

Master Thesis

# Improving Text Representation Learning by Modeling Heterogeneous Information

Department of System Engineering  
Graduate School of Informatics  
Shizuoka Institute of Science and Technology  
Student Number 2121026 Xiaoran Li  
Supervisor: Prof. Toshiaki Takano

January 17, 2023

# Improving Text Representation Learning by Modeling Heterogeneous Information

Xiaoran Li

## Abstract

“*The success of machine learning algorithms generally depends on data representation*”  
– Yoshua Bengio<sup>1</sup>

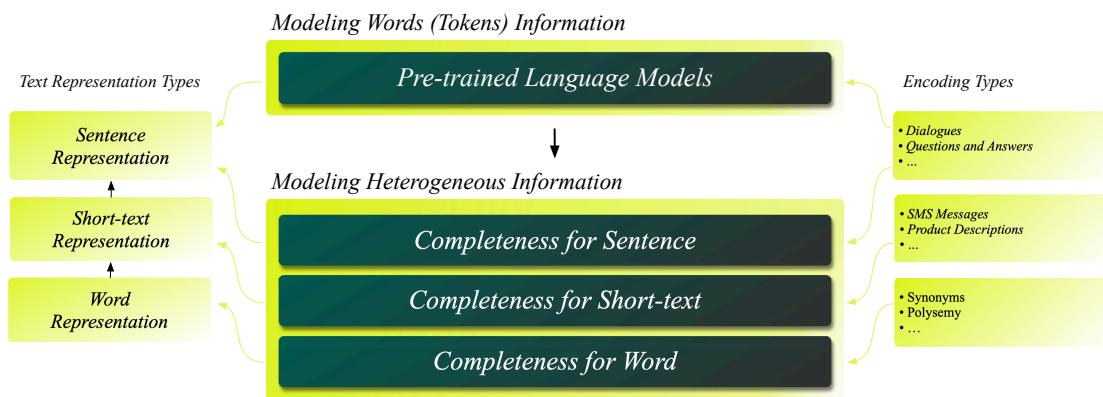


Figure 1: Text Representation Learning with Heterogeneous Information

Different representations for different text data types (e.g., word, short-text, sentence, etc.) may result in the lack or exposure of helpful information, which determines whether the algorithm can solve the problem effectively. Although language-agnostic sequence-to-sequence pre-training methods lead to nontrivial improvements in text representation learning, such methods still require the support of numerous monolingual corpora, and the model is also enormous. Previous work showed that text embedding performance could be effectively enhanced by injecting domain-specific prior knowledge. However, domain-specific prior knowledge is time-consuming and laborious to acquire. Thus, we suppose that injecting semantic-level general language knowledge in the training phase of

<sup>1</sup>Bengio received the 2018 ACM A.M. Turing Award.

the model can likewise effectively improve the expressiveness and reduce the model parameters. Moreover, we experimented with multi-type linguistic knowledge (this knowledge is usually heterogeneous information) to model learning text representations, and the performance was further improved, as shown in Figure 1. In this work, we explore the representation of three text structures:

### I: Word Representation

Most popular word representation methods are often accompanied by the training of large language models (i.e., Pre-trained Language Models) today. Since it can encode the current meaning of words according to their context, e.g., Bidirectional Encoder Representations from Transformers (BERT). This work explores the potential connection between BERT-based pre-trained language models and sememe and provides new reflections for constructing cross-linguistic sememe knowledge. More further, One of the causes why sememe has yet to be widely employed is that it does not satisfy the requirements of specific fields. Thus, we propose a sememe-oriented data augmentation method, which can effectively make sememe cover specific fields.

### II: Short-text Representation

Compared to long texts, due to limited length, short texts lack context information and strict syntactic structure, which are necessary for text understanding. In recent years, the marvelous success of Short-text Representation has depended on utilizing graph neural networks to fuse large proportions of heterogeneous information. However, with the enhancement of text specialization, the scale of knowledge nodes will furthermore become immense. This work takes advantage of the properties of sememe to significantly reduce the scale of knowledge nodes and maintain the expressiveness of the original model.

### III: Sentence Representation

Encoding sequences using Graph Neural Networks can effectively fuse knowledge (e.g., Short Text Classification), though this approach completely abandons the se-

quence order. This work utilizes a combination of sequence-to-sequence encoding and heterogeneous information in fusing sequence information with external knowledge to improve the expressiveness of the original model effectively. To this end, we designed a Paper Recommendation System based on this model, namely MIYU (ミユ). Furthermore, we use state-of-the-art unsupervised multi-hop question generation methods to construct the training dataset, which effectively helps students to retrieve relevant papers utilizing fewer terminologies.

**Keywords:** Representation Learning, Sememe Knowledge Base, Knowledge Extraction, Graph Neural Network, Deep Clustering Network, Word Embedding, Pretrained Language Model, Natural Language Processing, Deep Neural Network

# Table of contents

<b>List of Figures</b>	<b>i</b>
<b>List of Tables</b>	<b>vi</b>
<b>List of Notations and Abbreviations</b>	<b>viii</b>
<b>1 Introduction</b>	<b>1</b>
<b>2 Word Representation</b>	<b>7</b>
2.1 Related Work . . . . .	7
2.1.1 Contextualized Representation . . . . .	9
2.1.2 Bidirectional Language Models . . . . .	9
2.1.3 Masked Language Models . . . . .	10
2.2 The Completeness of Word Representations . . . . .	11
2.3 Building Sememe Knowledge Base by Deep Clustering Network . . . . .	12
2.3.1 Deep Sememe Clustering . . . . .	13
2.3.2 Cross-lingual Sememe . . . . .	16
2.3.3 Experiments . . . . .	17
2.4 Conclusion . . . . .	22
<b>3 Short Text Representation</b>	<b>23</b>
3.1 Related Work . . . . .	25

3.1.1	Graph Attention networks . . . . .	25
3.1.2	Hierarchical Heterogeneous Graph Representation Learning . . . . .	27
3.2	The Completeness of Short Text Representations . . . . .	28
3.3	Data Augmentation of Sememe Knowledge Base . . . . .	29
3.3.1	Building Sememe Knowledge Base . . . . .	30
3.3.2	Evaluations . . . . .	34
3.4	Modeling Heterogeneous Graph Neural Networks with Sememe Knowledge	37
3.4.1	Construct An Entity Knowledge Base . . . . .	39
3.4.2	Model Construction . . . . .	40
3.4.3	Experiments . . . . .	42
3.5	Conclusion . . . . .	43
<b>4</b>	<b>Sentence Representation</b>	<b>46</b>
4.1	Related Work . . . . .	47
4.1.1	Bidirectional Long Short-Term Memory Networks . . . . .	47
4.1.2	Bidirectional Encoder Representations from Transformers . . . . .	48
4.2	The Completeness of Sentence Representations . . . . .	51
4.3	NLP Paper Recommender: MIYU . . . . .	52
4.3.1	Finding NLP Papers by Asking a Multi-hop Question . . . . .	53
4.3.2	Improving Finding NLP Papers via Heterogenous Information . . . . .	63
4.3.3	Experiments . . . . .	66
4.4	Conclusion . . . . .	70
<b>5</b>	<b>Conclusion</b>	<b>71</b>
<b>Acknowledgments</b>		<b>73</b>
<b>Published Papers</b>		<b>85</b>

<b>A Appendices</b>	<b>88</b>
A.1 Sememes Cluster Prediction in English . . . . .	88
A.2 Sememe comparison on the Ohsumed dataset . . . . .	88

# List of Figures

1	Text Representation Learning with Heterogeneous Information . . . . .	a
1.1	Representation learning lies at the heart of the empirical success of deep learning for dealing with the curse of dimensionality. . . . .	1
1.2	Three ways of learning/storing knowledge. The right subfigure (c) is a new way proposed in this thesis. We argue that decomposing heterogeneous information is the key to learning representations. . . . .	2
1.3	Word-based semantic units (e.g. “ <i>apple</i> ”). It has very many senses. However, No matter how a word changes its sense, it is always composed of several primary and single sememes.) . . . . .	4
1.4	Details of Heterogeneous Knowledge . . . . .	5
2.1	Catrgory of Word Representation . . . . .	7
2.2	Word Representation for Non Language Models . . . . .	8
2.3	NNLM vs. RNN-based . . . . .	9
2.4	Embedding from language models . . . . .	10
2.5	Bidirectional Encoder Representations from Transformers . . . . .	10
2.6	Modeling Heterogeneous Knowledge Learning Word Representations . .	11
2.7	Construction of sememes based on DCN . . . . .	14
2.8	Restore data dimensions . . . . .	15

2.9	After aligning the word embedding space, only using monolingual word embedding for sememe clustering can be applied to other languages. . . . .	17
2.10	The illustration shows the distribution of the number of sentences. The X axis is the index of the word, and the Y axis is the number of sentences corresponding to the word. On the right is the distribution of the number of sentences after setting the upper and lower bounds. . . . . . . . . . .	18
2.11	Reconstruction error of the autoencoder during the pre-training stage. (We set a batch size of 64 sentences and observed that it quickly converges and stabilizes within 1,000 batches) . . . . . . . . . . .	21
3.1	Multilayer Graph Neural Networks . . . . . . . . . . .	23
3.2	Heterogeneous vs Homogeneous Graph . . . . . . . . . . .	24
3.3	Types of Graph Filters . . . . . . . . . . .	25
3.4	Heterogeneous Information Network for Short Texts . . . . . . .	27
3.5	Replacing Entities with Sememes to Learn Representations . . . . . .	28
3.6	If “widow” is a complex word and “husband” is not in sememes, we can use the definition instead of “husband” . . . . . . . . . . .	30
3.7	SKB Expansion Flow Chart: We use WordNet as the original dictionary and lexical expansion through Wikipedia, sharing a sememe set (As $CDV_{ori}$ ) between them. From the right side of the illustration, we used the look-up table method to replaced the <i>Definition</i> of $Dict_{oth-DA}$ with the <i>Sememe</i> of $SKB_{ori}$ , which is a straightforward operation. . . . . . . . . . .	31
3.8	Sememe based bipartite graph: We only learn the embedding representation of words by sememe, which means there is no line between words; we use TF-IDF for the weights of edges. The figure shows that “wife” and “husband” contain the same sememe, which should have approximate embedding representations. Note that here we do not consider the embedding representation of the sememe, so we use index instead of the word of the sememe itself. . . . . . . . . . .	34

