtasks of NLP. Its purpose is to locate and classify named entities in text information. Common named entities generally include three categories (entity, time,

and number) and seven subcategories (person, place, organization, time, date, currency, and percentage) named entities. In some scenarios, not only need to identify entities in general cognition, but also need to identify similar product names, models, professional terms, and law names as named entities. In some scenarios, not only need to identify entities in general cognition, but also need to identify similar product names, models, professional terms, and legal names as named entities. In some cases, long texts such as book titles, song titles, journal names, or short texts such as abbreviations, phone numbers, and the like are required to be recognized as entities. A example of NER is:

John bought 100 shares of Apple Inc. in 2019.

And producing an annotated block of text that highlights the names of entities:

*[John]*_{Person} bought 100 shares of *[AppleInc.]*_{Organization} in *[2019]*_{Time}.

From the perspective of natural language processing, NER can be regarded as a type of unregistered word recognition in lexical analysis. It is the problem with the largest number of unregistered words, the most difficult recognition, and the greatest impact on the effect of Tokenization. The most common difficulties for NER include:

- Lost token (for example, missing the last "Ph.D" of "John Smith, Ph.D.")
- Extra token (for example, including the first word of "The Stanford University")
- Divide an entity into multiple (for example, treating "John, Sam Smith" as 2 against 3 entities)
- Wrong type (for example, calling a personal name a company)
- Nested entity (for example, treating "Johns Hopkins" as a name, when it's part of "Johns Hopkins University")

Due to the existence of these problems, the evaluation of NER is a challenging problem. Common evaluation methods include accuracy, recall and F1 algorithm. But because most real-world texts have only a few named entities, the probability of not being named entities is very high. Another problem is that failure of predict full span of an entity name is not properly penalized. In the actual implementation of the evaluation algorithm, we can think that the algorithm is always pessimistic. For example, when a system always ignores titles such as "Ph.D" and "Sir", the system has zero points in the entity recognition of each person's name.

Part of the credit can also be obtained for overlapping named entities by evaluation models based on a token-by-token matching which have been proposed by Esuli and Sebastiani (2010).

Rau (1991) first described a system for extracting and identifying company names. The system mainly uses heuristic algorithms and manual rules. Grishman and Sundheim (1996) was the first to propose the concept of named entity recognition in Message Understanding Conferences. Traditional machine learning methods were introduced into NER tasks in the late 1990s. Bikel, S. Miller, et al. (1998) presents a machine learning NER method by Hidden Markov Model to finding names and other non-recursive entities in text. Maximum Entropy Model was introduced to solve NER problem by Borthwick et al. (1998). Then there is the support vector machine algorithm proposed by Asahara and Matsumoto (2003) and Boosting and voted perceptron by Collins (2002). Conditional Random Field (CRF) is the most popular model of NER in traditional machine learning algorithms. Its objective function not only considers the input state characteristic function, but also includes the label transition characteristic function. When the model is known, finding the predicted output sequence for the input sequence, that is, finding the optimal sequence that maximizes the objective function, is a dynamic programming problem, which can be decoded by the Viterbi algorithm to obtain the optimal label sequence. The advantage of CRF is that it can utilize rich internal and contextual feature information in the process of labeling an entity (McCallum and Li, 2003).

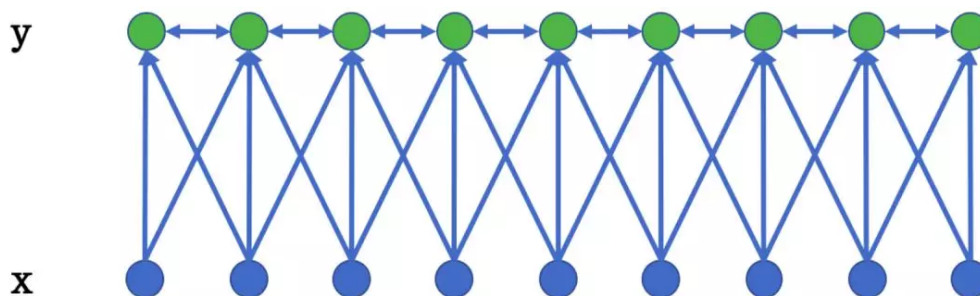


Figure 2.4: A Linear Chain Conditional Random Field.

In recent years, due to the proposed bag-of-words model and the improvement of machine performance, neural networks have begun to be applied to NER tasks. The token is mapped from discrete one-hot encoding to a low-dimensional space to become dense embedding. Then input the embedding sequence of the sentence into the RNN, use the neural network to automatically extract the features, and Softmax to predict the label of each token. Huang, Xu, and K. Yu (2015) proposed a bidirectional Long Short-Term Memory (*LSTM*) based models, with Embedding layer and CRF layer. The experimental results show that biLSTM-CRF has

reached or exceeded the CRF model based on rich features. The model does not require feature engineering, and the use of word vectors and character vectors can effectively improve the recall rate.

When CNN is applied to sequence annotation, the neurons in the final layer may only get a small part of the original input text. For NER, every word in the entire input sentence may have an impact on the recognition of the current position, which is the so-called long-distance dependency problem. In order to improve CNN's use of full sentence information, F. Yu and Koltun (2015) proposed the dilated CNN model. The filters of CNN all act on a continuous area of the input matrix, continuously sliding for convolution. The dilated CNN adds a dilation width to this filter. When it is applied to the input matrix, it will skip all the input data in the middle of the dilation width. The size of the filter itself remains unchanged, so that the filter obtains the data on a broader input matrix, which looks like it is "expanded".

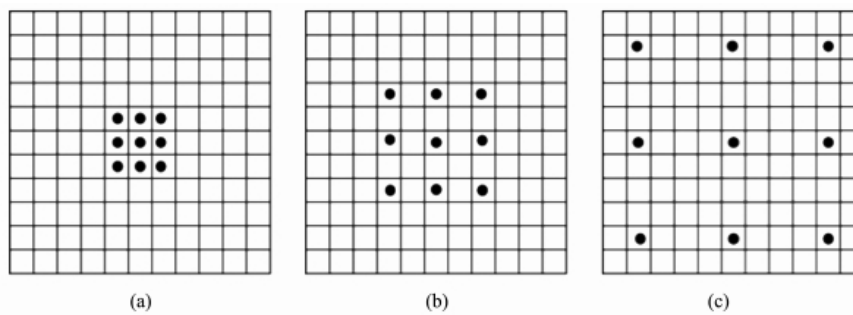


Figure 2.5: Dilated convolution diagram

Strubell et al. (2017) introduced dilated convolution into the field of natural language processing and proposed the IDCNN model. In IDCNN, the perception range increases exponentially as the number of layers increases, but the parameters only increase linearly, so that the perception domain can quickly cover all input sequences. The model is to stack 4 expansion convolution blocks of the same size together, and the expansion width in each expansion convolution block is 1, 1, and 2 layers. Input sentences into IDCNN, extract features through the convolutional layer, and connect to the CRF layer through the mapping layer. Finally, the sequence labeling result is obtained.

The NER method based on the neural network structure inherits the advantages of the deep learning method without a lot of artificial features. But the disadvantage is that it relies on a large amount of labeled data, and it is best to have high-quality dictionary features. For the problem of a small number of labeled training sets, transfer learning and semi-supervised learning should be the focus of future research.

2.2.6 Syntactic Parsing

Syntactic Parsing is a basic task in natural language processing. Its task is to analyze the sentence structure syntax tree given a sentence. The tasks of syntactic analysis can be divided into the following three categories:

- Syntactic structure parsing, also known as phrase structure parsing, or constituent syntactic parsing. The role is to identify the phrase structure in the sentence and the hierarchical syntactic relationship between the phrases.
- Dependency analysis, also known as dependency syntactic parsing, or dependency analysis for short, is used to identify the interdependence between vocabulary in a sentence.
- Deep grammar syntax analysis, that is, the use of deep grammar, such as Lexicalized Tree Adjoining Grammar (LTAG), Lexical Functional Grammar (LFG), Combinatory Categorical Grammar (CCG), etc. Perform in-depth syntactic and semantic analysis.

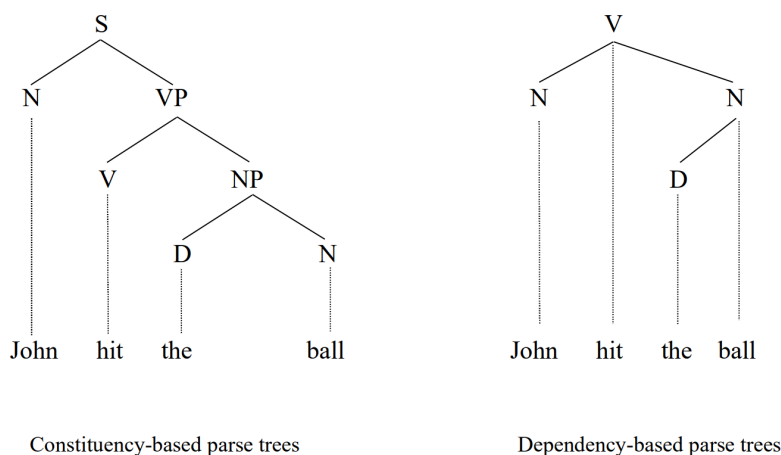


Figure 2.6: Example of syntactic structure parsing and dependency syntactic parsing

Constituent structure parsing

We first discuss constituent structure parsing. Syntactic structure analysis refers to judging whether the input word sequence (usually a sentence) conforms to the grammar of a specific language and obtains the syntactic structure of a grammatical sentence. The syntactic structure is generally represented by a tree-like data structure, usually called a parsing tree, as shown on the left side of Figure 2.6. Usually there are three

tasks for syntactic structure analysis: 1. Determine whether the input string belongs to a certain language; 2. Eliminate ambiguities in terms of morphology and structure in the input sentence; 3. Analyze the internal structure of the input sentence, such as component composition, contextual relations, etc. If a sentence has multiple structural representations, the syntactic analyzer should analyze the most likely structure of the sentence. In the actual operation process, usually the system already knows which language the sentence to be analyzed belongs to, so generally only the latter two tasks are considered.

When we know the part-of-speech tagging of the text, given the rules of formula 2.1, we can get the grammar tree in Figure 2.6.

John(Noun) hit(Noun/Verb) the(Determiner) ball(Noun/Verb).

$$\begin{aligned}
 S &\rightarrow N VP \\
 VP &\rightarrow V NP \\
 NP &\rightarrow D N
 \end{aligned}
 \tag{2.1}$$

Generally speaking, the construction of a syntactic analyzer needs to consider two parts of work: The first part is the formal representation of grammar and the description of vocabulary information. Formal grammar rules constitute a rule base, and vocabulary information (including part of speech, verb valence and head word information, etc.) is provided by the dictionary. The rule base and dictionary form the knowledge base of syntactic analysis. The second part of the work is to analyze the design of the algorithm.

Grammar formalism belongs to the category of syntactic theory research. In the early research on natural language processing, context-free grammar (CFG) and constraint-based grammar(or unification grammar) were widely used. Unification grammar has now formed a type of formal representation widely used in natural language processing. Samuelsson and Wiren argued that, compared with augmented transition networks (ATNs), from the perspective of grammar engineering and grammar reusability (reusability), the constraint-based formal method is more advantageous, and this formal method has been more extensive application(2000). Commonly used constraint-based syntaxes are:

- Functional unification grammar(Kay, 1984).
- Tree-adjointing grammar(Joshi, Levy, and Takahashi, 1975).
- Lexical-functional grammar(Bresnan, Kaplan, et al., 1982).
- Generalized phrase structure grammar(Gazdar et al., 1985).

- Head-driven phrase structure grammar(Pollard and Sag, 1994).

The specific methods of syntactic structure parsing can be divided into two categories: rule-based analysis methods and statistical-based analysis methods.

The rule-based syntactic structure analysis method is to organize grammatical rules manually, build a grammatical knowledge base, and realize the elimination of syntactic structure ambiguity through conditional constraints and checks(Allen, 1995). Cocke-Younger-Kasami parsing was proposed by Kasami (1965) and Younger (1967). The CYK algorithm is essentially a dynamic programming algorithm. Then few decades, researchers have proposed various syntactic analysis algorithms. Earley parsing by Earley (1970), Chart parsing by Kay (1985), Shift-reduction parsing by Aho, Sethi, and Ullman (1986), Generalized LR parsing by Tomita (1985) and Left-corner parsing by Rosenkrantz and Lewis (1970). Some of these algorithms are designed to be used in compilers, and some are suitable for NLP. Researchers have done a lot of improvement work on these algorithms. Socher et al. (2013) proposed the combined vector grammar (CVG), which applied the recurrent neural network to the component syntax analysis, and gave each phrase structure a vector representation.

According to the difference in the construction direction of the syntactic analysis tree, models can be divided into three categories: top-down analysis method, bottom-up analysis method and combined analysis methods. The top-down analysis algorithm implements the process of forward derivation of rules. The analysis tree is continuously constructed from the root node, and finally forms the leaf node of the analysis sentence. The bottom-up analysis algorithm's realization process is just the opposite. It starts from the sentence tag string, executes the process of continuous reduction, and finally forms the root node. The combination method is mainly to simulate people's understanding habits, partially execute the top-down method, and then reduce. The main advantage of the rule-based syntactic structure analysis method is that the analysis algorithm can use hand-written grammatical rules to analyze all possible syntactic structures of the input sentence. For specific fields and purposes, the use of hand-written targeted rules can better deal with some ambiguities and some extra-grammatical phenomena in the input sentence. However, the rule analysis method also has some shortcomings: For a medium-length input sentence, it is very difficult to analyze all possible sentence structures using large-coverage grammatical rules. The complexity of the analysis process often makes the program impossible to implement; Even if all possible structures of sentences can be analyzed, it is difficult to achieve effective disambiguation in the huge set of syntactic analysis results, and select the most likely analysis results; Hand-written rules are generally subjective, and for practical application systems, it is often difficult to cover all complex languages in large areas; Writing rules by hand is a complex task with a large

Parser	Model	Language	Publications
Collins Parser	PCFG based lexicalized probabilistic model	English	Collins, 1997
Bikel Parser	PCFG based lexicalized probabilistic model	Multilingual	Bikel and Marcus, 2004
Charniak Parser	PCFG based lexicalized probabilistic model	Multilingual	Charniak and Johnson, 2005
Berkeley Parser	PCFG based unlexicalized probabilistic model	Multilingual	Petrov and Klein, 2007
Stanford Parser	PCFG based unlexicalized probabilistic model	Multilingual	Klein and C. D. Manning, 2003

Table 2.1: PCFG phrase structure analyzer

amount of work, and the rules written are closely related to specific fields, which is not conducive to the transplantation of syntactic analysis systems to other fields(Oepen and Carroll, 2000).

The difference between the syntactic analysis method of natural language and the syntactic analysis method of programming language is that the coverage of prior knowledge of the syntactic analyzer in natural language processing is always limited. The syntactic parser may always encounter new language phenomena that have not been learned, and this is impossible for computer programming languages. When people use computer programming languages, all expressions must obey the requirements of the machine. It is the process of a person obeying the machine. This process is a mapping process from the infinite set of the language to the finite set. Therefore, some parsers that run well in the compiler do not perform well in natural language processing.

In view of the many limitations of rule-based syntactic analysis methods, researchers began to explore **statistical syntactic analysis** methods in the mid-1980s. In view of the many limitations of rule-based syntactic analysis methods, researchers began to explore statistical syntactic analysis methods in the mid-1980s. At present, the most researched statistical syntax analysis method is grammar-driven, and its basic idea is to use generative grammar to define the grammatical rules of the language being analyzed. The distribution of various language phenomena observed in the training data is coded together with grammatical rules in the form of statistical data. In the process of syntactic analysis, when ambiguity is encountered, statistical data is used to sort or select various analysis results.

The probabilistic context-free grammar (PCFG) phrase structure analysis method is currently a more popular grammar-driven statistical syntactic analysis method. The models used in this method mainly include lexicalized probabilistic model and unlexicalized probabilistic model. Table 2.1 lists five representative open source phrase structure analyzers.

A syntactic analysis method based on Probabilistic Context-Free Grammar (PCFG) was proposed in the

1980s. This method not only has the characteristics of a rule method, but also uses probability information. PCFG is an extension of CFG, and the rule representation of PCFG is: $A \rightarrow \alpha, p$, where A is a non-terminal symbol and p is the probability that A derives α . That is, $p = P(A \rightarrow \alpha)$, the probability distribution must meet the following conditions:

$$\sum_{\alpha} p(A \rightarrow \alpha) = 1 \quad (2.2)$$

Next we use rules and text as examples:

$$\begin{array}{llll}
 S \rightarrow NP VP & 1.0 & NP \rightarrow NP PP & 0.4 \\
 PP \rightarrow P NP & 1.0 & NP \rightarrow He & 0.2 \\
 VP \rightarrow V NP & 0.65 & NP \rightarrow Lily & 0.06 \\
 VP \rightarrow VP PP & 0.35 & NP \rightarrow flower & 0.16 \\
 P \rightarrow with & 1.0 & NP \rightarrow books & 0.186 \\
 V \rightarrow met & 1.0 & &
 \end{array} \quad (2.3)$$

According to the grammar of formula 2.3, the sentence He met Lily with flowers has two possible syntactic structures, as shown in Figure 2.7.

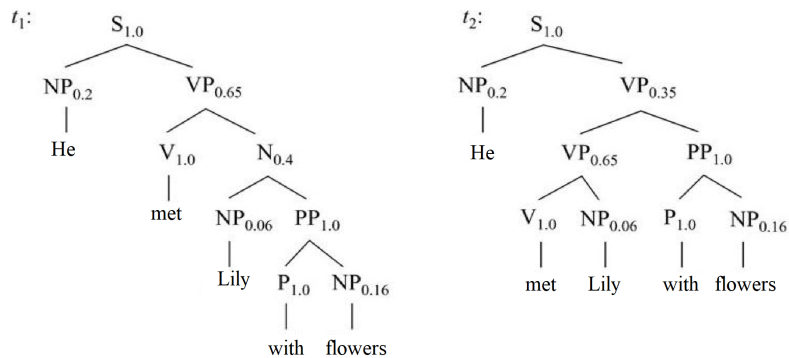


Figure 2.7: Two different PCFG parse trees of He met Lily with flowers.

In the PCFG-based syntactic analysis model, it is assumed that the following three conditions are met:

- Place invariance: The probability of a subtree does not depend on the position of the word contained in the subtree in the sentence;

- Context-free: the probability of the subtree does not depend on words outside the control range of the subtree;
- Ancestor-free: The probability of the subtree does not depend on the ancestor node of the derived subtree.

Therefore, the probabilities of the two subtrees in Figure 2.6 are:

$$\begin{aligned}
 P(t_1) &= 1.0 \times 0.2 \times 0.65 \times 1.0 \times 0.4 \times 0.06 \times 1.0 \times 1.0 \times 0.16 \\
 &= 0.0004992 \\
 P(t_2) &= 1.0 \times 0.2 \times 0.35 \times 0.65 \times 1.0 \times 0.06 \times 1.0 \times 1.0 \times 0.16 \\
 &= 0.0004368
 \end{aligned}
 \tag{2.4}$$

Dependency syntactic parsing

Different from syntactic structure parsing, dependency syntactic parsing mainly describes the dependencies between various words. It also points out the syntactic collocation relationship between words, and this collocation relationship is related to semantics. The framework that uses the dependency relationship between words to describe language structure can also be called dependence grammar.

According to Figure 2.6, compared with component syntax analysis, dependency syntax analysis analyzes the subject predicate and other components of a sentence more directly. Another point is that in the result of dependency syntax analysis, the relationship between words is more direct. For example, ball and hit are the direct relationship between predicate and object. Syntax establishes a subordination relationship composed of head words and dependency words. Predicate verbs are the core of the sentence and dominate other components. Predicate verbs are not be dominated by any other components.

Hays (1964) researched computer linguistics system at RAND. Its grammar part is based on Lucien Tesnière (1959) dependent grammar. Robinson (1970) proposed four axioms of dependency grammar:

- *One and only one element is independent;*
- *All others depend directly on same element;*
- *No elements depends directly on more than one other; and*
- *If A depends directly on B and some element C intervenes between them(in linear order of string), then C depends directly on A or on B or on some other intervenes element.*

These four axioms are equivalent to the formal constraints on the dependency tree: single parent node, connectedness, acyclic and Projective, so as to ensure that the result of sentence dependency analysis is a rooted tree structure. Here we need to discuss Projective, we first define the edge (arc) meets the condition of Projective: suppose this edge connects w_i and w_j , where $i < j$, and its central word is head (which is one of w_i and w_j). If there is a path from head to it for every word between i and j , then this edge satisfies the Projective property. If each edge of a dependency tree satisfies the Projective property, then the tree satisfies the Projective property, as shown in Figure 2.8. If a dependency tree does not satisfy the Projective property, then at least two edges of the drawn dependency tree must intersect, as shown in Figure 2.10(Jurafsky, 2000).

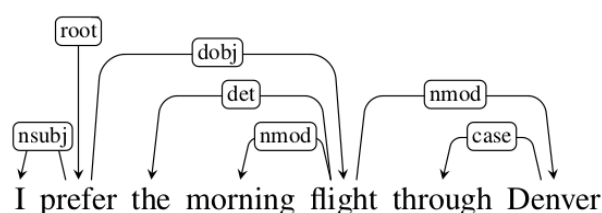


Figure 2.8: Projective dependency tree(Jurafsky, 2000)

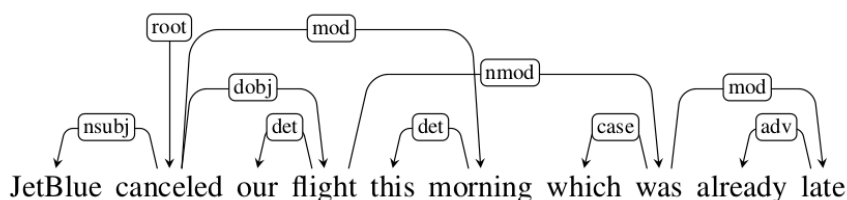


Figure 2.9: Non-projective dependency tree(Jurafsky, 2000)

For head word H and dependent word D , Zwicky (1985) and Hudson (1984) put forward some basic principles about dependency(Nivre, 2005).

1. H determines the syntactic category of C and can often replace C .
2. H determines the semantic category of C ; D gives semantic specification.
3. H is obligatory; D may be optional.
4. H selects D and determines whether D is obligatory or optional.
5. The form of D depends on H (agreement or government).

6. *The linear position of D is specified with reference to H .*

Hudson also proposed most of the standards met by some typical examples of head word. Some authors also pointed out the need to distinguish between different kinds of dependencies. Mel’cuk et al. (1988) purposed three category of word forms in a sentence: morphological, syntactic and semantic.

The mainstream algorithms for dependency parsing are transition-based, mainly including Arc-Standard algorithm and Arc-Eager algorithm. The conversion-based algorithm has a stack and a word queue, and a data structure for storing the confirmed head dependencies. At the same time, a classifier is needed to determine what operations need to be performed at the moment (Nivre, 2004; Collins and Roark, 2004).

Arc-Standard Dependency Parsing has the following three steps where w_i and w_j are arbitrary word tokens:

- The transition **Left-Reduce** combines the two topmost tokens on the stack, w_i and w_j , by a left-directed arc $w_j \rightarrow w_i$ and reduces them to the head w_j .
- The transition **Right-Reduce** combines the two topmost tokens on the stack, w_i and w_j , by a right-directed arc $w_i \rightarrow w_j$ and reduces them to the head w_i .
- The transition **Shift** pushes the next input token w_i onto the stack.

There is only one root in the initial state stack, and all words in the queue, and then loop until the state is the end state.

Unlike Arc-Standard, the Arc-Eager algorithm compares not the two elements at the top of the stack each time, but the element at the top of the stack and the beginning of the word queue. Arc-Eager has one more available operation to pop the top element of the stack. The four steps are following where w_i and w_j are arbitrary word tokens:

- The transition **Left-Arc** adds an arc $w_j \xrightarrow{r} w_i$ from the next input token w_j to the token w_i on top of the stack and pops the stack.
- The transition **Right-Arc** adds an arc $w_i \xrightarrow{r} w_j$ from the token w_i on top of the stack to the next input token w_j , and pushes w_j onto the stack.
- The transition **Reduce** pops the stack.
- The transition **Shift** pushes the next input token w_i onto the stack.

Both Arc-Standard and Arc-Eager have the following properties: the dependency tree obtained by the arc-eager transformation algorithm satisfies the projective nature; for each projective dependency tree, there is at least one sequence of operations to get the dependency tree. In addition, for a sentence with a length of n words, the length of the operation sequence is at most $2n$.

The biggest problem of the transfer algorithm is to determine which operation should be taken under the current stack and queue status. The algorithm needs to output a sequence of operations for dependency analysis based on a given sentence. The simplest idea is the greedy algorithm, which selects the best operation according to the current configuration each time, and then executes this operation until it enters the termination state. The simpler to improve the greedy algorithm is to use Beam Search, while the more complex method is to use the results predicted by the dynamic model to execute. D. Chen and C. D. Manning (2014) introduced a way to improve the greedy algorithm using deep learning, uses just a small number of dense features, and it can work very fast.

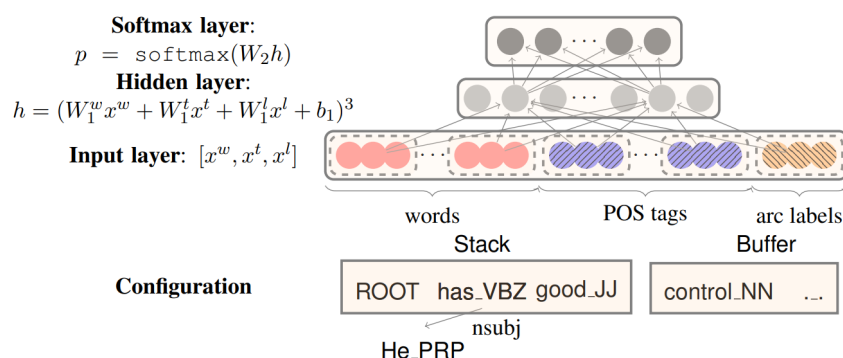


Figure 2.10: Neural network architecture of Dependency parser by D. Chen and C. D. Manning (2014)

2.2.7 Reference Resolution

The definition of Reference Resolution is identify all mentions that refer to the same real word entity. Mention refers to nouns, noun phrases, pronouns, etc. that appear in a sentence. In natural language, in addition to the basic single reference, there are also a large number of nested references. There are many application scenarios for reference resolution, for example:

- Full text understanding: There are a lot of references in natural language articles.
- Machine translation: Turkish does not distinguish between male and female. When translating to

English, it must be resolved by reference

- Information extraction: In the process of extracting entity relationships, it is necessary to clarify the reference relationship and replace the entity.

Use a simple sentence as an example of Reference Resolution. "He" refers to "David", "it" refers to "concert".

David went to the concert. He said it was an amazing experience.