3.9	Using Sememe Knowledge Bases to construct entity networks. We have added some sense nodes in the illustration on the right side. Before connecting the sememe nodes should first determine the correct sense nodes via sense discriminator. Note, the sense nodes will be discarded after helping us to learn the weights of the edges between the entity nodes and the sememe nodes. . . . .	38
3.10	Construction of sememe-based edges. We utilize the sense discriminator to select the correct sense. “Apple” is an entity in both sentences, the left side indicates the band, and the right side indicates the company; however, it is not well differentiated for TagMe. Note that the dotted line indicates that this sense will be discarded. . . . .	41
3.11	Analyze the effect of the average sememe number for senses on the model.	
	43	
4.1	Learning Hierarchical Representations . . . . .	46
4.2	Encoding Types of RNNs or LSTMs. Each rectangle is a vector and arrows represent functions (e.g. matrix multiply) . . . . .	47
4.3	Encoding Types of BERTs. The subillustrations (a), (b), (c) and (d) represent Sentence Pair Classification Tasks, Single Sentence Classification Tasks, Question Answer Tasks, and Single-sentence Tagging Tasks, respectively. Note, the final hidden vector of the special [CLS] token as $C$ , and the final hidden vector for the $i^{th}$ input token as $T(i)$ . . . . .	49
4.4	Sentence Representations Learning with Heterogeneous Information . . .	51
4.5	NLP Paper Recommender: MIYU . . . . .	53
4.6	Automatic generation of terminology dictionary-based questions from unstructured text. . . . .	54

4.7	The pipeline of terminology dictionary construction: In Terminology Extraction and Question Generation, The <b>blue font</b> represents terminologies extracted by the Q&A generation, the <b>red font</b> represents the keyphrase extraction results, and the <b>green font</b> represents the shared terminologies. Finally, the questions are converted into declarative sentences using straightforward word replacement and fill mask. . . . .	57
4.8	One-hop and multi-hop questions building: Entity Declaration . . . . .	58
4.9	One-hop and multi-hop questions building: Schema. Where rectangles represent entities and ellipses represent entity relationships. . . . .	59
4.10	One-hop and multi-hop questions building: Template Definition. We defined 17 possible questioning styles for the questioner based on the schema.	61
4.11	BiLSTM Mapping Flowchart: The whole network architecture consists of three parts: sentence representation learning, paper representation learning, and paper prediction. . . . .	62
4.12	Modeling Heterogeneous Information for Paper Recommendation Systems	64
4.13	The accuracy@1 performances by using the BiLSTM model on <i>Seen</i> test set. . . . .	69
4.14	The decreasing trend of the model's losses on <i>Seen</i> test set. . . . .	70
5.1	Toward Cross-lingual and Multi-round Conversational MIYU. . . . .	71
B1	The accuracy@10 performances by using the BiLSTM model on <i>Seen</i> test set. . . . .	91
B2	The accuracy@100 performances by using the BiLSTM model on <i>Seen</i> test set. . . . .	91
B3	The accuracy@10 performances by using the BiLSTM model on <i>Unseen</i> test set. . . . .	92
B4	The accuracy@100 performances by using the BiLSTM model on <i>Unseen</i> test set. . . . .	92

B5	The accuracy@1 performances by using the BiLSTM model on <i>Unseen</i> test set.	93
B6	The decreasing trend of the model's losses on <i>Unseen</i> test set.	93

# List of Tables

2.1	Sememes Cluster Prediction in English. More examples can be found in Appendix Table A1 . . . . .	22
3.1	Statistics of WordNetSKB, WikiSKB & WikiSKB-DA. WikiSKB-DA is a data augmented version of WikiSKB, and compared with HowNet and DictSKB. The gray font represents the previous SKB results (The same is true for the following tables). #AvgSem denotes the average Sememe number per sense, and “ + ” represents this average over four. . . . .	32
3.2	The results on Consistency Check of Sememe Annotations: MAP score of WordNetSKB exceeds the DictSKB, which we indicated in boldface. . . . .	35
3.3	The results on the Sememe graph: We merged WordNetSKB and WikiSKB. These tasks provide human scoring of the relationship between two words, thus assessing the degree of positive word relatedness. The method is to first calculate the cosine similarity of the two words and then compare them with the manual tags to calculate the Spearman correlation coefficient for scoring. . . . .	36
3.4	Sememe comparison on the Ohsumed dataset. Where “ <i>non</i> ” means no sememe of the word, we have merged WordNetSKB and WikiSKB+ as SKB-DA. More examples can be found in Appendix Table A2. . . . .	37

3.5	Statistics of SKB-Entity, and compared with HowNet and DictSKB. Note, #AvgSen denotes the average sense number per word, #AvgSem denotes the average sememe number per sense. . . . .	39
3.6	Test performance (%) measured on Snippets dataset. Our model is shown in <b>bold</b> , and it boosts the ACC and F1 score of the current best performing model (marked in underline) by 0.57 and 0.08 percentage points, respectively. . . . .	44
4.1	Buliding questions based on the entity definition for entity extraction. . .	60
4.2	Statistics about removing common terminology and terminology normalization. Note, #trem_wiki denotes the number of terms after removing common words in Wikipedia. #trem_norm denotes the number of terms after normalization. . . . .	67
4.3	Q&A dataset Statistics: Represent the number of Q&A, the number after merging, the number after de-duplication, and the number after splitting into training and test sets, respectively. . . . .	67
4.4	Examples of each question type. . . . .	67
4.5	The performance of the BiLSTM model on all test sets. Accuracy@1/10/100 denotes the accuracy that target papers appear in top 1/10/100 (higher better)	69
A1	Sememes cluster prediction in English . . . . .	88
A2	Sememe comparison on the Ohsumed dataset. . . . .	90

## List of Notations and Abbreviations

<b>Definition</b>	<b>Notation</b>
Vocabulary	$V$
Source language	$s$
Target language	$t$
Corpus of words used for training	$C$
The neural network parameters to learn	$\theta$
The $i^{th}$ word in corpus C	$w_i$
The embedding of word $w_i$	$\mathbf{w}_i$
The embedding vector's dimensionality	$d$
The $i^{th}$ in sentence in source(target) language	$sent_i^s (sent_i^t)$
Representation of sentence $sent_i^s (sent_i^t)$	$\mathbf{s}_i^s (\mathbf{s}_i^t)$
Representation of language model hidden layer of word $w_i$	$\mathbf{h}_i^{LM}$
A graph	$\mathcal{G}$
Edge set	$\mathcal{E}$
Node set	$\mathcal{V}$
A single node $v_i \in \mathcal{V}$	$v_i$
A single edge $e_{i,j}$ (connecting $v_i$ and $v_j$ )	$e_{i,j}$
Adjacent matrix of a graph	$A$
The embedding of vertex $v_i \in \mathcal{V}$	$\mathbf{h}_i$
The embedding of edge $e_{i,j} \in \mathcal{E}$	$\mathbf{e}_{i,j}$

<b>Term</b>	<b>Abbreviation</b>
Representation Learning	RL
Deep Neural Network	DNN
Graph neural network	GNN
Deep Clustering Network	DCN

**Table 0.2 continued from previous page**

<b>Term</b>	<b>Abbreviation</b>
Knowledge Base	KB
Knowledge Graph	KG
Sememe Knowledge Base	SKB
Short Text Classification	STC
Byte Pair Encoding	BPE
Sequence to Sequence	Seq2Seq
Matrix Factorization	MF
Long short-term memory	LSTM
Bidirectional Long short-term memory	BiLSTM
Language Model	LM
Bidirectional Language Model	BiLM
Large Language Model	LLM
Cross-lingual Language Model	XLM
Feedforward Neural Net Language Model	NNLM
Embeddings from Language Models	ELMo
Continuous Bag-of-Words Model	CBOW
Global Vectors for Word Representation	GloVe
Bidirectional Encoder Representations from Transformers	BERT
Graph Attention Network	GAT

# 1. Introduction

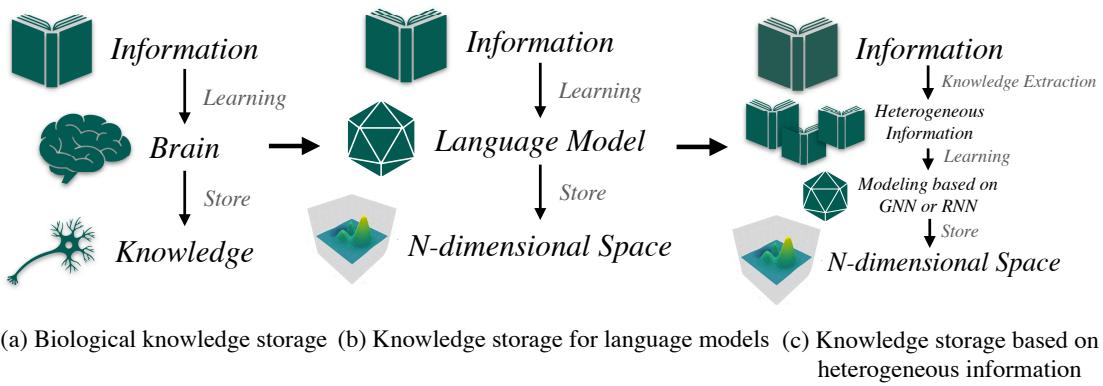
Representation Learning (RL), i.e., learning representations of the data, has considerable importance in machine learning and artificial intelligence (Figure 1.1). The performance of machine learning methods heavily depends on the choice of data representation on which they are applied. The rapidly developing field of representation learning concerns questions about how we can best learn meaningful and valuable data representations. “*The Sciences of the Artificial*” suggested that human information processing (including problem-solving, learning and discovering new knowledge) could be abstracted into simple mathematical models [1], as early as 1969. According to a simple information processing model, coupled with computers’ computing speed and massive storage space, artificial intelligence should be more powerful than humans. However, in any case, as we have seen, this did not occur. One important reason is that we have not yet been able to decipher how the human brain encodes and stores knowledge.