Reference Resolution can usually be divided into three categories:

- Anaphora Resolution refers to the problem of determining which noun phrase the dominant pronoun points to in the text. The pronoun is called anaphor, and the noun phrase it points to is called antecedent. According to the position between Anaphor and Antecedent, it can be divided into Anaphora and Cataphora.
- Zero Anaphora Resolution, A special kind of resolution for the phenomenon of Zero Anaphora in pronoun resolution. In the text, the parts that users can infer from the context are often omitted, and the omitted parts are represented by Zero Pronoun. Zero Pronoun anaphora to certain linguistic unit in previous.
- Co-reference Resolution is the process of dividing words that refer to the same real-world objective entity (Entity) in a text into the same equivalence set. The divided words are called expressions or Mention, and the equivalent set formed is called the Coreference Chain.

Based on the rules of linguistics, Hobbs (1978) proposed a very complex set of rules to resolve pronoun references. The algorithm is written according to the intuition of the English language, and can get an accuracy of 80%, and is sometimes used as a feature of other machine learning classifiers. Winograd and other old-school AI scholars believe that to resolve reference, an external knowledge base is necessary. That is, Knowledge-based Pronominal Coreference. But its performance is not as good as manual rules(Winograd, 1972). Mitkov (1998) proposed a pronoun resolution method based on limited linguistic and domain knowledge. He designed some indicators to evaluate each potential anaphor candidate.

In the study of rule-based anaphora resolution, researchers have noticed the omission of pronouns. Kameyama (1986) studied Zero Anaphora in Japanese and compared the processing differences with English text. she add another dimension called the "speaker identification" to deal with this particular zero pronoun problem.

Nakaiwa and Shirai (1996) uses some specific rules for zero pronoun scenarios, called Deictic Resolution. Seki, Fujii, and Ishikawa (2002) supposed using probabilistic model framework to solve zero pronoun. Zhao and Ng (2007) introduced machine learning into the scene of Chinese zero pronouns, and solved the identification of zero pronouns, not just resolution. Yin et al. (2018) applied Deep Reinforcement Learning to solve the zero pronoun problem and improves the efficiency of using contextual information.

With the development of Coreference Resolution, the algorithms of Anaphora Resolution and Coreference Resolution in many scenarios are no longer distinguished, although the reference relationship is different, as shown in Figure 2.11.

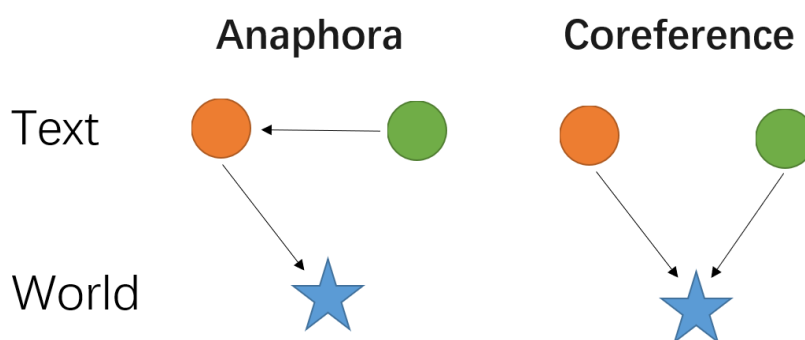


Figure 2.11: Anaphora vs. Coreference Resolution

There are three models that are most used in Coreference Resolution. The Mention Pair models method transforms the problem of referential resolution into a binary classification problem. Traverse the sentence from left to right. Whenever a reference is found, it is regarded as a pair with each reference found before, and the classifier is used to determine whether the pair refers to the same entity, and if so, connect them. The loss of two classification is cross entropy(Luo et al., 2004). This model has the risk of over-cluster. If one pair is wrong, all references originally belonging to the two clusters will be classified into one category. The method of Mention Ranking is to evaluate whether each reference is in the same cluster with all previous references at the same time, normalize with softmax, and find the antecedent with the highest probability(Denis and Baldrige, 2008). Entity-Mention Model finds out all entities and their conversation context. According to the conversation context clustering, mentions in the same cluster are resolved into the same entity(X. Yang et al., 2008).

Clark and C. D. Manning (2016) uses Reinforcement Learning technology to optimize the Mention-ranking scoring process. Enter the neural network with word embedding and a small amount of artificial features, a reward-rescaled max-margin objective, and good results are obtained.

2.3 Knowledge Graph

2.3.1 Triples Extraction

The main task of triple extraction is to, given a sentence text, extract two entities in the sentence and the relationship between the entities to form a triple (s, p, o) , also called Basic Element (Hovy, C.-Y. Lin, and Zhou, 2005). s is subject, which represents the main entity, o is object, which represents the object entity, and p is predicate, which represents the relationship between the two entities. There are two main ways to deal with triple extraction. The first is the pipeline method, which splits the relationship extraction into two steps like a pipeline, named entity recognition and then performs relationship recognition. The second is the end-to-end method, also known as the joint extraction method. Only input a sentence, transform it into a vector sequence representation, and then extract entity relationship triples from it.

Triple extraction has the following usage scenario (Jurafsky, 2012):

- Create a new structured knowledge base and enhance the existing knowledge base.
- Construct knowledge graph in vertical fields: medical, chemical, agriculture, education, etc.
- Support upper-level applications: question and answer, search, reasoning, etc. For example, for such a question: *The granddaughter of which actor starred in the movie "E.T."?* It can be transfer to:

(acted-in ?x "E.T.") && (is-a ?y actor) && (granddaughter-of ?x ?y)

In the early information extraction work, most of the triple extraction uses the pipeline method. Hearst (1992) used the lexico-syntactic patterns method to obtain a ordered pair representing the correlation, but did not concern about the difference of the relationship. Riloff and Lehnert (1994) uses the relevancy index to extract the relevant information in the text, which is a triple of the form: (signature, slot filler, case outline). This algorithm is more suitable for strong keywords as features and is not sensitive to context. Turney (2005) uses Latent Relational Analysis to measure semantic distance. Pantel and Pennacchiotti (2006) present Espresso, a weakly-supervised algorithm for harvesting semantic relations. Gamallo, Garcia, and Fernández-Lanza (2012) published an article that fully described the process of multilingual Open Information Extraction in pipeline mode, and used Coreference Resolution to obtain in-depth grammatical information. It also proves that other steps using NLP can effectively improve the extraction accuracy of triple.

One of the difficulties in relation extraction is feature construction and lack of labeled data. An early study,

Hearst (1992) proposed to use the bootstrapping approach to discover patterns in the corpus using basic seed examples, and then use the obtained patterns to find more examples. Mintz et al. (2009) proposed the use of Distant Supervision for relation extraction, and put forward the hypothesis: *If two entities participate in a relation, all sentences that mention these two entities express that relation.* But this hypothesis has some counterexamples, and feature extraction using NLP technology often produces errors in previous steps, and errors accumulate and propagate in a multi-step NLP system. In 2014, Zeng et al. used convolutional neural networks to automatically extract features, and in the following year, Zeng and his colleague used multi-instance learning and improved CNN to Piecewise Convolutional Neural Networks (Zeng, Liu, Lai, et al., 2014; Zeng, Liu, Y. Chen, et al., 2015). The neural network structure is shown in Figure 2.12.

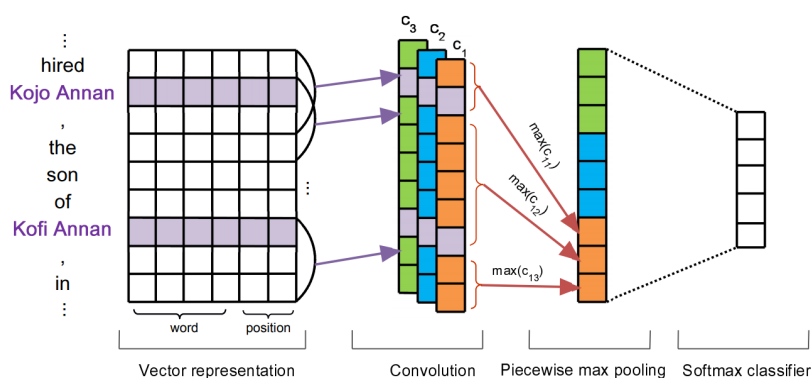


Figure 2.12: The architecture of PCNNs (Zeng, Liu, Y. Chen, et al., 2015)

Y. Lin et al. (2016) introduced the attention mechanism in his article to solve the problem of using only one instance to represent a relation. The introduction of selective attention mechanism is also used to reduce label errors. Miwa and Bansal (2016) proposed a relationship extraction model based on an end-to-end neural network. This model uses bidirectional Long-Short Term Memory (LSTM) and tree LSTM to simultaneously model entities and sentences. The design of the entire system is shown in the figure 2.13.

The results obtained by relation extraction often fall into the following categories: upper and lower relations or called IsA relations; attribute relations, or called meronymy; and Co-ordinate term (co-hyponym) relations; Instance-of relations. For example:

- IsA: Giraffe **IS-A** ruminant **IS-A** ungulate **IS-A** mammal **IS-A** vertebrate **IS-A** animal
- Instance-of: New York **instance-of** city
- Meronymy: engine **is-part-of** car

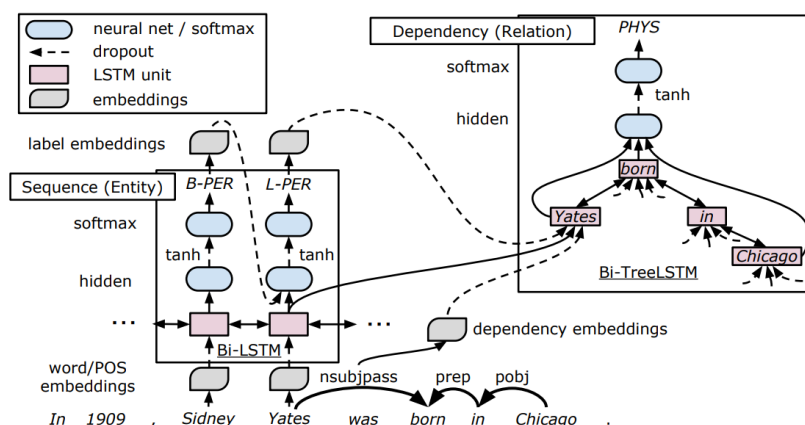


Figure 2.13: bidirectional sequential and bidirectional tree-structured LSTM-RNNs(Miwa and Bansal, 2016)

- Co-ordinate term: Tokyo, Paris, London

2.3.2 Graph building

When we have the triples obtained from the analysis of the text, they can be used to construct a knowledge graph. As the key technology of this article, first we need to discuss what needs a knowledge graph and what is a knowledge graph. Computers are suitable for processing data, but the semantic information of natural text is not structured data before processing. At the same time, computers need to understand human expert knowledge to solve some problems. Therefore, on the basis of semantic network and expert system, researchers have proposed the definition of knowledge graph: A knowledge graph consists of a set of interconnected typed entities and their attributes(Pan et al., 2017). Generally speaking, the knowledge graph is composed of pieces of knowledge, and each piece of knowledge is represented as an SPO triple(Subject-Predicate-Object). The definition of triple here is the same as above, so the method of extracting triple is also the same.

In the 1970s, Expert Systems, as an important branch of artificial intelligence, refers to computer programs that use knowledge and reasoning to solve problems that can only be solved with the help of human expert knowledge. Expert system generally consists of two parts: knowledge base and reasoning engine. Human experts provide knowledge, and then map and store this explicit knowledge in the knowledge base for reasoning. Lenat, Prakash, and Shepherd (1985) set up an excellent program named CYC. The goal of the CYC is to collect common sense knowledge and structure it into a comprehensive knowledge base. Cyc not only includes knowledge, but also provides a lot of reasoning engines, supporting deductive reasoning and

inductive reasoning. Under Miller's guidance, Princeton University began to build and maintain an English dictionary called WordNet. In WordNet, nouns, verbs, adjectives, and adverbs are grouped according to cognitive synonyms, called synsets, and each synset represents a certain concept. Synsets are linked by concept semantics and lexical relations(G. A. Miller, 1998).

These artificially constructed knowledge bases require a huge amount of time and manpower, and the coverage is not large enough. In 1998, Tim Berners Lee(Inventor of the World Wide Web) proposed the Semantic Web. By adding metadata to documents on the WWW (HTML documents etc.), the semantics of Internet content can be understood by computers(Berners-Lee, Hendler, and Lassila, 2001). Berners-Lee (2006) then proposed the concept of Linked Data, encouraging everyone to make the data public and follow 3 principles. Many structured knowledge bases use Wikipedia to build knowledge bases. DBpedia and Yago belong to this type of knowledge base. Yago is a project started by Max Planck Institute in Germany in 2007. It combines the knowledge of WordNet and Wikipedia and supplements the knowledge of hypernyms of entities in Wikipedia, thereby obtaining a large-scale, high-quality, high-coverage knowledge base.(Suchanek, Kasneci, and Weikum, 2007) DBpedia uses mapping technology and extraction templates to achieve the unity and consistency of knowledge description. At the same time, DBpedia has a lot of cross-language knowledge, and developed DBpediaLive to keep in sync with Wikipedia(Auer et al., 2007).

All of the above are based on online encyclopedia data, and the format is relatively structured, but most of the information on the Internet is presented in the form of unstructured free-form text. Banko et al. (2007) proposes Open Information Extraction to directly extract entity relationship triples from large-scale natural language texts. Never-ending learning proposed by Tom Mitchell is a different method from traditional machine learning. It is similar to the seed data bootstrap method proposed by Hearst above. Never-Ending Language Learning (NELL) constantly reads (extracts) knowledge facts from web pages every day to fill the knowledge base, and removes incorrect knowledge facts that previously existed in the knowledge base. Each knowledge has a certain degree of confidence and Reference source(T. Mitchell et al., 2018). Basically, all above construction of the knowledge graph include steps such as entity recognition, relationship extraction, and co-reference resolution etc. Like Figure 2.14, in the final step, the obtained entity relationship will be co-reference resolved and disambiguated, and linked into a directed acyclic graph.

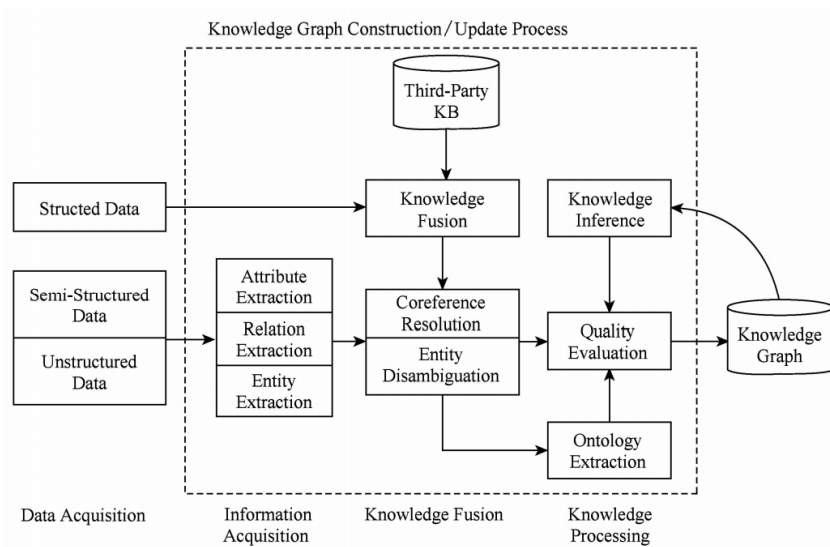


Figure 2.14: Technical architecture of knowledge graph

2.3.3 Relational Learning

Although according to the process in Figure 2.14, we get a knowledge graph with (s, r, o) relationship. But there are two problems with this result. The first is that the process of relationship extraction does not ensure that all entity associations are completely extracted; the second is that there are many unexpressed entity relationships hidden in natural language. Therefore, we need to supplement the missing relationship links through relationship prediction to obtain a complete knowledge map. Due to the large scale of many knowledge graphs, the algorithm needs to consider the parameter scale and computing performance.

We can mathematically express the entity relationship prediction problem as follows: For the knowledge graph composed of triples $G = \{(s, r, o)\} \subseteq E \times R \times E$, the objective is a scoring learning function $\psi : E \times R \times E \rightarrow \mathbb{R}$. Given an input triple $x = \{(s, r, o)\}$, its score $\psi(x) \in \mathbb{R}$ is proportional to the likelihood that the fact encoded by x is true (Dettmers et al., 2018).

Like the development of other NLP technologies, rule-based relational reasoning first appeared. From the perspective of basic reasoning concepts, the task of reasoning is divided into deductive reasoning, inductive reasoning and proof reasoning. According to the reasoning method, the method of reasoning is roughly divided into deterministic reasoning and uncertain reasoning. Deterministic reasoning is also called logical reasoning, and logical reasoning can accurately derive a unique conclusion under the pre-defined rules of experts. Typical logical reasoning methods include the rete algorithm based on forward connection (Forgy,

1989); GSAT and WALKSAT algorithms for solving boolean satisfiability problems (Selman, Kautz, B. Cohen, et al., 1993; Selman, Levesque, and D. Mitchell, 1992). Uncertain reasoning is also called probabilistic reasoning. It uses statistical learning and other methods to build a probability model according to a certain model based on existing experience and data. The typical probabilistic reasoning methods include: probabilistic graph model, probabilistic logical reasoning and association rule mining.

Deduction-based knowledge graph reasoning techniques mainly include: ontology-based reasoning method, logic programming-based reasoning method, query rewriting-based method and generation rule-based method. Graph structure-based reasoning uses graph theory related algorithms to perform knowledge graph reasoning. Among them, the classic reasoning algorithm is Path Ranking Algorithm, which uses the path between the entity nodes as features to perform link detection reasoning (Lao and W. W. Cohen, 2010). Reasoning based on rule learning, rules are defined by domain experts. However, because the rules need to be provided manually and the knowledge graph changes dynamically, rule reasoning is very limited in the application of large-scale knowledge graphs. In order to overcome this difficulty, experts have proposed an automated rule learning method. Galárraga et al. (2013) invented by AMIE in 2013, which is an automated rule learning algorithm based on Horn's rules. It evaluates the quality of the rules through SPARQL queries on the knowledge graph, thereby automatically managing the rule base.

The latest relational reasoning algorithms are basically based on deep neural networks. The neural network embeds the triple vector data into the matrix as input, and then predicts whether the triple is a usable relationship. Here are two related studies as an introduction. Y. Yang et al. (2016) proposed a position-encoding CNN algorithm based on dependency parsing tree in EMNLP 2016, challenging the end-to-end approach. Yang believes that the syntactic parsing tree is very valuable in the task of relation classification. Since the distance of the dependency path of the parsing tree is often smaller than the distance in the sentence, the dependency path between entities after pruning reduces a lot of noise information. They take the word vector and the position feature of the dependency tree of the word as the representation input, and a convolution window of CNN obtains the parent node and child node of the input word as its adjacent context each time. They use two convolution kernels, Kernel-1 and Kernel-2. Kernel-1 aims to extract features from the dependency tree at multiple levels, while Kernel-2 focuses on mining the semantic information between words which share the same parent. The framework is shown in Figure 2.15.

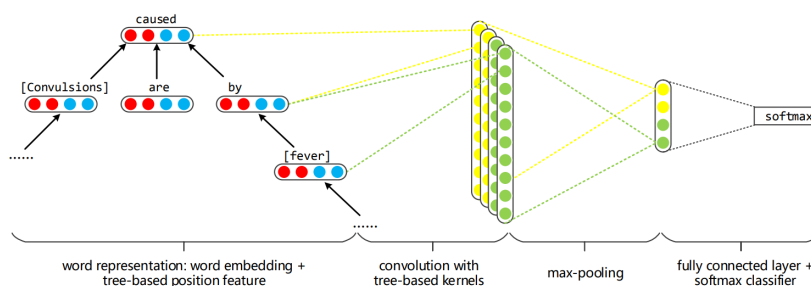


Figure 2.15: The framework of PECNN. The red and blue circles represent the word embeddings and tree-based position features of words. The yellow and green circles stand for the feature maps extracted by two kinds of convolution kernels respectively.(Y. Yang et al., 2016)

Dettmers et al. (2018) proposed the Convolutional 2D Embeddings method, which is an excellent end-to-end method. They divided the method into four steps, as shown in Figure 2.16. The first part is Embeddings, which embeds the entity $e1$ and the relation r as a vector and concatenates them as a matrix. Then use the convolution kernel to calculate the two-dimensional convolution on the cascaded matrix. The obtained feature matrix is flattened into a vector and projected through a fully connected layer in third step. Calculate the inner product of the entity embedding matrix and the vector obtained from the fully connected layer, and finally perform softmax evaluation. The model uses ReLU as a nonlinear activation function, and uses dropout and batch normalization. The algorithm runs swiftly and performs better in high in-degree graphs.

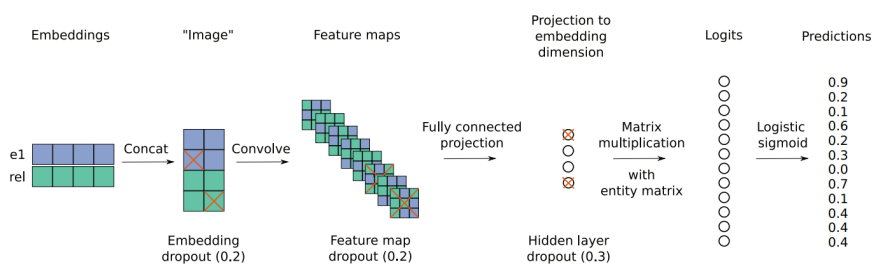


Figure 2.16: The ConvE model, by four steps(Dettmers et al., 2018)

The development of relational reasoning technology has brought progress in large-scale knowledge graphs, and these confirmed entity relationships can be used as training sets to enhance model performance. But how to evaluate the quality of the relationship obtained by reasoning is also a difficult problem. Since the knowledge extracted by the Open IE technology may have errors, the relationship obtained through knowledge reasoning is also not guaranteed. Due to the cumulative effect of errors, it is very necessary to

conduct quality inspection directly in the knowledge storage. In addition, the quality of each open source knowledge base also differs, and a feasible unified evaluation method is very necessary. When evaluating the REVERB system, Fader, Soderland, and Etzioni (2011) used artificially labeled triples obtained from 1,000 sentences as a training set to train a logistic regression model. This method can calculate the confidence of information extraction results, and also provides a enlightening idea for the knowledge quality scoring method.

Chapter Three

Experiments

3.1 Method Describe

According to the needs of evaluating the logic of essays in the research objectives, we first determine the standard of the strength of logic, that is, the number of expressions of entity relationships. First, we discuss how to obtain the entity relationship. This process is based on all the knowledge points mentioned in the previous article, and each step has a sequence and dependency relationship. Usually the steps we get to the ternary relationship depend on the extraction process as shown in Figure 2.14. According to the basic paper written by Angeli, Premkumar, and C. D. Manning (2015) of the Stanford OpenIE system I plan to use, this method does use several functions of Stanford CoreNLP, including dependency syntactic parsing and named entity segmentation. Figure 3.1 shows the running steps of the classic tool Stanford CoreNLP(C. D. Manning et al., 2014).

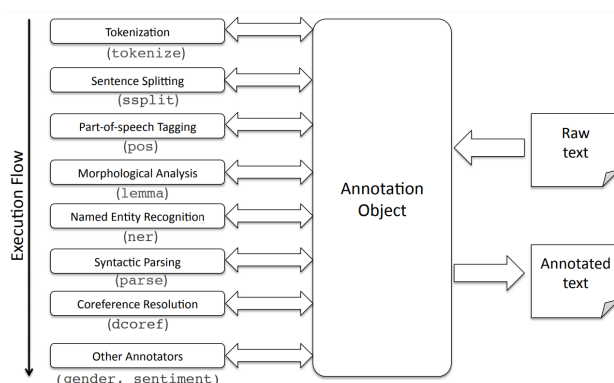


Figure 3.1: Overall system architecture of Stanford CoreNLP toolkit(C. D. Manning et al., 2014)

So when we get a natural language text, we need to perform Sentence Splitting and Tokenization first, then perform Part of speech Tagging according to the position and verb inflection in the text, and then obtain the prototype of the vocabulary through Morphological Analysis or it can be called Lemmatisation. For the text annotated with the above steps, you can proceed to Named Entity Recognition, Parsing, and Reference Resolution. The entities and the dependent parsing tree obtained by the above steps will be used by the Open IE system to extract entity relationships, and cluster and merge relationships through Coreference Resolution. We will discuss the need for relational reasoning in the experimental design section, and discuss the construction of knowledge graphs.

The experiment in this article only discusses one method that can be used to perform essay-based knowledge graph construction. Regarding the extraction of triples, there are many end-to-end methods that do not rely on parsing and NER. We divided the experiment into two parts. The first part is the qualitative method of step disassembly and analysis, discussing whether it is feasible to use the knowledge graph construction to score the logic of the essay; the second part is the quantitative verification under small batch data. Use the various indicators of the knowledge graph to score the essay.

3.2 Qualitative analysis

3.2.1 Experiment Aim

As the first part of the experiment in this article, the author hopes to demonstrate that it is feasible to extract the knowledge graph from the essay and use the knowledge graph to evaluate the logic. Based on

the summary of open information extraction in the previous article, and according to the research purpose of this article, the author proposes the following qualitative analysis goals.

- Demonstration information extraction work is feasible in essay score.
- Demonstrate whether reliance on syntactic analysis is more suitable for information extraction than component syntactic analysis.
- Prove that the Coreference Resolution can be used to merge entity relationships.
- Discuss which scoring method is more suitable for short essays.

The results of qualitative analysis will help us build a workflow that automatically evaluates the logic of the essay and determines which steps are needed. At the same time, we also need to discuss which indicator of the extracted knowledge graph is related to essays of different levels, such as size, number of edges, and longest chain. By obtaining the above necessary information, we will be able to construct an effective and credible essay logical automatic scoring method, which can help essay automatic scoring software to improve their performance in logical evaluation, and improve fairness and anti-cheating ability.

3.2.2 Experiment Design

As part of qualitative analysis, the author will describe the experimental design from the following perspectives.

Method

The subject of qualitative analysis uses case analysis. According to the research aim, specific case analysis is divided into text preprocessing methods, comparison of different parsing methods, relationship extraction, knowledge graph construction and scoring standards. This article will select an essay sample to discuss the construction of the knowledge graph and its dependent steps, compare the knowledge graphs obtained by essays with different scores, and discuss the choice of scoring criteria.

Data

Due to the use of case analysis based on technical operation steps as the main research method, this article uses the writing sample on IELTS-Blog as sample data for analysis. The 9-point and 5-point articles are mainly used. The data has removed irregular punctuation and misspellings in advance.

Evaluate

The evaluation of this qualitative analysis is mainly from the following dimensions. First, the rationality of the technical construction, that is, whether each step is an optimized solution, whether there are other feasible solutions, and horizontal comparison; second, the feasibility of the overall program is evaluated, including its operating cost and technical complexity; third, assess the robustness of the technology, whether it can deal with certain interference, wrong data, and cheating.