Figure 1.1: Representation learning lies at the heart of the empirical success of deep learning for dealing with the curse of dimensionality.

Proverbial, humans usually acquire knowledge by reading books (in this thesis, we do not consider audio-visual data, i.e., multi-modal data), despite the fact that books contain a large amount of plain text data (or symbolic data). Making the computer represent and store text data conveniently for calculation is the crux of triumph. This is also the core problem that Natural Language Processing (NLP) deals with. Simply put, no matter how long the text is, we can break it down into numerous token components. A token can be a word, a sentence, or even an article depending on the task. We can regard token  $X$  to the representation of token  $Y$  as an injective function, though it does not satisfy the surjective

condition, and the goal is to find a mapping function  $f_{mapping}$  such that  $f_{mapping} : X \rightarrow Y$  and can preserve the basic structure of  $X$ . When  $Y$  exists in a continuous space, it is called distributed representation learning [2] (the representation learning involved in this thesis is all based on distributed representation), and first developed in the context of statistical language modeling in neural net language models [3].



**Figure 1.2:** Three ways of learning/storing knowledge. The right subfigure (c) is a new way proposed in this thesis. We argue that decomposing heterogeneous information is the key to learning representations.

However, one of the significant drawbacks of deep learning-based representational learning approaches is that it requires a large amount of training data, which consumes many resources and leaves much cutting-edge technology in the hands of big companies with deep pockets, which is not what we want to see. We want like to find a simple way to improve this problem. For example, to lose as little semantic information as possible during information acquisition and training, retain as much heterogeneous text information as possible. In other words, utilize diverse textual information as training data. That is *Improving Text Representation Learning by Modeling Heterogeneous Information*. We argue that **information is diverse**, and humans can learn and store information fast since they can fully absorb diverse information. However, getting machines to learn diverse knowledge as well as humans is a challenging assignment. Historically, there have been

efforts to bridge such a gap to better use the information that exists in the world, as shown in Figure 1.2.

- (a) Humans are suitably educated from childhood, so they can easily understand the diverse information in life and record the knowledge in neurons.
- (b) Researchers leverage self-supervised learning [4] to learn the information from large corpora (i.e., Language Models) [5] and store knowledge in n-dimensional space.
- (c) Employ external knowledge or knowledge extraction to decompose diverse information into heterogeneous information, then perform representation learning through simple models and lightweight data.

This thesis focuses on utilizing heterogeneous information to encode text representations.

We mainly discussed the following issues:

1. What kind of knowledge do we need? It should be universal, cross-linguistic, and able to resolve semantic disagreements.
2. How to obtain such knowledge.

The first step in processing text information is encoding and projecting it into a particular space. However, text-based feature extraction is incredibly complex. For example, the vocabulary that exists in the world is enormous. Furthermore, sometimes a word means nearly the same as another word, and sometimes words are the same, but the meaning is different. We wanted to explore a simple way to solve these problems, and the Sememe Knowledge Base (SKB) seemed to be what we needed. Before the advent of the deep learning era, sememe 's low-manifold properties in high-dimensional space were one of the effective methods for interpreting and computing natural language. However, pure data-driven based on deep learning requires numerous computational resources, so we suppose that invoking a cross-linguistic sememe knowledge base as a priori knowledge data-driven is a good choice at this stage.

A sememe is a semantic language unit of meaning [6]; it is indivisible. However, people usually employ words as the minimum semantic unit because words as semantic

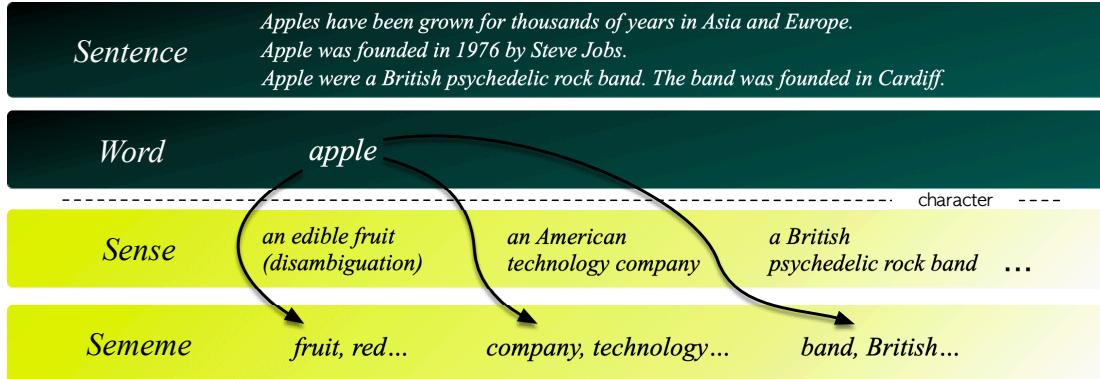


Figure 1.3: Word-based semantic units (e.g. “apple”). It has very many senses. However, No matter how a word changes its sense, it is always composed of several primary and single sememes.)

representations are available for writing, yet sememe is only a semantic concept. Usually, there is a sense semantic unit between the sememe and the word (as shown in Figure 1.3). linguists believe that all languages have the same limited sememe space [7] (e.g. HowNet SKB [8]). Moreover, cooperate with the multilingual encyclopedic dictionary as BabelNet [9] to build a multilingual SKB as [10]. Furthermore, Sememe can synthesize words and represent the essential meaning was successfully applied to Neural Networks [11] [12], Reverse Dictionaries [13] and Textual Adversarial Attacking [14], etc. We found that **sememes are a remarkable “Completeness” in Natural Language Understanding**, which can be easily embedded in text representations to compensate for the shortcomings of Language Models (like in Figure 1.4).

SKB has been successfully applied to many NLP tasks, and by learning the smallest unit of meaning, computers can more easily understand human language. However, Existing sememe KBs are built on only manual annotation, human annotations have personal understanding biases, the meaning of vocabulary will be constantly updated and changed with the times, and artificial methods are not always practical. To address the issue, we propose an unsupervised method based on a deep clustering network (DCN) [15] to build a sememe KB, and you can use any language to build a KB through this method. We

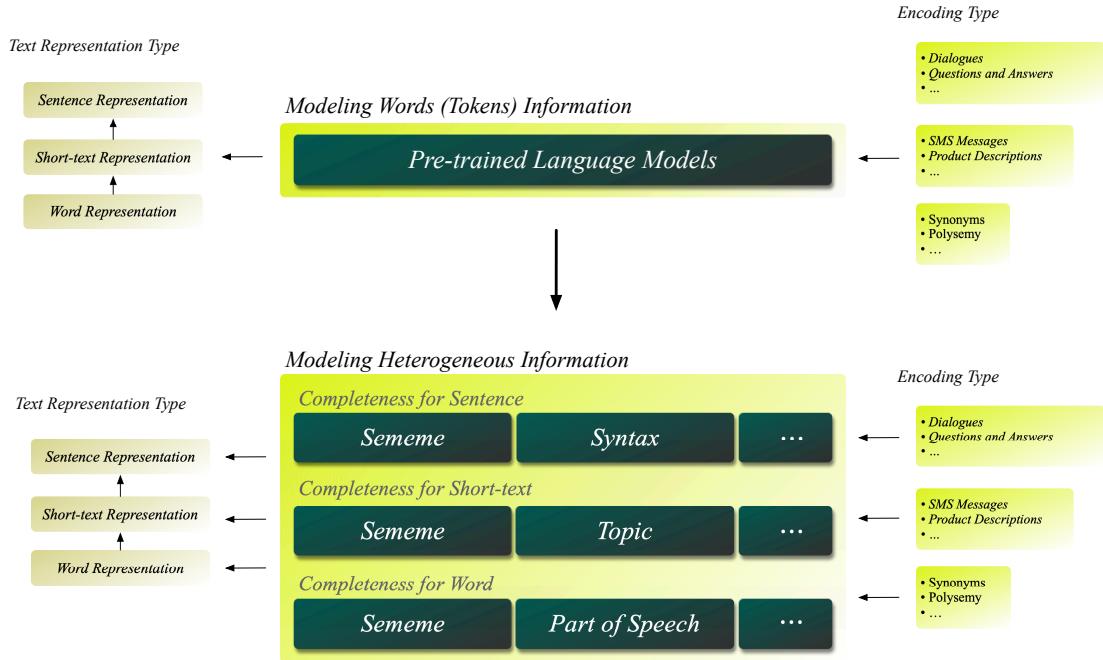


Figure 1.4: Details of Heterogeneous Knowledge

first learn the distributed representation of multilingual words, then align them in a single vector space, learn the multi-layer meaning of each word through the self-attention mechanism, and use a DCN to cluster sememe features. Finally, we completed the prediction using only the 10-dimensional sememe space in English. Moreover, One of the reasons why sememe has yet to be widely employed is that it does not satisfy the requirements of specific fields. We suppose that if the sememe can represent the basic meaning of a word, then replacing the word with its sememes does not change its original meaning. More specifically, we solve the problem of Controlled Defining Vocabulary (CDV) [16] coverage by replacing the definitions of some words that CDV does not cover with definitions consisting of sememe. Thus, we propose a data augmentation method for building SKB via reconstructing dictionary definitions, which can effectively construct SKB to cover specific fields (detailed in Chapter 2).

For modeling utilizing heterogeneous information, we tried two main natural lan-

guage processing tasks, Short Text Classification (STC) [17] and Recommendation System [18] (detailed in Chapter 3 and Chapter 4 respectively).

- Compared to long texts, due to limited length, short texts lack context information and strict syntactic structure, which are necessary to text understanding. One approach is to construct a heterogeneous information network by referencing external entity knowledge base information [19] or using topic models to discover latent topics information in the original corpus [20]. However, the introduction of this external knowledge has a general disadvantage because, with the increasing amount of information, the senses of an entity go far beyond its definition. Therefore we proposed an alternative way to construct entity networks, using SKBs to construct entity network connections. Extensive experiment results showed that our method outperforms the baseline model by 0.57 percentage points in accuracy and 0.08 percentage points in F1 score (detailed in Chapter 3).
- Researchers quickly find and understand the articles relevant to their research remains challenging due to the rapid iteration of technologies and the ever-increasing volume of scientific articles. In this thesis, we propose a method for question generation based on unsupervised multi-hop question answering to adapt to the beginner’s questioning style. we aim to add heterogeneous information while encoding the syntactic structure information of text to improve the accuracy of downstream tasks. For this purpose, we have developed an NLP paper recommendation system. In extensive experiments, our method shows comparable performance, we add some heterogeneous information (such as the title, author, and sememe) to the baseline model to further improve the accuracy of the paper recommendation.  $acc@1$ ,  $acc@10$  and  $acc@100$ <sup>1</sup> improved by  $65.38 \rightarrow 70.26$ ,  $77.40 \rightarrow 84.95$  and  $84.89 \rightarrow 94.36$  (detailed in Chapter 4).

---

<sup>1</sup> $acc@1/10/100$  denotes the accuracy that target papers appear in top 1/10/100, higher better

# 2. Word Representation

In this chapter, we will briefly introduce the mainstream word representation models and analyze their advantages and disadvantages in section 2.1. Then we will explain how our method addresses these problems in section 2.3, then present our method in detail. Finally, the experimental results are summarized in section 2.4.

## 2.1 Related Work

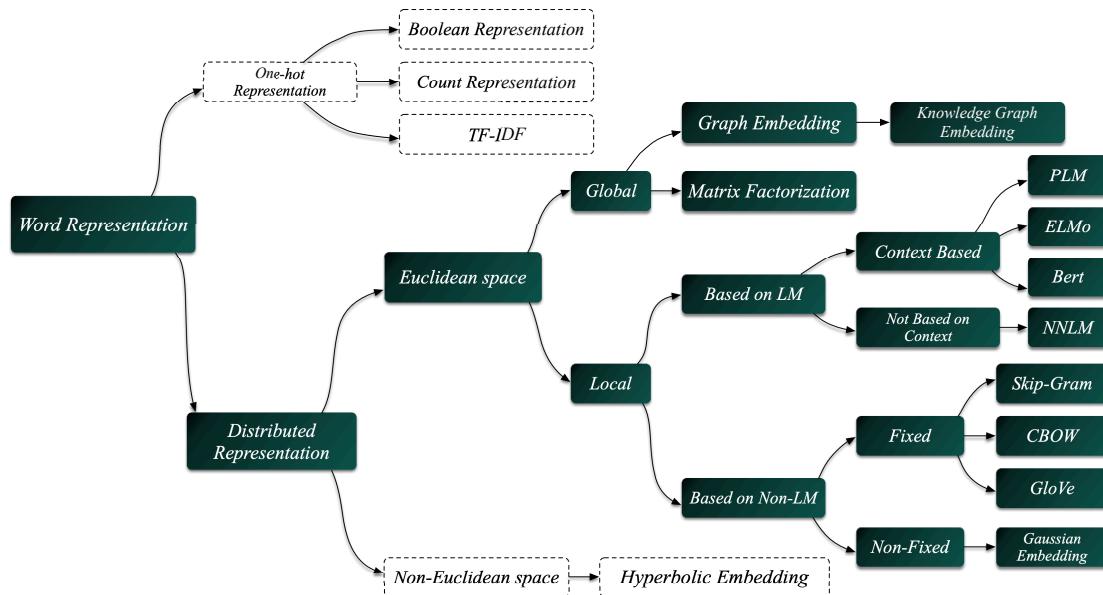


Figure 2.1: Category of Word Representation

How to represent a word as a vector is the most basic and foremost technology in NLP. It has a long history [21–23]. Many various models were proposed for computing word representations, including the famous Latent Semantic Analysis (LSA) and Latent Dirichlet Allocation (LDA), although they are computationally expensive. We can use a tree to summarize the word representation, as shown in Figure 2.1. Since the one-hot

based method is a sparse representation and cannot express the similarity of words. In this thesis, we only focus on distributed representation of words.

There are two classes of word representations in the Euclidean space, one is to consider the global information of the corpus (e.g., Matrix Factorization (MF) [24]), and the other is to only focus on the information of the current window (e.g., Skip-Gram, CBOW [25] etc.). MF-based methods such as Singular Value Decomposition (SVD) are not described in this thesis. The concept of the Skip-Gram is to maximize the probability of predicting nearby words for a known central word  $w_i \in C$  in the current window, when the window size is  $m$ , the likelihood function can be written as

$$\prod_{i=1}^{|C|} \prod_{-m \leq j \leq m, j \neq 0} P(w_{i+j} | w_i). \quad (2.1)$$

On the other hand, the CBOW model predicts the central word as the target word according to its context. The context is represented by words contained in a fixed-size window around the central word (like Figure 2.2). Some researchers combine MF and Skip-Gram

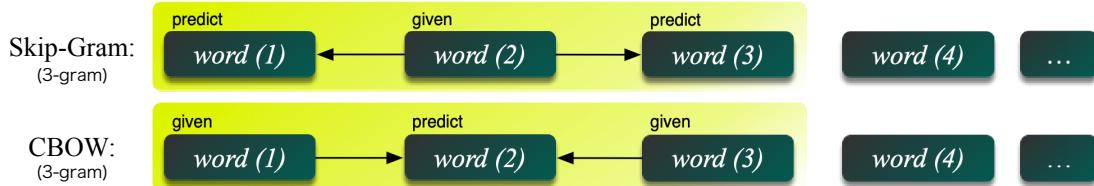


Figure 2.2: Word Representation for Non Language Models

to compensate for word representation's computational complexity and lack of holistic view issues, such as Global Vectors for Word Representation (GloVe) [26]. Moreover, since the word representation learned by GloVe is fixed, it does not consider the word confidence. Thus the Gaussian Embedding [27] is born. He utilized a Gaussian distribution to represent is a word. The representation of a word can be written as

$$Embedding_{word} \sim \mathcal{N}(\mu, \Sigma). \quad (2.2)$$

We can fast assess a word's confidence by the variance of Gaussian distribution. KL divergence [28] is usually used to compare the similarity of two word distributions.

### 2.1.1 Contextualized Representation

The earliest way to learn word representations is the Feedforward Neural Net Language Model (NNLM), proposed by Bengio et al. in 2003 [29]. However, NNLM has a fatal shortcoming. Its window is limited. That is, when predicting the next word, it only considers the number of words in the previous window size, not all previous words. This

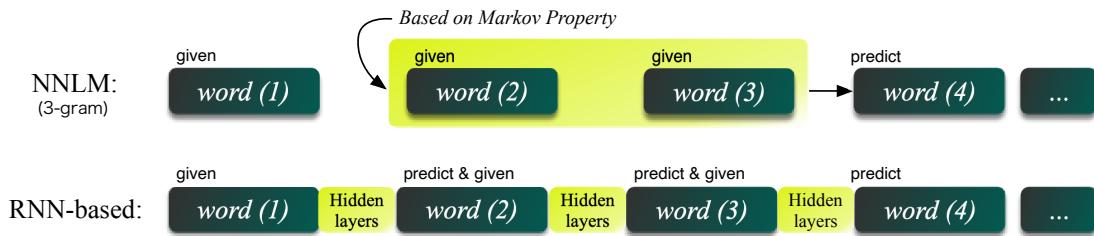


Figure 2.3: NNLM vs. RNN-based

problem can be solved using sequence encoding-based Bidirectional Language Models or Masked Language Models (e.g., ELMo [30], BERT, etc.).

### 2.1.2 Bidirectional Language Models

Word2Vec trains a fixed word vector for each word through a large-scale corpus but not addressing the polysemous word problem. However, the bidirectional language model solves this problem by increasing the neural network's depth. In an RNN-based encoder-decoder machine translation system, [31] showed that the representations learned at the first layer in a 2-layer LSTM encoder are better at predicting POS tags than second layer. Finally, the top layer of an LSTM for encoding word context [32] has been shown to learn representations of word sense. A typical case is ELMo (Figure 2.4), which uses a deep biLSTM to compute semantic-level representations of words. Each biLSTM layer outputs a context-dependent representation  $\overrightarrow{\mathbf{h}}_{i,l}^{LM}$  or  $\overleftarrow{\mathbf{h}}_{i,l}^{LM}$  where  $l = 1, \dots, L$ . The top layer biLSTM output,  $\overrightarrow{\mathbf{h}}_{i,L}^{LM}$  or  $\overleftarrow{\mathbf{h}}_{i,L}^{LM}$  is used to predict the next word  $w_{i+1}$  or  $w_{i-1}$  with a Softmax

layer. The word representation can be defined as

$$\{\mathbf{w}_i, \overrightarrow{\mathbf{h}}_{i,l}^{LM}, \overleftarrow{\mathbf{h}}_{i,l}^{LM} \mid l = 1, \dots, L\}, \quad (2.3)$$

Then we concatenate bi-directional hidden states and the word embedding to obtain the word representations.