3.2.3 Experiment Procedure

This experiment uses the Stanford CoreNLP system as a demonstration, and the operating environment is configured as follows:

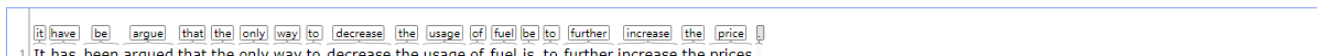
- CPU Intel Core i7-8809G 3.10Ghz 4 Core 8 Threads
- Memory 32 GB
- OS Ubuntu 20.04 LTS
- Java™SE Development Kit 8, Update 261
- Stanford CoreNLP 4.1.0

The case analysis uses the IELTS 9-point essay in the appendix as the test data. Use python language to program, call python-stanford-corenlp library to get the result. At the same time, for some steps, directly use CoreNLP's local visualization platform to display the results. The deployment guide for Stanford Core NLP system is at "<https://stanfordnlp.github.io/CoreNLP/corenlp-server.html>"

Pre-processing

The pre-processing part includes four steps, including segmentation, Tokenization, Lemmatisation and POS Tagging. The first step is segmentation. English uses spaces and hyphens to determine the boundaries between words, and uses periods to segment sentences. The second step is Lemmatisation, the result is shown in Figure 3.2. In the case, we can see: "argued" is reduced to "argue", "has" is reduced to "have", and "is" is reduced to "be". It is helpful for us to construct token or word embedding.

Lemmas:



```

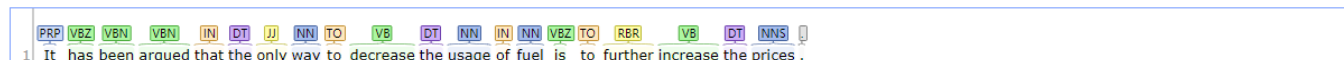
it have be argue that the only way to decrease the usage of fuel is to further increase the price .
1 It has been argued that the only way to decrease the usage of fuel is to further increase the prices .

```

Figure 3.2: Lemmatisation running example on Stanford CoreNLP.

Then we need to perform Tokenization and POS Tagging. The English part of CoreNLP provides the tokenize function of PTB-style. This part saves the correspondence between the token and the original text word. POS Tagging is based on Toutanova, Haghghi, and C. D. Manning (2005) paper and uses joint loglinear models to classify each word in the input sequence. The output result is shown in Figure 3.3. Each word is marked with part of speech, and this step is the basis of parsing.

Part-of-Speech:



```

PRP VBZ VBN VBN IN DT JJ NN TO VB DT NN IN NN VBZ TO RBR VB DT NNS .
1 It has been argued that the only way to decrease the usage of fuel is to further increase the prices .

```

Figure 3.3: Part-of-speech tagging running example on Stanford CoreNLP.

Parsing

The Dependency Parsing component of Stanford CoreNLP, whose technology was proposed by D. Chen and C. D. Manning, 2014. It mainly uses neural network technology to perform transition-based operation prediction on serialized data with POS Label. The `nlp.trees.DependencyScoring` class contains a tool for scoring common dependency analysis. The operating program code is in the appendix, and the visualized running result is shown in Figure 3.4. This result shows the dependence of words in a sentence, and successfully analyzes the attributive clause. At the same time, the dependency syntax also pointed out the core verb "invest" in the entire sentence, and obtained the projective parsing tree

Enhanced++ Dependencies:

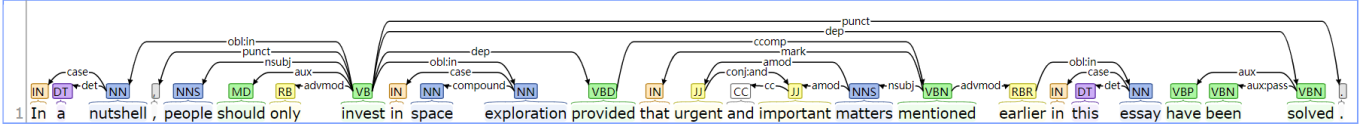


Figure 3.4: Dependency Parsing running example on Stanford CoreNLP.

Correspondingly, let’s compare the results of Constituent Parsing, as shown in Figure 3.5. The technology is based on Zhu et al. (2013) Shift-Reduce Constituent Parsing, which is a bottom-up method of building a parsing tree. The algorithm optimizes the problem of different action-sequence lengths, and incorporated a set of semi-supervised features. We can see that in the complete constituent parsing tree, the attributive clause has also been successfully parsed into the subtree in the lower right corner.

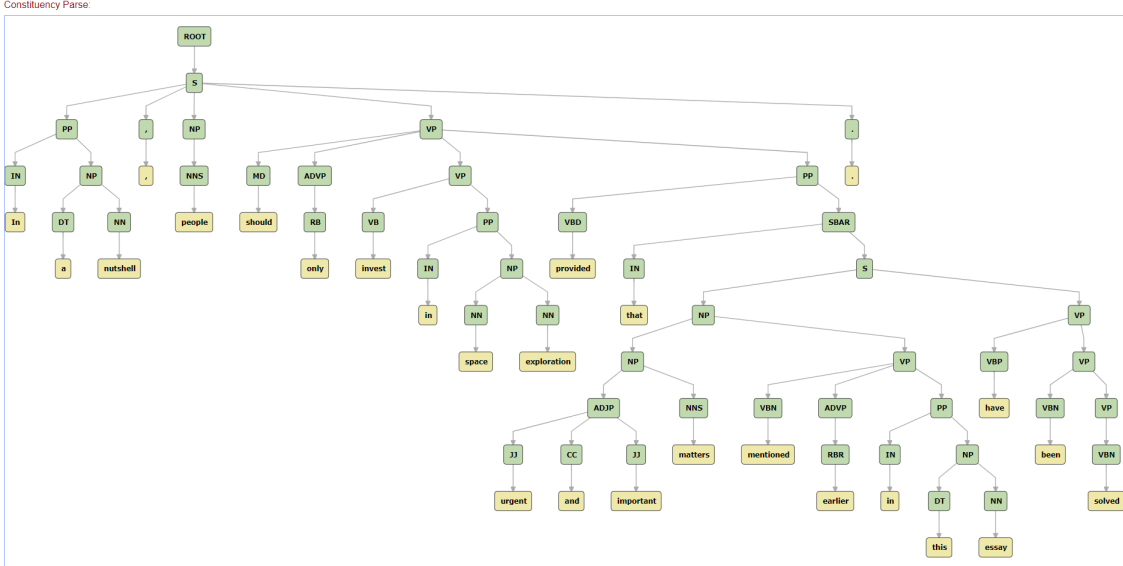


Figure 3.5: Constituent Parsing running example on Stanford CoreNLP.

Comparing the two different parsing methods, we can see that although Constituent Parsing provides more complete and detailed information, it is more difficult to find multi-level dependencies. Because when a child node of a Constituent Parsing Tree wants to find a dependency, it needs to traverse the child node of its parent node to get it. When we want to extract triples directly, we can get them directly from the edge of the Dependency Parsing Tree. In fact, Constituent Parsing is more used for clause and phrase extraction.

Relation Extraction

Relation extraction is one of the most difficult and challenging tasks in the project. This article will experiment with two different ways to extract relations. One group is the relationship extraction using Stanford CoreNLP's built-in Open IE system, and the other group is the end-to-end extraction method based on the neural network pre-training model. The technology used by Stanford Open IE is based on Angeli, Premkumar, and C. D. Manning (2015) proposed model for extracting short clauses and using a parsing tree. The running result is shown in Figure 3.6.

Open IE:

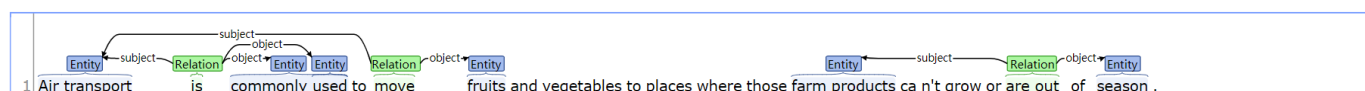


Figure 3.6: Open IE running example on Stanford CoreNLP.

As a comparison object, this article chooses a Deep based on BERT Transformers Model. Soares proposed this model, they build on extensions of Harris' distributional hypothesis to relations. However, this algorithm requires entity span identification as the preamble. Here, the author uses the basis mentioned in the NER part in Chapter 2 to preprocess it. The execution result of one sentence is shown in Figure 3.7. The results of this model on the SemEval 2010 Task 8 data set are as follows:

- Accuracy: 0.8048427991886409
- F1 Score: 0.7547323427640953
- Precision: 0.7515200648561006
- recall: 0.7579721995094031


```

Sentence: [E2]After eating the chicken[/E2] , [E1]he[/E1] developed a sore throat the next morning .
Predicted: Product-Producer(e2,e1)

Sentence: After eating the chicken , [E1]he[/E1] developed [E2]a sore throat[/E2] the next morning .
Predicted: Cause-Effect(e1,e2)

Sentence: [E1]After eating the chicken[/E1] , [E2]he[/E2] developed a sore throat the next morning .
Predicted: Product-Producer(e1,e2)

Sentence: [E1]After eating the chicken[/E1] , he developed [E2]a sore throat[/E2] the next morning .
Predicted: Cause-Effect(e1,e2)

Sentence: After eating the chicken , [E2]he[/E2] developed [E1]a sore throat[/E1] the next morning .
Predicted: Cause-Effect(e2,e1)

Sentence: [E2]After eating the chicken[/E2] , he developed [E1]a sore throat[/E1] the next morning .
Predicted: Cause-Effect(e2,e1)

```

Figure 3.7: An end-to-end model based on BERT.

Comparing the two models, we can find some problems. For the example sentence: *"Furthermore ,cultural globalization will result in one homogenous global culture where everyone speaks the same and thinks the same."* Stanford CoreNLP uses the results of dependency parsing, but extracts too many repetitive relationships, such as:

- 'subject': 'globalization', 'relation': 'will result in', 'object': 'one global culture'
- 'subject': 'globalization', 'relation': 'Furthermore will result in', 'object': 'one homogenous global culture'
- 'subject': 'globalization', 'relation': 'will result in', 'object': 'one homogenous culture'
- 'subject': 'globalization', 'relation': 'will result in', 'object': 'one culture'

We can see that only entity 3 is the correct relationship, and the relationships 1, 2, and 4 are all redundant.

The End-to-End model only got one result:

Sentence: Furthermore , [E1]cultural globalization[/E1] will result [E2]in one homogenous global culture where everyone speaks the same[/E2] and thinks the same.
Predicted: Cause-Effect(e1,e2)

It can be seen here that the algorithm is not ideal for entity recognition. Only when the entity is accurately labeled can the model predict the relationship more accurately. In the experimental results, half of the clause was redundantly classified as entity E2. As Stanford Open IE uses more grammatical information, the clause part is removed from the relationship.

Graph Building

The construction of knowledge graph mainly uses two methods, Coreference Resolution and directed graph construction. Coreference Resolution is to solve the problem of merging similar entities. Many essay writers will replace different expressions of the same meaning in their writing to achieve the purpose of expression diversification. However, this writing method makes it impossible for computers to directly use the same endpoint to connect different relationships, which brings difficulties to the construction of knowledge graphs. Here we continue to use the coreference module of Stanford CoreNLP. The execution result is shown in Figure 3.8. Stanford CoreNLP discovered the co-referential relationship between most pronouns and subjects with the same noun. However, some synonym substitutions have not been pointed out, which may be related to the lack of dictionary libraries.

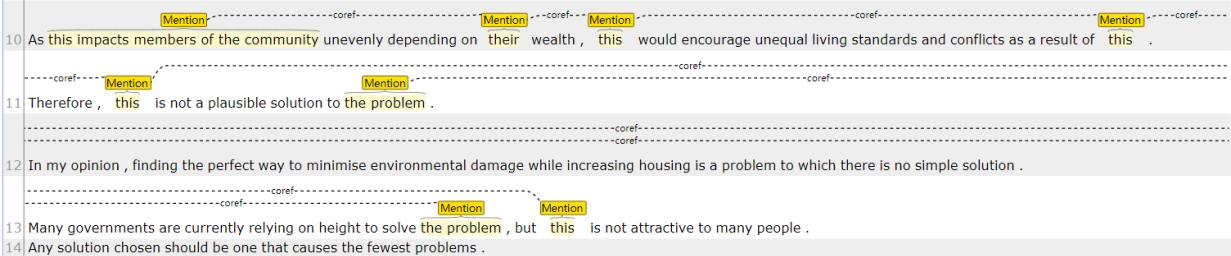


Figure 3.8: Coreference Resolution Result Example

Figure 3.9 shows a manually revised knowledge graph based on "Qualitative analysis example essay 2" in the appendix. Some problems can also be seen here: 1. The direction of the relationship caused by the passive voice of English is not clear; 2. The implicit entity relationship has not been discovered, such as "skyscrapers" and "height of buildings". In fact, this graph is not a strongly connected graph, which contains isolated subgraphs. Most of the knowledge graphs extracted directly from essays contain several subgraphs.

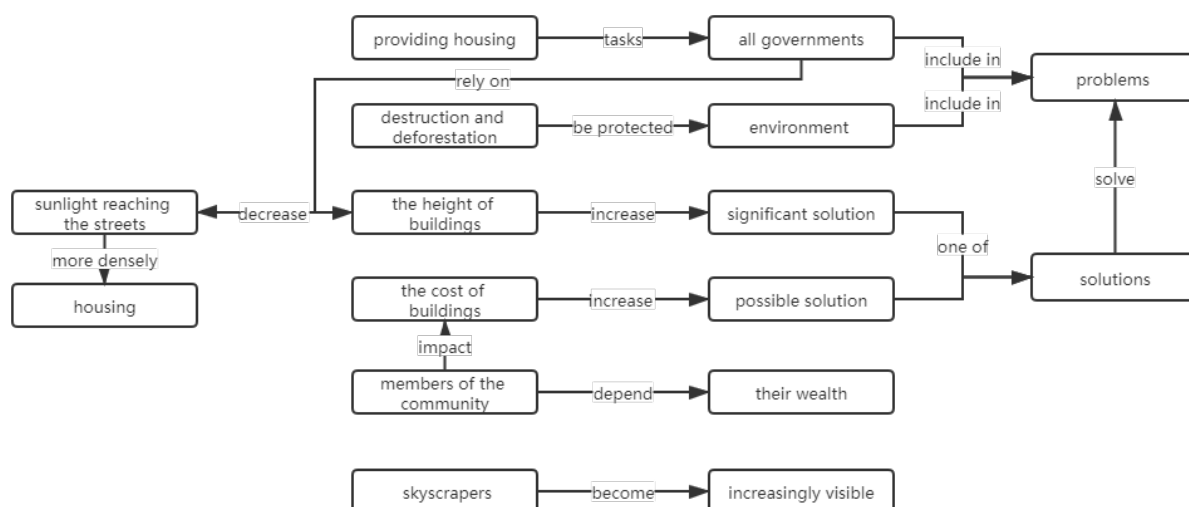


Figure 3.9: Knowledge Graph Example from essay

Scoring Standard

When we have obtained all the relationships and the knowledge graph obtained by using these relationships, we need to complete the research goal of this article, which is to score the logic of the essay. These knowledge graphs contain some indicators, including the size of the largest connected subgraph, the longest logical relationship link, and the total number of relationships. Here we analyze the advantages and disadvantages of using each indicator.