Figure 2.4: Embedding from language models

### 2.1.3 Masked Language Models

BiLM actually learns two separate recurrent neural networks, which is different from what we want. Then how to employ a simpler network to implement it becomes a topic of interest for researchers. However, Masked LM is one of the solutions. Masked LM was proposed as early as 1953, but was only used extensively in the advent of BERT like Figure 2.5. BERT is for pre-training Transformer's [33] encoder. Bert randomly masks

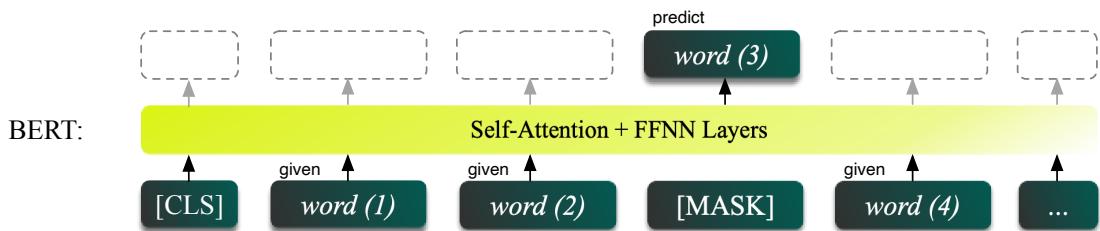


Figure 2.5: Bidirectional Encoder Representations from Transformers

words, uses the masked words as labels, and predicts them using a Scaled Dot-Product Attention mechanism. The input consists of queries and keys of dimension  $d_k$ , and values of dimension  $d_v$ , then compute the dot products of the query with all keys, and apply a

softmax function to obtain the weights on the values as

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V.$$

$Q$ ,  $K$ , and  $V$  are the queries, keys, and values matrix. The values matrix is usually utilized in pre-training to predict the masked words. The values here are defined as packed word embedding matrices. Finally, the target word is queried in the vocabulary through a feed-forward neural network, which consists of two linear transformations with a ReLU activation in between.

## 2.2 The Completeness of Word Representations

A common drawback of all these models such as Bidirectional Language Models and Masked Language Models etc. is that they ignore the completeness of the word representations. Learning representations of a word by its co-occurrence ignores many characteristics, which can characterize the general meaning of a word but cannot predict a word with certainty, as in Figure 2.6. Word co-occurrence can effectively discover the features

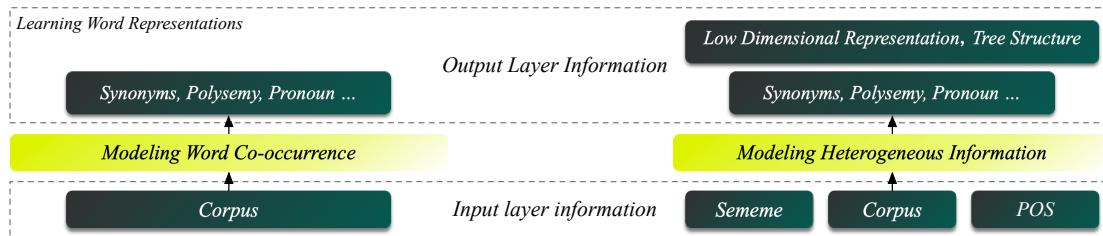


Figure 2.6: Modeling Heterogeneous Knowledge Learning Word Representations

of synonyms and polysemous words. However, this approach relies heavily on large corpora. We found that this knowledge is often entirely hidden in the sememe knowledge, and the same performance can be obtained by merely modeling the sememe knowledge. In the remaining sections, we do not elaborate too much on how to learn word vectors through sememes but focus on how to build a sememe knowledge base and expand the

sememe knowledge base, which is the most critical step to applying sememe knowledge base better.

## 2.3 Building Sememe Knowledge Base by Deep Clustering Network

Because modern people use more diversified words, some words have more than one meaning. However, No matter how a word changes its meaning, it is always composed of several primary and single meanings. In linguistics, a sememe is defined as the minimum semantic unit of human languages. Linguists believe that all languages have the same limited sememe space.

At this stage, people use manual annotation to build a sememe KB, such as HowNet SKB, which uses about 2,000 language-independent sememes to manual annotate senses of over 100 thousand Chinese and English words). Moreover, cooperate with the multilingual encyclopedic dictionary as BabelNet to build a multilingual SKB as [10]. However, semantics is an iterative system of continuous learning, HowNet is made purely by hand, and there is no interface for learning and evolution. And the meaning of words is a collection of multiple attributes. The construction of manual annotations will be biased by humans and ignore some word attributes.

To solve the flaws of manual labeling. In this thesis, we tried to construct sememes in an unsupervised manner. Our method is motivated by [34] multiple senses of a word reside in linear superposition within the standard word embeddings [25], and GloVe. Our idea is that if the surrounding words determine the meaning of a word, then the word's current meaning is determined by the weighted summation of the meanings of the surrounding words. According to this idea, we segment the meaning of each word through the self-attention mechanism based on word embedding. We took the sub-meaning space of words by weighted summation of each word in each sentence as the original space of sememes for clustering.

However, the words after the weighted summation still have the original words' dimensions, which is unsuitable for clustering tasks. Because each word in each sentence in the corpus will generate a single word meaning, this will consume much memory for storage and calculation of clustering space. Moreover, the sememe vector space with a single word meaning should have a relatively low dimensionality. However, although many data clustering methods have been proposed, conventional clustering methods usually perform poorly on high-dimensional data due to the inefficiency of similarity measures used in these methods. Furthermore, Word embedding has a highly complex data underlying structure. We want to find an effective method that can cluster a large amount of high-dimensional data.

In recent years, owing to the development of deep learning, deep neural networks (DNNs) can be used to transform the data into more clustering-friendly representations due to its inherent property of highly non-linear transformation (since DNNs can approximate any continuous mapping using a reasonable number of parameters [35]). We hope to learn the low-dimensional minimum meaning of each word through the DNN to make it more friendly for clustering.

### 2.3.1 Deep Sememe Clustering

In this study, we use the simplest and most effective deep clustering network (DCN) [36]. DCN is one of the most remarkable methods in this field. It first learns the low-dimensional representation of the data by pre-training a DNN and then clusters the low-dimensional data as the initial value. Finally, the clustering effect is optimized through continuous iterative learning of low-dimensional space. It is fully compliant and can be applied to our sememe clustering(as shown in Figure 2.7).

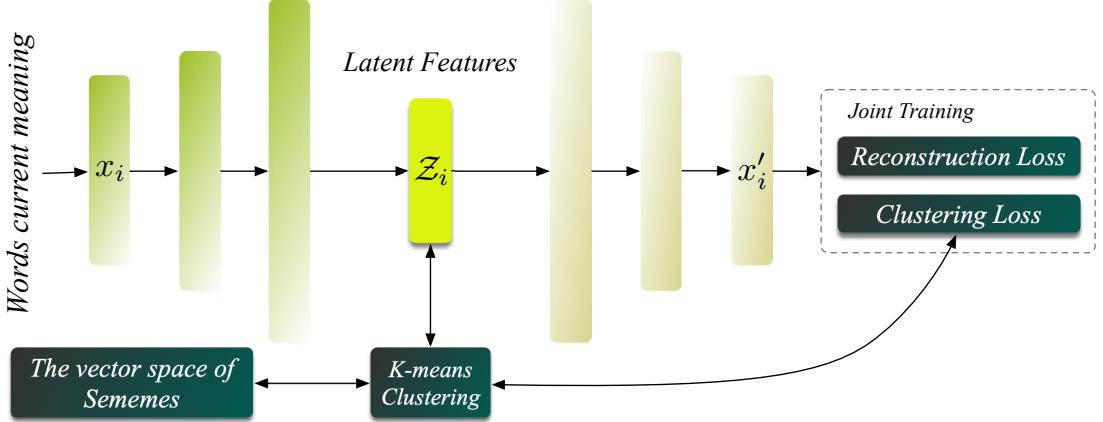


Figure 2.7: Construction of sememes based on DCN

## Dimensionality Reduction

DCN adopts an autoencoder (AE) [37] to learn clustering-friendly representations. AE is a powerful method to train a mapping function based on DNNs, which ensures the minimum reconstruction error between the coder layer and data layer. Specifically, the approaches look into an optimization problem of the following form:

$$L_n = \min_{\mathcal{W}, \mathcal{Z}} \sum_{i=1}^N \ell(\mathbf{g}(\mathbf{f}(\mathbf{x}_i; \mathcal{W}); \mathcal{Z}), \mathbf{x}_i), \quad (2.4)$$

where  $\mathbf{f}(\cdot; \mathcal{W})$  denotes the nonlinear mapping function and  $\mathcal{W}$  denote the set of parameters, i.e.,

$$\mathbf{f}(\cdot; \mathcal{W}) : \mathbb{R}^M \rightarrow \mathbb{R}^R, \quad (2.5)$$

$\mathbf{f}(\mathbf{x}_i; \mathcal{W})$  is the encoder network output given a set of data samples  $\{\mathbf{x}_i\}_{i=1,\dots,N}$ , where  $\mathbf{x}_i \in \mathbb{R}^M$  and  $R \ll M$ . Since the hidden layer usually has smaller dimensionality than the data layer, it can help find the most salient features of data. where  $\mathbf{g}(\cdot; \mathcal{Z}) : \mathbb{R}^R \rightarrow \mathbb{R}^M$  denotes the reconstruction function and  $\mathcal{Z}$  denote the set of parameters. In the construction of sememes, it can help us map the low-dimensional sememes space to the original dimensional sememes space for evaluation (shown in Figure 2.8), we will explain in detail

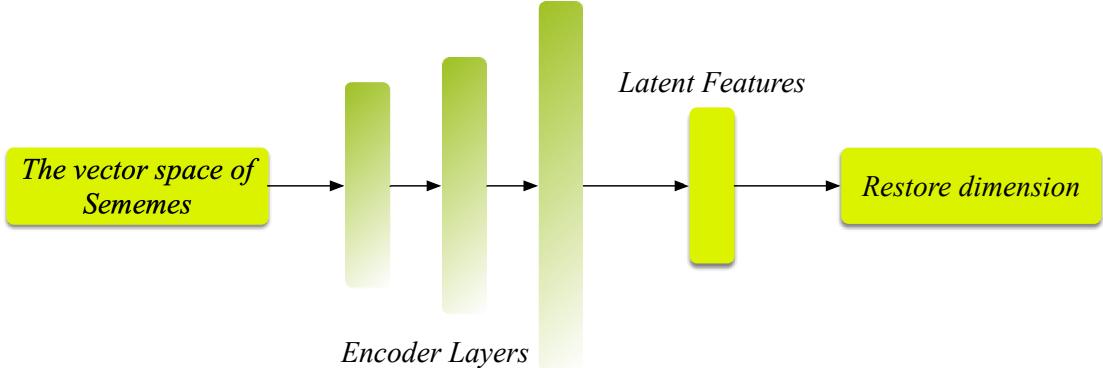


Figure 2.8: Restore data dimensions

in the experimental part of this article. The function  $\ell(\cdot) : \mathbb{R}^M \rightarrow \mathbb{R}$  is a certain loss function that measures the reconstruction error. In the DCN, the least square loss is adopted as the reconstruction error, i.e.,  $\ell(\cdot) = \|\cdot\|_2^2$ .