The total number of relationships index is the simplest and most intuitive, and it does not even need to build relationships into a graph. This indicator can directly reflect how many logical relations the writer has completed in total, and how many logical relations each sentence contains. At the same time, this is also the simplest algorithm for calculation, and does not require Coreference Resolution and merging of relations. But the shortcomings are also obvious. A logical relationship that is continuously connected into a network is obviously better than a scattered, unconnected logical relationship. Such an algorithm is also very easy to be deceived by a large number of simple sentences, and it can even repeat a relationship multiple times to increase the total number of logical relationships.

The size of the largest connected subgraph is an indicator initially selected by the author. The calculation of this indicator requires Coreference Resolution and node merging to obtain a complete knowledge graph. It directly reflects the total number of connected logical relations in the entire essay, and it is also easy to

use mature algorithms for the largest connected subgraph to solve problems, such as Tarjan’s algorithm. However, it is difficult to obtain an accurate knowledge map itself. It faces the following problems: 1. Repetition or omission in relation extraction; 2. Inaccurate entity recognition; 3. Coreference Resolution fails to merge synonyms; 4. Implied relationships cannot be discovered.

The longest logical relationship link is also an ideal indicator, but when the author chooses this indicator for evaluation, he found that there are circular relationships in the knowledge graph in some cases, especially when the active and passive grammars are not reversed. In fact, this problem should be transformed into finding the shortest logical chain length between any two entities, finding the two entities with the farthest distance, and using the distance as the length indicator. This scoring method can reflect the logical continuity of the essay writer, especially the closeness of each step from the starting point to the conclusion. The problem is that the solution of this link is relatively difficult. The author believes that it will be easier to solve the single relationship as undirected. At the same time, it also has all the problems of the maximum connected subgraph index.

3.2.4 Experiment Result & Analysis

The following conclusions can be drawn from the qualitative analysis of essay logic score using knowledge graph.

The existing relationship extraction methods are difficult to meet the accuracy and performance requirements for natural language texts. Not only traditional methods, but also deep neural networks. Natural language understanding is still a very difficult subject. It is feasible to use information extraction and knowledge graph construction to perform essay logical scoring, but the accuracy of triple extraction is required. At the same time, it is necessary to use Coreference Resolution to merge synonymous entities.

And we can see that the accuracy of the deep neural network for discriminating the relationship between correctly labeled entities has exceeded 80%, so if the accuracy of the NER part can get a higher level, use the deep neural network for relationship extraction and use for graph construction It works. The author also proposed that in the face of some implicit relationships, you can consider using relational reasoning for relation completion.

3.3 Quantitative analysis

3.3.1 Experiment Aim

The purpose of the quantitative analysis experiment is to verify the feasibility of the essay logic scoring workflow proposed in the qualitative experiment and discuss the appropriate scoring standards.

The experiment will hope to obtain the following data:

- Knowledge graphs corresponding to essays completed by writers of different writing levels.
- Each index of each knowledge graph, such as the largest connected subgraph, etc.

3.3.2 Experiment Design

Quantitative analysis experiments are mainly based on the results obtained on different essays by comparing relationship extraction and knowledge graph construction. Then compare these results to determine the appropriate scoring criteria.

Data

The data is divided into 3 groups, 5 essays in each group, taken from the IELTS Essay Samples on ielts-blog.com. The three groups are composed of 5 points, 7 points and 9 points respectively. The data uses CoreNLP for named entity recognition and split into single sentences.

Method

The quantitative analysis experiment uses the BERT model used in the previous qualitative analysis experiment. The data has been pre-made named entity recognition, and manually marked and corrected. Part of the program code is in Appendix B. After the relationship extraction is completed, all the extracted relationships are put into the set R separately. And all entities s , o belong to $r = \{s, r, o\}$, and they are included in the set E for coreference resolution. The purpose of coreference resolution is to merge identical or similar entities, as well as to merge referents and pronouns. This process uses CoreNLP's dcoref module, based on the system scheme proposed by Lee et al. (2011), which is a collection of deterministic coreference

resolution models that incorporate lexical, syntactic, semantic, and discourse information. Finally, we apply the relation set R to the entity set E that has passed the coordinate resolution to obtain the knowledge graph.

Evaluate

In order to evaluate the scoring model of the essay, we first need to determine the indicators to be compared. Based on the conclusion of qualitative analysis, the number of statistical relationships and the size of the largest connected subgraph are selected as the control indicators. Compare the results of the three sets of data to observe the relationship between the numerical distribution and the IELTS score.

3.3.3 Experiment Procedure

Experiment preparation

As with the qualitative analysis experiment, Stanford Core NLP is installed on the experimental machine. Because of the need to process text data in batches, python is used to complete the processing. After removing the text data without standard symbols, perform NER operations in batches, and mark the marked results as [E1] and [E2]. In order to improve the accuracy rate and eliminate the interference of the system error on the theory proposed in this paper, 15 test samples were manually cleared of incorrect entity labels and missing entities were added.

Triple extraction

First train the relation extraction model, using ALBERT as the pre-training language model and FewRel Dataset as the training data for training. The training results are shown in Figure 3.10. Then the labeled sentences are sent to the BERT-based relationship extraction model for relationship prediction and judgment. After obtaining a meaningful relationship prediction, put the relationship into the set R .

3.3.4 Experiment Result

The triple extraction experimental data results are shown in Table 3.1. It includes the total number of relations extracted from different essays at different levels and the maximum connected graph size.

Score	IELTS W:5		IELTS W:7		IELTS W:9	
	Before	After	Before	After	Before	After
No.1	171	21	92	19	44	18
No.2	39	13	73	22	68	26
No.3	39	9	113	21	92	21
No.4	30	11	70	17	51	17
No.5	112	18	49	19	103	22

Table 3.1: The relationship number of different essays obtained after Triple extraction, including the duplicate relationship before and after the merger.

Obviously, it can be seen that the complete triple logic proposed by the 5-point writer is relatively small, and the simple sentence can also be seen by observing the text. Since the length of IELTS essays is mostly around 250 words, the triples provided by a single sentence are limited, and the number of relations that can be extracted by 7-point and 9-point essays is relatively close.

The final knowledge graph is processed by the Torjan algorithm to get the maximum number of edges in the subgraph. It can also be seen that it is difficult to extract effective logical edges in a 5-point composition. However, the difference between 7 points and 9 points is not significant. After the author’s inspection, it is found that part of the 9 points essay is not recognized. IELTS essays encourage the use of compound sentences, and the reasoning relationships of high-scoring essays are often not fully extracted. Of course, this system cannot evaluate the pros and cons of logic, nor can it understand metaphors. The gap between a full score of IELTS and a score of 7 is often more difficult to be recognized by the machine, such as the level of words used, the rationality of opinions, and the coherence of expression.

Score	IELTS W:5	IELTS W:7	IELTS W:9
No.1	11	14	16
No.2	5	20	25
No.3	6	13	17
No.4	8	15	12
No.5	13	19	16
Avg.	8.6	16.2	17.2

Table 3.2: The number of sides of the constructed knowledge graph after co-reference resolution.

Chapter Four

Conclusion

4.1 Achievements

This article has completed the following work:

- Discussed and compared a number of feasible methods to extract the relationship triple from the essay. Including the dependency syntax extraction based on Stanford CoreNLP, and the extraction method using the BERT network model.
- Researched and compared various possible scoring standards based on knowledge graph and relationship triples. And 15 essays were used as experimental subjects for analysis. And discussed that the low-segment essay does lack the connection of logical entities.
- It is found that the relationship extraction and coreference resolution of each step at this stage still have a high error rate in natural language texts. It is also analyzed that even though the entire automated scoring workflow is feasible, the performance of the tool chain is still not satisfactory.
- Researched how to merge the repetitive logical relationship, mainly using the coreference resolution algorithm.

4.2 Critical analysis

4.2.1 Effectiveness of the scoring method

One question worth discussing is whether the scoring method proposed by the author is effective. In the research of this article, the author also discovered the relevance of several numerical features. First, the number of relationships that can be extracted is basically proportional to the length of the essay; Second, complex expressions may make it impossible to extract the correct relationship, but complex expressions often mean a higher level of language use; Third, the disconnection of the logical relationship caused by a small grammatical error may cause the subgraph to be incorrectly segmented, in other words, the robustness is insufficient. But in general, the extracted knowledge graph, or semantic network, does express most of the logical relationships of the full text.

4.2.2 The feasibility of a pure end-to-end approach

With the development of statistical methods and neural network technology, the use of end-to-end methods to replace traditional workflow methods has been widely praised. The end-to-end method performs well in avoiding accumulation of errors, reducing system complexity, and improving task accuracy. But we can still find several problems with the end-to-end approach. The first is to rely on a huge amount of learning data. Google's BERT model has $12 * 768 * 12 = 110M$ parameters, while BERT-Large has 340M parameters. The author of BERT trained a language model on a corpus of 3.3 billion texts. Such a training scale is obviously not achievable by researchers in small institutions. At the same time, various models that combine grammatical features at this stage will always perform better on pure end-to-end models, such as the use of dependency syntax trees for relationship extraction in background chapters. Relational triple extraction is the most difficult part of natural language understanding. There are still few triple extraction algorithms that can achieve a recall rate of more than 90% on natural language texts, so there is no conclusion that end-to-end method or step-by-step method is better.

4.2.3 Unreliable NLP technology

Almost all NLP processing techniques do not perform as well in the experimental environment of the thesis when encountering natural language texts. For the larger-scale KBP37 data, the BERT-based model used in

the previous chapter only has an F1 value of 68.3(Soares et al., 2019). The F1 value of the Stanford OIE system on the KBP data set is only 28.3(Angeli, Premkumar, and C. D. Manning, 2015). These values mean that it is difficult for us to get accurate relationship predictions. In terms of coreference resolution, both Stanford CoreNLP and Spacy only obtained F1 values of 59.52 and 65.73 on the data of CoNLL 2012(Clark and C. D. Manning, 2016; Recasens, Marneffe, and Potts, 2013). These performances mean that NLP systems have huge room for improvement in language understanding. Because of the high error rate of these systems, the author needs to manually correct the results after each step is completed.

4.3 Problems analysis

The author encountered the following problems in the research process:

4.3.1 Error accumulation problem

The author found in the research that when we use more intermediate steps, the errors produced by each step will add up. For example, if we miss an entity in entity recognition, all subsequent relationships related to this entity will not be recognized. This problem is more serious in NLP, which requires more steps to complete the experimental goal. Even the most end-to-end algorithm requires basic tagging or embedding. We assume that the error rate of each step i is Err_i , and the overall error rate can be counted as:

$$Err_1 * Err_2 * Err_3 \cdots Err_n = Err_{total} \quad (4.1)$$

So when we evaluate the relationship extraction algorithm, we should directly calculate the accuracy, recall and F1 of the algorithm on natural language text. On the other hand, when an automatic scoring system involving educational fairness and ethics is launched, it must go through a large number of natural language text tests and choose the method with the highest accuracy to avoid the fairness impact on candidates.

4.3.2 Entity extraction problem

Entity recognition is an important step that is very error-prone and also affects subsequent operations. There are often two reasons for its errors, one is the inaccuracy of POS-Tagging, and the other is the error caused by the algorithm itself. There are often two reasons for its errors, one is the inaccuracy of POS-Tagging,

and the other is the error caused by the algorithm itself. The first problem comes from the ambiguity of words, and the same word may serve as different grammatical components. The second problem comes from the characteristics of many natural languages, including difficult to determine entity boundaries, compound relationships, and adjectives. Because the entity recognition designed in relation extraction is not only named entities, but also pronoun entities, it is very difficult to accurately extract them, and it also brings many subsequent errors due to nested entity recognition or errors.

4.3.3 Relationship duplication problem

Whether it is the Stanford OIE system or the BERT model, a large number of repetitive relationships have been extracted. These repetitive relationships can be divided into two categories, one is caused by nested entities or relationship keywords, as shown in Figure 4.1; the other is due to the active and passive repetition of the relationship, as shown in Figure 4.2.

```
- { 'subject': 'parent', 'relation': 'have', 'object': 'influence over children' }
- { 'subject': 'parent', 'relation': 'have', 'object': 'great influence' }
- { 'subject': 'parent', 'relation': 'have', 'object': 'influence' }
```

Figure 4.1: Nested entities from "great influence over children", as "influence over children", "great influence" and "influence"

```
Sentence: [E2]After eating the chicken[/E2] , [E1]he[/E1] developed a sore throat the next morning .
Predicted: Product-Producer(e2, e1)

Sentence: [E1]After eating the chicken[/E1] , [E2]he[/E2] developed a sore throat the next morning .
Predicted: Product-Producer(e1, e2)
```

Figure 4.2: Duplication caused by an inverted relationship

The problem of entity recognition has been mentioned in the previous section. For nested entities, the author proposes to merge the results of each sentence according to the predicted relationship keywords. The repetition caused by the active and passive needs to use the result of POS-Tagging. According to the use of the passive voice, choose one to keep.