## Clustering

The optimization criterion of DNC is to connect DNN-based DR and clustering methods. The clustering method can be replaced, we only used the simplest and most effective K-means [38] in our research, which is also the clustering method adopted in the original work. The task of K-means is to group the  $N$  data samples into  $K$  categories by optimizing the following cost function:

$$L_c = \min_M \sum_{i=1}^N \|f(x_i; \mathcal{W}) - Ms_i\|_2^2 \quad (2.6)$$

where  $M \in \mathbb{R}^{R \times K}$  is the sememe space we want to get, where  $s_i \in \mathbb{R}^K$  is the assignment vector of data point  $i$  which has only one non-zero element, i.e.,  $Ms_i \in \mathbb{R}^R$ ,  $s_i$  helps us to find the vector closest to the  $f(x_i; \mathcal{W})$  in the  $M$  matrix, and then update the  $Ms_i$  to achieve our goal by continuously updating the  $M$  matrix. If we set  $j$  as each element in

$s_i$ , then the  $s_i$  vector can be defined as follows:

$$s_{i,j} \leftarrow \begin{cases} 1, & \text{if } j = \underset{k=\{1,\dots,K\}}{\operatorname{argmin}} \|f(x_i; \mathcal{W}) - M_k\|_2 \\ 0, & \text{otherwise.} \end{cases} \quad (2.7)$$

For the above formula, we must first know the distribution of the  $M$  matrix to calculate, thus we must first obtain the low-dimensional representation of all the data through the AE, and obtain the initial matrix  $M$  by clustering all the low-dimensional data in advance distributed.

## Optimization

Finally, we use stochastic gradient descent (SGD) to iteratively optimize the encoder parameter  $\mathcal{W}$ , the decoder parameter  $\mathcal{Z}$  and the sememe space  $M$  through the following formula:

$$\min_{\mathcal{W}, \mathcal{Z}, M} \sum_{i=1}^N (L_n + \lambda L_c) \quad (2.8)$$

$\lambda \geq 0$  is a regularization parameter which balances the reconstruction error versus finding K-means friendly latent representations.

### 2.3.2 Cross-lingual Sememe

Linguists have discovered that using a sememe space can be applied to any language. thus we proposed whether we can learn the sememe space using only a common word embedding that has been aligned in a single vector space, we only need to learn a sememe space and it can be applied to all languages (as shown in Figure 2.9).

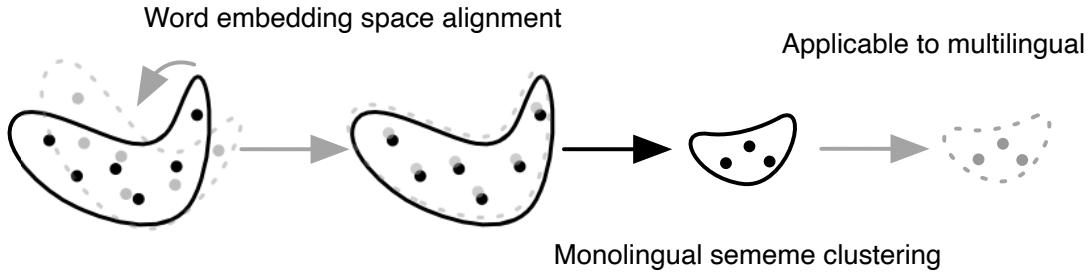


Figure 2.9: After aligning the word embedding space, only using monolingual word embedding for sememe clustering can be applied to other languages.

### 2.3.3 Experiments

#### Pre-processing

In the entire sememe clustering process, we must first obtain the original sememe space before clustering. Since we assume that each word has only one meaning in the current sentence, we need to find all the sentences containing this word in the entire corpus if we want to find all the meanings of a word. Therefore we need to create sentence indices for each word to find which sentence index contains this word in the corpus.

#### Sentence Indices Dictionary (SID)

We intercept 5GB in the Wikipedia corpus as a raw corpus for preprocessing and use SentencePiece [39] 's BPE [40] tokenizer to segment the raw corpus. We fixed the vocabulary of BPE to 200k. In terms of sentence segmentation, to make the meaning of each sentence as complete as possible, we did not directly use a fixed length for segmentation but used 42 kinds of symbols like “.,?!()” for segmentation. Moreover, because we use the self-attention mechanism to sum the proportions of all words in the sentence to get the current word meaning, if the sentence is too long, the weight of the important words will be assigned to the secondary words, which is difficult to get the obvious features of the current word meaning, thus we filter out sentences greater than 20 words and less than 2

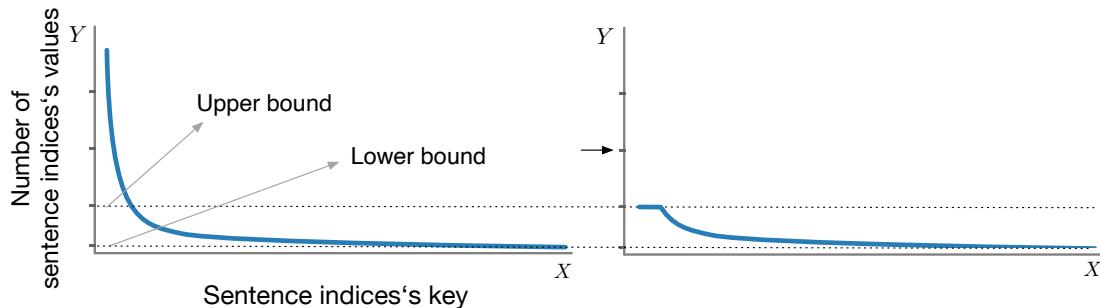
words. Finally, we got 60 million qualified sentences to create sentence indices.

After constructing the sentence indices, we found that the number of sentences for specific words is enormous. Since our clustering network is to put all the meanings of all words into a space for clustering, If the sum of sentences for some words is close to half of the entire clustering space, the result of our clustering will tend to the meaning of these words. Therefore we have to balance the number of sentences in the sentence indices. To solve this problem, we implemented the following methods:

1. Delete stop words (We use stop words defined in NLTK for filtering <sup>1</sup>).

2. Set the upper and lower bounds for the number of sentences (shown in Figure 2.10). We set the upper and lower limits to 5k and two batch numbers respectively. If it exceeds 5k, randomly select 5k sentences from it, and if it is less than two batch numbers, discard the word.

3. We multiply the original data with an expansion coefficient  $e \in \mathbb{R}$ ; and  $e > 0$  be-



**Figure 2.10:** The illustration shows the distribution of the number of sentences. The X axis is the index of the word, and the Y axis is the number of sentences corresponding to the word. On the right is the distribution of the number of sentences after setting the upper and lower bounds.

fore the normalization function of self-attention, thus that it is not easy to lose important features when encountering long texts, In this experiment, we set  $e = 4$ .

Finally, the number of sentences actually participating in clustering after being bal-

---

<sup>1</sup><https://gist.github.com/sebleier/554280>

anced is only 40 million. Note that the above parts are implemented locally, and will be used as input in DCN later.

## Clustering Process

### Pre-training

The primary purpose of pre-training is to learn a low-dimensional cluster center matrix as the initialization of the sememe space  $M$  (shown in Algorithm 1). Since the global distribution needs to be obtained during clustering, it is tremendous, but we cannot build this matrix in memory. Therefore, we adopted a sampling method when clustering. Each word only samples sentences of two batch numbers for clustering and loops ten times. Then the clustering results of ten times are clustered again to obtain the final cluster center initialization matrix.

In the experiment, we detection that the trained loss curve is jagged, Which may be caused by the uneven distribution of data samples. Since our method is to loop each word and perform the self-attention calculation on each word in turn. Although we scrambled the appearance order of the words before the loop, the loss was not smooth due to the inconsistency of word embeddings and sentences quality. Thus, according to the amount of memory used, we loop 500 words at a time and fully disrupt these words after doing self-attention (shown in Algorithm 1). We defined the DCN-encoder size as [200, 200, 800, 10, 800, 200, 200], the input size is 300 dimensional aligned fasttext word embedding(Bojanowski et al. [41]). The learning rate is 0.003. And, the cluster centers are set to 2048, because we hope to make the cluster centers more dispersed while retaining the semantics of the sememe space to the greatest extent, we use t-SNE [42] to reduce the dimensionality of the sememe space and analyze it by density, and found that 2k clusters can represent a rough meaning. If there are more clusters, the semantics will be more detailed. Figure 2.11 presents the reconstruction error of the autoencoder during the pre-training stage,