4.3.4 Entity failed to merge

A typical type of entities that are difficult to merge are synonyms. For example, "skyscraper" and "high building" in Figure 3.9. Due to the diversity of words in natural language and the implicit relationships between different entities, it is a difficult step to merge similar or referent entities. The author here has two ideas to solve the entity merger in such cases. The first is to directly use the coreference resolution technique. Co-reference resolution can cluster similar entity words and pronouns, and merge entities based on the clustering results. The second is to use the cosine similarity of word vectors. Entity words or phrases with higher cosine similarity will be merged directly.

But there is another problem hidden in entities merging scenario, that is, how to find the underlying entity relationship. The expressions in some essays include analogies and other stylistic devices, and there may not be obvious textual expressions containing logical relationships between these entities. Therefore, a question worth exploring in logical scoring is whether it is necessary to reason about potential logical relationships, and whether such potential relationship reasoning will bring extra credits to low-scoring essays.

4.3.5 High-level writing is difficult to distinguish

In the quantitative analysis in Section 3.2, we found that the number of logical relationships and the maximum subgraph size of IELTS 7 and 9 essays are very close. Like Figure A.1, there is no obvious difference performance in last two thirds. According to the author's reference to IELTS Writing task 2 assessment criteria, 9 points requirement: uses cohesion in such a way that it attracts no attention. Therefore, we can see that after reaching a certain level of logical coherence, users of higher levels should use logical coherence unobtrusively, or "not rigid". For the existing text comprehension technology, how to distinguish between blunt and coherent text is still a difficult subject. The author here only assumes that high-level writing can be embodied through the diversity of logical expressions. The relevant knowledge of pedagogy and literature is beyond the author's knowledge.

4.4 Future works

4.4.1 Improve the accuracy of each step

For the logical scoring method proposed by the author in this article, the most effective optimization is to improve the accuracy of each step, especially the accuracy of information extraction and coreference resolution. The former can extract relationships correctly, and the latter can merge entities correctly.

4.4.2 Predict possible relationships

The author believes that a certain degree of relational reasoning is necessary for entities that are difficult to merge in some essays and the implicit logical connections. However, the adoption threshold of inference prediction must be set higher to avoid low-scoring essays from gaining undue advantages.

4.4.3 Better scoring standards

The existing scoring standards based on the number of relationships and the characteristics of the directed graph forest proposed by the author do not reflect the richer characteristics of logical relationships, but only consider the relationship between quantity and scale. How to evaluate the quality of logic is where the scoring algorithm can be optimized.

Appendix One

Result Examples

A.1 Quantitative analysis result charts

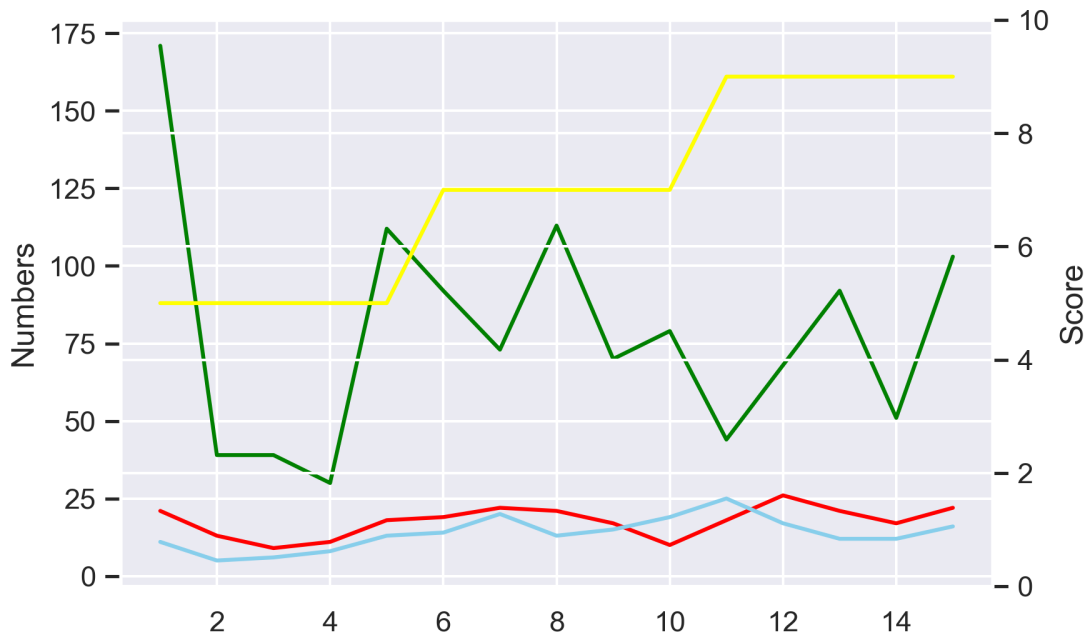


Figure A.1: Data result compare, green is relations before merge, red is after mergem, blue is scale of largest sub-graph, yellow is IELTS writing score

A.2 Program code and user guide

The whole workflow can be divided in to further steps:

- Download prepare dependency package and pre-train model code. Download Spacy, BERT large and Stanford CoreNLP.
- Run relation extraction infer steps using `main_task.py` in “bert-ie” folder. A replacement version by OpenIE in the file `corenlp.py`.
- Manually merge similar and duplicate relation
- Using code `graphbuild.py` to build Knowledge Graph, and calculate maxim sub-graph scale.
- An implements of Co-reference resolution merge code can be found in `coref.py` by Spacy and `neuarl-coref`(need to download).
- The Stanford Core NLP used for qualitative analysis can be found at <https://stanfordnlp.github.io/CoreNLP/usage.html>. The prerequisites of running this system include Java 8.

The bert-ie program file write by Soares et al., 2019, and running guide website on (<https://github.com/plkmo/BERT-Relation-Extraction>). Other tools and program write by author. The program code is submitted to git@git-teaching.cs.bham.ac.uk:mod-msc-proj-2019/yxs987.git. Running code by reading README.md. To train the model of BERT, I suggest an 8 core high performance CPU with more than 16GB memory.

References

- Aho, Alfred V, Ravi Sethi, and Jeffrey D Ullman (1986). “Compilers, principles, techniques”. In: *Addison wesley* 7.8, p. 9.
- Allen, James (1995). *Natural language understanding*. Pearson.
- Angeli, Gabor, Melvin Jose Johnson Premkumar, and Christopher D Manning (2015). “Leveraging linguistic structure for open domain information extraction”. In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 344–354.
- Asahara, Masayuki and Yuji Matsumoto (2003). “Japanese named entity extraction with redundant morphological analysis”. In: *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 8–15.
- Attali, Yigal and Jill Burstein (2006). “Automated essay scoring with e-rater® V. 2”. In: *The Journal of Technology, Learning and Assessment* 4.3.
- Auer, Sören et al. (2007). “Dbpedia: A nucleus for a web of open data”. In: *The semantic web*. Springer, pp. 722–735.
- Banko, Michele et al. (2007). “Open information extraction from the web”. In: *IJCAI’07 Proceedings of the 20th international joint conference on Artificial intelligence*, pp. 2670–2676.
- Beeferman, Doug, Adam Berger, and John Lafferty (1999). “Statistical models for text segmentation”. In: *Machine learning* 34.1-3, pp. 177–210.
- Bergmanis, Toms and Sharon Goldwater (2018). “Context sensitive neural lemmatization with lematus”. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1391–1400.
- Berners-Lee, Tim (2006). “Linked data-design issues”. In: <http://www.w3.org/DesignIssues/LinkedData.html>.

-
- Berners-Lee, Tim, James Hendler, and Ora Lassila (2001). “The semantic web”. In: *Scientific american* 284.5, pp. 34–43.
- Bikel, Daniel M and Mitchell P Marcus (2004). *On the parameter space of generative lexicalized statistical parsing models*. Citeseer.
- Bikel, Daniel M, Scott Miller, et al. (1998). “Nymble: a high-performance learning name-finder”. In: *arXiv preprint cmp-lg/9803003*.
- Borthwick, Andrew et al. (1998). “Exploiting diverse knowledge sources via maximum entropy in named entity recognition”. In: *Sixth Workshop on Very Large Corpora*.
- Braverman, Simone (2005). URL: <https://www.ielts-blog.com/>.
- Bresnan, Joan, Ronald M Kaplan, et al. (1982). “Introduction: Grammars as mental representations of language”. In: *The mental representation of grammatical relations*, pp. xvii–lii.
- Burstein, Jill and Martin Chodorow (1999). “Automated essay scoring for nonnative English speakers”. In: *Computer mediated language assessment and evaluation in natural language processing*.
- Cai, Deng et al. (2017). “Fast and accurate neural word segmentation for Chinese”. In: *arXiv preprint arXiv:1704.07047*.
- Cambridge, UP (2009). “Introduction to information retrieval”. In:
- Charniak, Eugene and Mark Johnson (2005). “Coarse-to-fine n-best parsing and MaxEnt discriminative reranking”. In: *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL '05)*, pp. 173–180.
- Chen, Danqi and Christopher D Manning (2014). “A fast and accurate dependency parser using neural networks”. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 740–750.
- Chrupała, Grzegorz (2006). “Simple data-driven context-sensitive lemmatization”. In:
- Church, Kenneth Ward (1989). “A stochastic parts program and noun phrase parser for unrestricted text”. In: *International Conference on Acoustics, Speech, and Signal Processing, IEEE*, pp. 695–698.
- Clark, Kevin and Christopher D Manning (2016). “Deep reinforcement learning for mention-ranking coreference models”. In: *arXiv preprint arXiv:1609.08667*.
- Collins, Michael (1997). “Three generative, lexicalised models for statistical parsing”. In: *arXiv preprint cmp-lg/9706022*.
- (2002). “Ranking algorithms for named entity extraction: Boosting and the votedperceptron”. In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 489–496.

-
- Collins, Michael and Brian Roark (2004). “Incremental Parsing with the Perceptron Algorithm”. In: *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*. Barcelona, Spain, pp. 111–118. DOI: [10.3115/1218955.1218970](https://doi.org/10.3115/1218955.1218970). URL: <https://www.aclweb.org/anthology/P04-1015>.
- De Marneffe, Marie-Catherine et al. (2014). “Universal Stanford dependencies: A cross-linguistic typology.” In: *LREC*. Vol. 14, pp. 4585–4592.
- DeJong, Gerald (1982). “An overview of the FRUMP system”. In: *Strategies for natural language processing* 113, pp. 149–176.
- Del Corro, Luciano and Rainer Gemulla (2013). “Clausie: clause-based open information extraction”. In: *Proceedings of the 22nd international conference on World Wide Web*, pp. 355–366.
- Denis, Pascal and Jason Baldridge (2008). “Specialized models and ranking for coreference resolution”. In: *Proceedings of the 2008 conference on empirical methods in natural language processing*, pp. 660–669.
- DeRose, Steven J (1988). “Grammatical category disambiguation by statistical optimization”. In: *Computational linguistics* 14.1.
- Dettmers, Tim et al. (2018). “Convolutional 2d knowledge graph embeddings”. In: *Thirty-Second AAAI Conference on Artificial Intelligence*.
- Earley, Jay (1970). “An efficient context-free parsing algorithm”. In: *Communications of the ACM* 13.2, pp. 94–102.
- Esuli, Andrea and Fabrizio Sebastiani (2010). “Evaluating information extraction”. In: *International Conference of the Cross-Language Evaluation Forum for European Languages*. Springer, pp. 100–111.
- Fader, Anthony, Stephen Soderland, and Oren Etzioni (2011). “Identifying relations for open information extraction”. In: *Proceedings of the 2011 conference on empirical methods in natural language processing*, pp. 1535–1545.
- Foltz, Peter W, Darrell Laham, and Thomas K Landauer (1999). “The intelligent essay assessor: Applications to educational technology”. In: *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning* 1.2, pp. 939–944.
- Forgy, Charles L (1989). “Rete: A fast algorithm for the many pattern/many object pattern match problem”. In: *Readings in Artificial Intelligence and Databases*. Elsevier, pp. 547–559.
- Galárraga, Luis Antonio et al. (2013). “AMIE: association rule mining under incomplete evidence in ontological knowledge bases”. In: *Proceedings of the 22nd international conference on World Wide Web*, pp. 413–422.