---

**Algorithm 1:** Sememe DCN Pre-training

---

**Data:** Sentence indices dictionary  $SID$ , Word embeddings  $WE$

**Result:** Initialization matrix  $M$

```
// Train an autoencoder network

1 while  $epoch$  do
2   while  $all\ vocabulary$  do
3      $batch \leftarrow \text{SELFATTENTION}(WE; SID);$ 
4     while  $batch$  do
5        $\mathcal{W}^*, \mathcal{Z}^* \leftarrow L_n(\mathcal{W}; \mathcal{Z});$ 
6     end
7   end
8 end

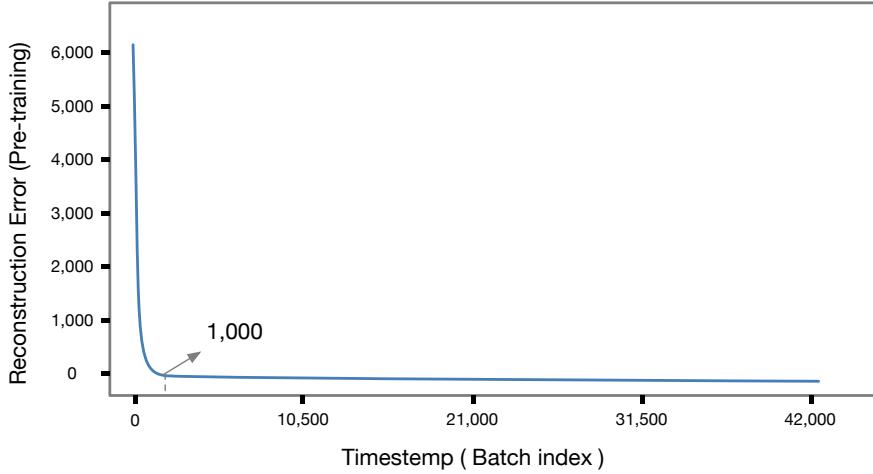
// Initialize the clustering space

9  $latent\_data \leftarrow \text{new Array};$ 

10 while  $all\ vocabulary$  do
11    $batch \leftarrow \text{SELFATTENTION}(WE; SID);$ 
12   while  $batch$  do
13      $latent\_data.append(\mathbf{f}(x_i; \mathcal{W}^*));$ 
14   end
15 end

16  $M \leftarrow \text{KMEANS}(latent\_data);$ 
```

---



**Figure 2.11:** Reconstruction error of the autoencoder during the pre-training stage. (We set a batch size of 64 sentences and observed that it quickly converges and stabilizes within 1,000 batches)

## Fine-tuning And Results

In the Fine-tuning stage, we use the original DCN to optimize the sememe space, and we found that the reconstruction loss of the autoencoder in the Fine-tuning phase is consistent with the pre-training phase and there is no downward trend.

Finally, we predicted the sememes of some words and their probabilities and used the existing English *SID* to search for sememes in other languages (shown in Table 2.1). Through the table, we found that sememe can express certain characteristics of words. The decimals in the table represent the probability of this sememe in the overall sememe space (Only the 6 with the highest probability are listed). The sememe we calculated is just a vector of the original word embedding space, it is not a word, therefore we find the nearest three words by cosine similarity to represent the meaning of sememe. Since the sememes predictions in other languages depend entirely on the quality of English sememes and the degree of alignment with English, and you can use the data set that MUSE has already trained, thus we will not show the relevant results of other language predictions here.

	1st	2nd	3rd	4th	5th	6th
projector	interfacing; device; directionality (0.03854)	bluescreen; screensavers; audio/visual (0.03646)	wirelessly; handsets; functionality (0.03229)	imageworks; spotlighting; showcase (0.02812)	film; movie; screenplay (0.02083)	windshields; plexiglass; windscreens (0.01667)
woodland	grassy; thickets; vegetation (0.07993)	outcroppings; bottomlands; riverbeds (0.05168)	porcupines; birds; raccoons (0.04127)	lakeview; riverside; park (0.02564)	vegetation; habitats; habitat (0.02404)	northeast; southwest; northwesternmost (0.02143)
stamen	prickles; fleshy; leathery (0.03854)	grasses; berries; shrubs (0.03768)	laterally; tapering; serrations (0.03401)	cottonwoods; cattails; meadowsweet (0.02941)	pinkish; yellowish; brownish (0.02574)	protruding; shallowly; lengthwise (0.01195)

Table 2.1: Sememes Cluster Prediction in English. More examples can be found in Appendix Table A1 .

## 2.4 Conclusion

In this chapter, we discuss the construction and expansion of SKB in the unsupervised case and prove that sememe exists in low-dimensional spaces, which coincides with the properties of sememe. Moreover, we indirectly revealed that the pre-trained language model implies the representation information of sememe in the learning process of the word representation.

### 3. Short Text Representation

Since people prefer to utilize more concise text to express what they want to say in product reviews, movie reviews, queries, or tweets, utilizing NLP techniques to help understand these short texts would have been inevitable [17]. Compared to sentences, due to limited length, short texts lack context information and strict syntactic structure, which are necessary to text understanding.

A practical solution is using graphs to represent text information and learning text representations through Graph Neural Networks (GNNs). GNNs are essentially graph representation learning models. GNNs learn embeddings for each node in the graph and aggregate the node embeddings to produce the graph embeddings, which can also be multi-layered such as Figure 3.1.

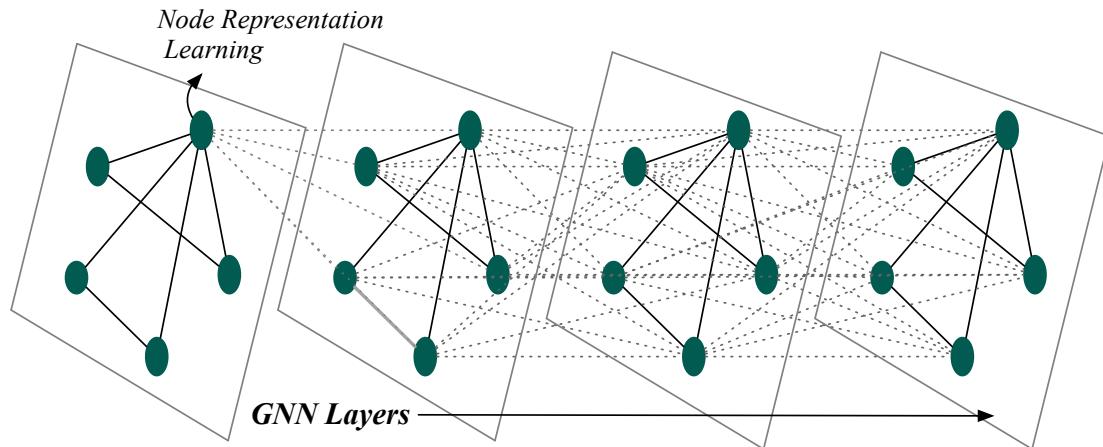


Figure 3.1: Multilayer Graph Neural Networks

Conventionally, text sequences are considered tokens such as TF-IDF in NLP tasks. Thus, popular deep learning techniques such as recurrent neural networks [43] and convolutional neural networks [44] have been widely applied for modeling text sequences. However, these methods are unable to express structural information (i.e. syntactic pars-

ing trees like dependency and constituency parsing trees). The most widespread solution is to encode structural information by constructing graphs and using graph neural networks (GNNs) [45–48]. In practice, many graphs have various node and edge types, such as knowledge graph, AMR graph, etc., which are called heterogeneous graphs (such as Figure 3.2) . Our motivation is to use heterogeneous graphs to encode heterogeneous information e.g., Text-GCN [49], HGAT [50], SHINE [51] etc.

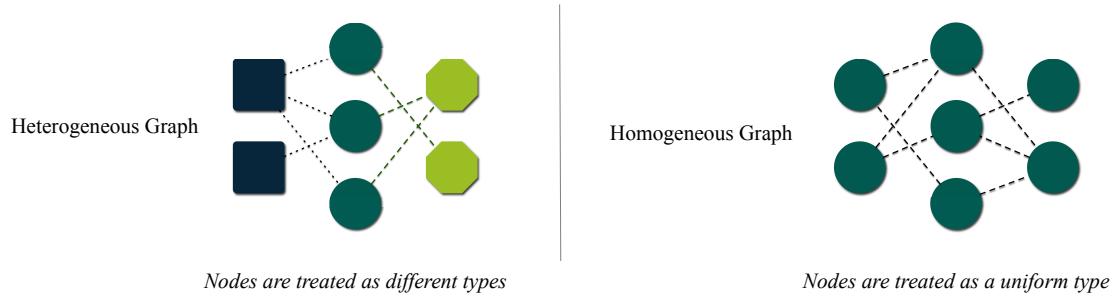


Figure 3.2: Heterogeneous vs Homogeneous Graph

With the enhancement of text specialization, the scale of knowledge nodes will also become immense. Therefore we take advantage of the properties of sememe to significantly reduce the scale of knowledge nodes and maintain the expressiveness of the original model. However, since the vocabulary in SKBs is limited, it is difficult for manually constructed SKBs to encompass many proper nouns. Due to its lower accuracy, the DCN-based approach is challenging to apply to practical downstream tasks. We thus design a simple SKB augmentation method, which significantly improves the coverage of SKBs on proper nouns and effectively compresses the size of graph neural networks. We will describe this method in detail in Section 3.3.

This chapter explores and explains how GNNs can be used to learn short text representations. We divide this work into four parts. First, we introduce the basics of GNNs, and a state-of-the-art method for solving short texts is introduced in Section 3.1. In Section 3.2., we analyze the problems of previous methods. Finally, in Sections 3.3 and 3.4,

we elaborate on our proposed solution.

## 3.1 Related Work

In the following two subsections, we will introduce the leading technology of graph embedding, Graph Attention Networks in subsection 3.1.1, and a heterogeneous network based on Graph Attention Networks in subsection 3.1.2.

### 3.1.1 Graph Attention networks

The learning process of the graph neural network is to map the embedded representation of the nodes in the previous layer to the next layer through a filter function or named a graph filter as  $\text{FILTER}(\cdot, \cdot)$ . Graph filtering layers are stacked to layers to generate final node embeddings. In order to determine the relationship between nodes, generally, this relationship is expressed as an adjacency matrix  $A \in \mathbb{R}^{n \times n}$ . Therefore the learning process of GNN can be written as

$$\mathbf{h}_i^{(l)} = \text{FILTER}(A, \mathbf{H}^{l-1}) \quad (3.1)$$

where

$$\mathbf{H}^{l-1} = \{\mathbf{h}_1^{(l-1)}, \mathbf{h}_2^{(l-1)}, \dots, \mathbf{h}_n^{(l-1)}\} \quad (3.2)$$

denotes the node embeddings at the  $l - 1$ th layer.