-
- Gamallo, Pablo, Marcos Garcia, and Santiago Fernández-Lanza (2012). “Dependency-based open information extraction”. In: *Proceedings of the joint workshop on unsupervised and semi-supervised learning in NLP*, pp. 10–18.
- Garside, Roger (1987). “The CLAWS word-tagging system”. In: *The Computational analysis of English: A corpus-based approach*. London: Longman, pp. 30–41.
- Gazdar, Gerald et al. (1985). *Generalized phrase structure grammar*. Harvard University Press.
- Grishman, Ralph and Beth M Sundheim (1996). “Message understanding conference-6: A brief history”. In: *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*.
- Hays, David G (1964). “Dependency theory: A formalism and some observations”. In: *Language* 40.4, pp. 511–525.
- Hearst, Marti A (1992). “Automatic acquisition of hyponyms from large text corpora”. In: *Coling 1992 volume 2: The 15th international conference on computational linguistics*.
- Hobbs, Jerry R (1978). “Resolving pronoun references”. In: *Lingua* 44.4, pp. 311–338.
- Hovy, Eduard, Chin-Yew Lin, and Liang Zhou (2005). “A BE-based multi-document summarizer with sentence compression”. In: *Proceedings of Multilingual Summarization Evaluation*, p. 15.
- Huang, Zhiheng, Wei Xu, and Kai Yu (2015). “Bidirectional LSTM-CRF models for sequence tagging”. In: *arXiv preprint arXiv:1508.01991*.
- Hudson, Richard A (1984). *Word grammar*. Blackwell Oxford.
- Johansson, Stig et al. (1986). *The Tagged LOB corpus: users’ manual*. Norwegian Computing Centre for the Humanities.
- Joshi, Aravind K, Leon S Levy, and Masako Takahashi (1975). “Tree adjunct grammars”. In: *Journal of computer and system sciences* 10.1, pp. 136–163.
- Jurafsky, Dan (2000). *Speech & language processing*. Pearson Education India.
- (2012). “Learning Thesauruses and Knowledge Bases”. In: Presented as the lecture 6 in Natural Language Processing, California.
- Kameyama, Megumi (1986). “A property-sharing constraint in centering”. In: *24th Annual Meeting of the Association for Computational Linguistics*, pp. 200–206.
- Kasami, Tadao (1965). “An efficient recognition and syntax algorithm for context-free languages”. In: *Technical report, Air Force Cambridge Research Lab*.
- Kay, Martin (1984). “Functional unification grammar: A formalism for machine translation”. In: *10th International Conference on Computational Linguistics and 22nd Annual Meeting of the Association for Computational Linguistics*, pp. 75–78.

-
- Kay, Martin (1985). “Parsing in functional unification grammar”. In: *Natural language parsing*, pp. 251–278.
- Klein, Dan and Christopher D Manning (2003). “Accurate unlexicalized parsing”. In: *Proceedings of the 41st annual meeting of the association for computational linguistics*, pp. 423–430.
- Lafferty, John, Andrew McCallum, and Fernando CN Pereira (2001). “Conditional random fields: Probabilistic models for segmenting and labeling sequence data”. In:
- Lao, Ni and William W Cohen (2010). “Relational retrieval using a combination of path-constrained random walks”. In: *Machine learning* 81.1, pp. 53–67.
- Lee, Heeyoung et al. (2011). “Stanford’s multi-pass sieve coreference resolution system at the conll-2011 shared task”. In: *Proceedings of the 15th conference on computational natural language learning: Shared task*. Association for Computational Linguistics, pp. 28–34.
- Lenat, Douglas B, Mayank Prakash, and Mary Shepherd (1985). “CYC: Using common sense knowledge to overcome brittleness and knowledge acquisition bottlenecks”. In: *AI magazine* 6.4, pp. 65–65.
- Lin, Yankai et al. (2016). “Neural relation extraction with selective attention over instances”. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2124–2133.
- Lovins, Julie Beth (1968). “Development of a stemming algorithm”. In: *Mech. Transl. Comput. Linguistics* 11.1-2, pp. 22–31.
- Luo, Xiaoqiang et al. (2004). “A mention-synchronous coreference resolution algorithm based on the bell tree”. In: *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pp. 135–142.
- Malaviya, Chaitanya, Shijie Wu, and Ryan Cotterell (2019). “A simple joint model for improved contextual neural lemmatization”. In: *arXiv preprint arXiv:1904.02306*.
- Manning, Christopher D et al. (2014). “The Stanford CoreNLP natural language processing toolkit”. In: *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pp. 55–60.
- McCallum, Andrew, Dayne Freitag, and Fernando CN Pereira (2000). “Maximum Entropy Markov Models for Information Extraction and Segmentation.” In: *Icml*. Vol. 17. 2000, pp. 591–598.
- McCallum, Andrew and Wei Li (2003). “Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons”. In:
- Mel’cuk, Igor Aleksandrovic et al. (1988). *Dependency syntax: theory and practice*. SUNY press.
- Miller, George A (1998). *WordNet: An electronic lexical database*. MIT press.

- Mintz, Mike et al. (2009). “Distant supervision for relation extraction without labeled data”. In: *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pp. 1003–1011.
- Mitchell, Tom et al. (2018). “Never-ending learning”. In: *Communications of the ACM* 61.5, pp. 103–115.
- Mitkov, Ruslan (1998). “Robust Pronoun Resolution with Limited Knowledge.” In: pp. 869–875. DOI: [10.3115/980691.980712](https://doi.org/10.3115/980691.980712).
- Miwa, Makoto and Mohit Bansal (2016). “End-to-end relation extraction using lstms on sequences and tree structures”. In: *arXiv preprint arXiv:1601.00770*.
- Nakaiwa, Hiromi and Satoshi Shirai (1996). “Anaphora resolution of Japanese zero pronouns with deictic reference”. In: *COLING 1996 Volume 2: The 16th International Conference on Computational Linguistics*.
- Ng, Hwee Tou and Jin Kiat Low (2004). “Chinese part-of-speech tagging: One-at-a-time or all-at-once? word-based or character-based?” In: *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pp. 277–284.
- Nivre, Joakim (2004). “Incrementality in deterministic dependency parsing”. In: *Proceedings of the workshop on incremental parsing: Bringing engineering and cognition together*, pp. 50–57.
- (2005). “Dependency grammar and dependency parsing”. In: *MSI report* 5133.1959, pp. 1–32.
- Oepen, Stephan and John Carroll (2000). “Parser engineering and performance profiling”. In: *Natural Language Engineering* 6.1, pp. 81–97.
- Page, Ellis Batten (2003). “Project Essay Grade: PEG.” In:
- Paice, Chris D (1990). “Another stemmer”. In: *ACM Sigir Forum*. Vol. 24. 3. ACM New York, NY, USA, pp. 56–61.
- Pan, Jeff Z et al. (2017). *Exploiting linked data and knowledge graphs in large organisations*. Springer.
- Pantel, Patrick and Marco Pennacchiotti (2006). “Espresso: Leveraging generic patterns for automatically harvesting semantic relations”. In: *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pp. 113–120.
- Paulheim, Heiko (2017). “Knowledge graph refinement: A survey of approaches and evaluation methods”. In: *Semantic web* 8.3, pp. 489–508.
- Peng, Fuchun et al. (2007). “Context sensitive stemming for web search”. In: *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 639–646.
- Peters, Matthew E et al. (2018). “Deep contextualized word representations”. In: *arXiv preprint arXiv:1802.05365*.

-
- Petrov, Slav and Dan Klein (2007). “Improved inference for unlexicalized parsing”. In: *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pp. 404–411.
- Plank, Barbara, Anders Søgaard, and Yoav Goldberg (2016). “Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss”. In: *arXiv preprint arXiv:1604.05529*.
- Pollard, Carl and Ivan A Sag (1994). *Head-driven phrase structure grammar*. University of Chicago Press.
- Porter, Martin F et al. (1980). “An algorithm for suffix stripping.” In: *Program* 14.3, pp. 130–137.
- Porter, Martin F (2001). *Snowball: A language for stemming algorithms*.
- Rau, Lisa F (1987). “Knowledge organization and access in a conceptual information system”. In: *Information Processing & Management* 23.4, pp. 269–283.
- (1991). “Extracting company names from text”. In: *Proceedings The Seventh IEEE Conference on Artificial Intelligence Application*. IEEE Computer Society, pp. 29–30.
- Recasens, Marta, Marie-Catherine de Marneffe, and Christopher Potts (2013). “The life and death of discourse entities: Identifying singleton mentions”. In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 627–633.
- Riloff, Ellen (1996). “Automatically generating extraction patterns from untagged text”. In: *Proceedings of the national conference on artificial intelligence*, pp. 1044–1049.
- Riloff, Ellen and Wendy Lehnert (1994). “Information extraction as a basis for high-precision text classification”. In: *ACM Transactions on Information Systems (TOIS)* 12.3, pp. 296–333.
- Robinson, Jane J (1970). “Dependency structures and transformational rules”. In: *Language*, pp. 259–285.
- Rosenkrantz, Daniel J and Philip M Lewis (1970). “Deterministic left corner parsing”. In: *11th Annual Symposium on Switching and Automata Theory (swat 1970)*. IEEE, pp. 139–152.
- Schneider, Edward W (1973). “Course Modularization Applied: The Interface System and Its Implications For Sequence Control and Data Analysis.” In:
- Seki, Kazuhiro, Atsushi Fujii, and Tetsuya Ishikawa (2002). “A probabilistic method for analyzing Japanese anaphora integrating zero pronoun detection and resolution”. In: *arXiv preprint cs/0206030*.
- Selman, Bart, Hector Levesque, and David Mitchell (1992). “A New Method for Solving Hard Satisfiability Problems”. In:
- Selman, Bart, Henry A Kautz, Bram Cohen, et al. (1993). “Local search strategies for satisfiability testing.” In: *Cliques, coloring, and satisfiability* 26, pp. 521–532.

-
- Shermis, Mark D and Jill C Burstein (2003). *Automated essay scoring: A cross-disciplinary perspective*.
Routledge.
- Soares, Livio Baldini et al. (2019). “Matching the blanks: Distributional similarity for relation learning”. In:
arXiv preprint arXiv:1906.03158.
- Socher, Richard et al. (2013). “Parsing with compositional vector grammars”. In: *Proceedings of the 51st
Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 455–
465.
- Stanovsky, Gabriel et al. (2016). “Getting more out of syntax with props”. In: *arXiv preprint arXiv:1603.01648*.
- Strubell, Emma et al. (2017). “Fast and accurate entity recognition with iterated dilated convolutions”. In:
arXiv preprint arXiv:1702.02098.
- Suchanek, Fabian M, Gjergji Kasneci, and Gerhard Weikum (2007). “Yago: a core of semantic knowledge”.
In: *Proceedings of the 16th international conference on World Wide Web*, pp. 697–706.
- Sun, Xu et al. (2009). “A discriminative latent variable chinese segmenter with hybrid word/character infor-
mation”. In: *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North
American Chapter of the Association for Computational Linguistics*, pp. 56–64.
- Tesnière, Lucien (1959). “Éléments de syntaxe structurale”. In:
- Tomita, Masaru (1985). “An Efficient Context-Free Parsing Algorithm for Natural Languages.” In: *IJCAI*.
Vol. 2. Citeseer, pp. 756–764.
- Toutanova, Kristina, Aria Haghighi, and Christopher D Manning (2005). “Joint learning improves seman-
tic role labeling”. In: *Proceedings of the 43rd Annual Meeting of the Association for Computational
Linguistics (ACL’05)*, pp. 589–596.
- Toutanova, Kristina, Dan Klein, et al. (2003). “Feature-rich part-of-speech tagging with a cyclic dependency
network”. In: *Proceedings of the 2003 conference of the North American chapter of the association
for computational linguistics on human language technology-volume 1*. Association for Computational
Linguistics, pp. 173–180.
- Toutanova, Kristina and Christopher Manning (2000). “Enriching the knowledge sources used in a maximum
entropy part-of-speech tagger”. In: *Proceedings of the 2000 Joint SIGDAT Conference EMNLP/VLC,
63-71, 2000*.
- Turney, Peter D (2005). “Measuring semantic similarity by latent relational analysis”. In: *arXiv preprint
cs/0508053*.

-
- White, Aaron Steven et al. (2016). “Universal decompositional semantics on universal dependencies”. In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1713–1723.
- Winograd, Terry (1972). “Understanding natural language”. In: *Cognitive psychology* 3.1, pp. 1–191.
- WIRÉN, MATS (2000). “CHRISTER SAMUELSSON”. In: *Handbook of Natural Language Processing*, p. 59.
- Wu, Fei and Daniel S Weld (2010). “Open information extraction using Wikipedia”. In: *Proceedings of the 48th annual meeting of the association for computational linguistics*, pp. 118–127.
- Xue, Nianwen (2003). “Chinese word segmentation as character tagging”. In: *International Journal of Computational Linguistics & Chinese Language Processing, Volume 8, Number 1, February 2003: Special Issue on Word Formation and Chinese Language Processing*, pp. 29–48.
- Yang, Xiaofeng et al. (2008). “An entity-mention model for coreference resolution with inductive logic programming”. In: *Proceedings of Acl-08: Hlt*, pp. 843–851.
- Yang, Yunlun et al. (2016). “A position encoding convolutional neural network based on dependency tree for relation classification”. In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 65–74.
- Yin, Qingyu et al. (2018). “Deep reinforcement learning for chinese zero pronoun resolution”. In: *arXiv preprint arXiv:1806.03711*.
- Younger, Daniel H (1967). “Recognition and parsing of context-free languages in time n^3 ”. In: *Information and control* 10.2, pp. 189–208.
- Yu, Fisher and Vladlen Koltun (2015). “Multi-scale context aggregation by dilated convolutions”. In: *arXiv preprint arXiv:1511.07122*.
- Zeng, Daojian, Kang Liu, Yubo Chen, et al. (2015). “Distant supervision for relation extraction via piecewise convolutional neural networks”. In: *Proceedings of the 2015 conference on empirical methods in natural language processing*, pp. 1753–1762.
- Zeng, Daojian, Kang Liu, Siwei Lai, et al. (2014). “Relation classification via convolutional deep neural network”. In: *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pp. 2335–2344.
- Zhang, Longkai et al. (2013). “Exploring representations from unlabeled data with co-training for Chinese word segmentation”. In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 311–321.

- Zhang, Meishan, Yue Zhang, and Guohong Fu (2016). “Transition-based neural word segmentation”. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 421–431.
- Zhao, Shanheng and Hwee Tou Ng (2007). “Identification and resolution of Chinese zero pronouns: A machine learning approach”. In: *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pp. 541–550.
- Zhu, Muhua et al. (2013). “Fast and accurate shift-reduce constituent parsing”. In: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 434–443.
- Zwicky, Arnold M (1985). “Heads”. In: *Journal of linguistics* 21.1, pp. 1–29.

Websites consulted

- Wikipedia – www.wikipedia.org