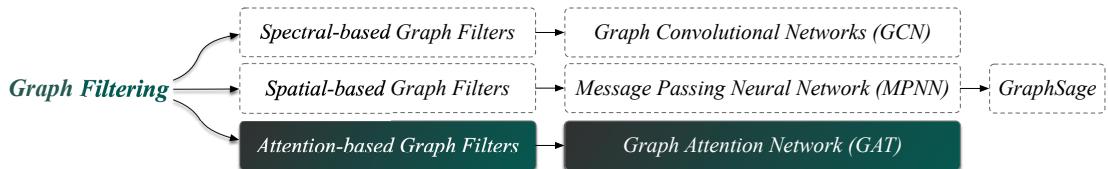


Figure 3.3: Types of Graph Filters

The key to the graph neural network's success depends on using graph filters. However, there exists a variety of implementations of graph filters such as Spectral-based Graph Filters [47], Spatial-based Graph Filters [48, 52], Attention-based Graph Filters [53]. This thesis only briefly introduces a primary graph filter, which will help the understanding of later sections.

## Attention-based Graph Filters

Inspired by the successful applications of the attention mechanism of the Transformer model, Petar Velickovic et al. proposed an attention mechanism for graphs called Graph Attention Network (GAT) [53]. The attention mechanism considers the semantic similarity between the target node and each neighboring node and assigns higher attention scores to important neighboring nodes when performing the neighborhood aggregation. The graph filter function can be defined as

$$\mathbf{h}_i^{(l)} = \text{FILTER} \left( A, \mathbf{H}^{(l-1)} \right) = \sigma \left( \sum_{v_j \in N(v_i)} \alpha_{ij} \vec{W}^{(l)} \mathbf{h}_j^{(l-1)} \right), \quad (3.3)$$

where  $\sigma(\cdot)$  is a non-linear function,  $\vec{W}^{(l)}$  is the weight matrix at  $l - th$  layer.  $\alpha_{ij}$  is defined as the attention scores for each pair of nodes  $v_i$  and  $v_j$ , formulated as

$$\alpha_{ij} = \frac{\exp \left( \text{LeakyReLU} \left( \vec{u}^{(l)T} \left[ \vec{W}^{(l)} \mathbf{h}_i^{(l-1)} \| \vec{W}^{(l)} \mathbf{h}_j^{(l-1)} \right] \right) \right)}{\sum_{v_k \in \text{one hop neighborhood}(v_i)} \exp \left( \text{LeakyReLU} \left( \vec{u}^{(l)T} \left[ \vec{W}^{(l)} \mathbf{h}_i^{(l-1)} \| \vec{W}^{(l)} \mathbf{h}_k^{(l-1)} \right] \right) \right)}, \quad (3.4)$$

where  $\|$  is the vector concatenation operation.

In practice, many graphs have various node and edge types, which are called heterogeneous graphs, such as heterogeneous information network (HIN) as shown in figure 3.4. The following subsection will introduce a state-of-the-art method for short text classification, and our model will be modified later using this model as a baseline.



Figure 3.4: Heterogeneous Information Network for Short Texts

### 3.1.2 Hierarchical Heterogeneous Graph Representation Learning

Hierarchical Heterogeneous Graph Representation Learning (SHINE) [51] models the interaction between words, POS, and entities mainly through word-level graphs, and dynamically learns the similarity between short texts, enabling the propagation of tags among connected similar short texts.

This part consists of three subgraphs: Word Graph, POS Tag Graph and Entity Graph. The introduction of subgraphs is to make short texts have richer contextual feature information. The PMI score between word nodes  $v_w^i$  and  $v_w^j$  is calculated to construct word adjacent matrix  $[\mathbf{A}_w]_{ij}$  as  $\max(\text{PMI}(v_w^i, v_w^j), 0)$ . The construction of part-of-speech tag graph is the same as word graph, and the entity graph is constructed by replacing the PMI with the cosine similarity. Finally, these features are concatenated to the complete short text features as

$$x_{all}^i = \text{CONCATENATE}(x_{word}^i, x_{pos}^i, x_{entity}^i). \quad (3.5)$$

The adjacency matrix can be expressed as

$$[\mathbf{A}_{all}]_{ij} = \begin{cases} (\mathbf{x}_{all}^i)^\top \mathbf{x}_{all}^j & \text{if } (\mathbf{x}_{all}^i)^\top \mathbf{x}_{all}^j \geq \delta_{all} \\ 0 & \text{otherwise} \end{cases} \quad (3.6)$$

where  $\delta_{all}$  is a threshold used to sparsity  $\mathbf{A}_{all}$  such that short documents are connected only if they are similar enough. Ultimately, two layers of GCN are used to learn the

representation of short text as

$$\text{SOFTMAX}(\mathbf{A}_{all} \cdot \text{ReLU}(\mathbf{A}_{all}\mathbf{X}_{all}\mathbf{W}_{all}^1) \cdot \mathbf{W}_{all}^2) \quad (3.7)$$

where  $\mathbf{W}_{all}^1$  and  $\mathbf{W}_{all}^2$  denote the weights of the two layers.

## 3.2 The Completeness of Short Text Representations

Since SHINE can effectively represent heterogeneous information, our experiments are implemented on this basis. However, for complete statistics, using sememes instead of entity knowledge can enrich the representation of information, as shown in Figure 3.5, because the number of sememes is limited, significantly reducing the graph's node space. In the following sections, we will utilize short text classification as a downstream task to prove our guess.

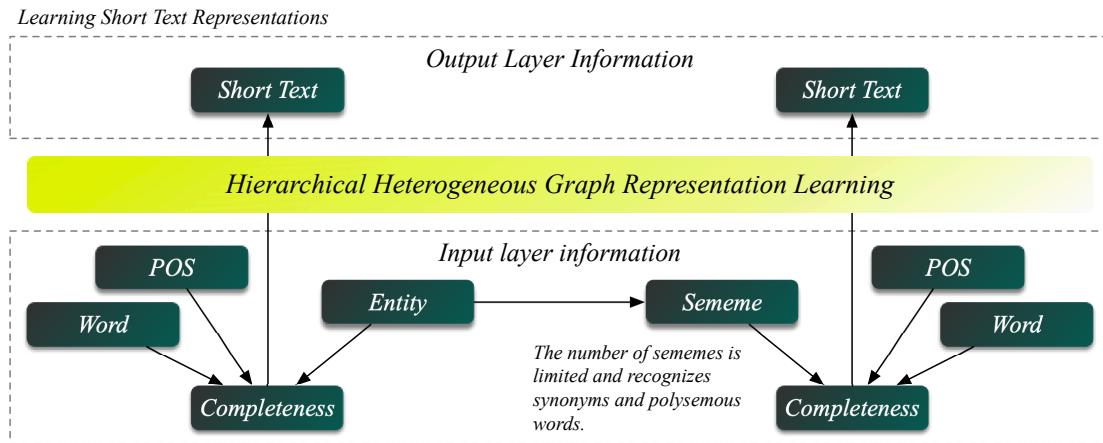


Figure 3.5: Replacing Entities with Sememes to Learn Representations

### 3.3 Data Augmentation of Sememe Knowledge Base

Manual annotation is flawed because the meaning of words is incremental with the amount of information, and it is impractical to add and modify SKB artificially. Some studies on automatic SKB construction have emerged recently. We previously proposed a method [54] based on deep clustering networks to learn sememe. Specifically, this method is based on an Auto-encoder to achieve minimal semantic clustering. Instead of generating specific language-independent sememes, it generates word embeddings with approximate sememe meaning, Regrettably, which is imprecise. Moreover, (Qi et al. 2021) [55] explored an automatic way to build an SKB via dictionaries with a Controlled Defining Vocabulary (CDV) [16], and demonstrate the effectiveness of this method; it is even superior to the most widely used HowNet SKB. The method is to extract the CDV as sememes in the dictionary definition. However A CDV is composed of high-frequency words. If the dictionary is large enough, it does not cover all words perfectly, which means some complex words can not acquire sememe.

To solve this problem, we proposed a way of reconstructing word definitions to increase the lexical coverage of CDV. We hypothesized that if the sememe can represent the basic meaning of a word, then if replacing the word with its sememes does not change its original meaning. More specifically, we solve the problem of CDV coverage by replacing the definitions of some words that CDV does not cover with definitions consisting of sememe (as shown in Figure 3.6).

Finally, we employed the sememe internal evaluation criteria defined in [56] for evaluation. Moreover, we proposed a novel method to evaluate sememe by constructing a sememe graph. Because we consider the weight of sememe when constructing the sememe graph, it performs excellently in both evaluation methods. We shown the results in section 3.3.2.

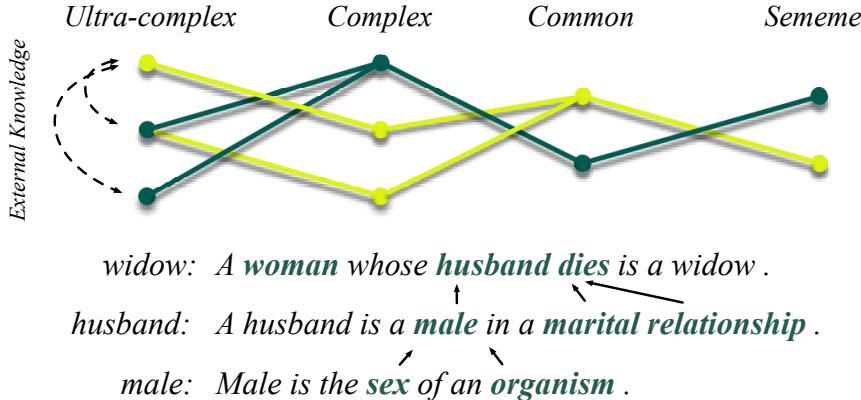


Figure 3.6: If “widow” is a complex word and “husband” is not in sememes, we can use the definition instead of “husband”.

### 3.3.1 Building Sememe Knowledge Base

This subsection will detail the SKB construction method based on reconstructed word definitions and how to build a sememe graph for evaluation.

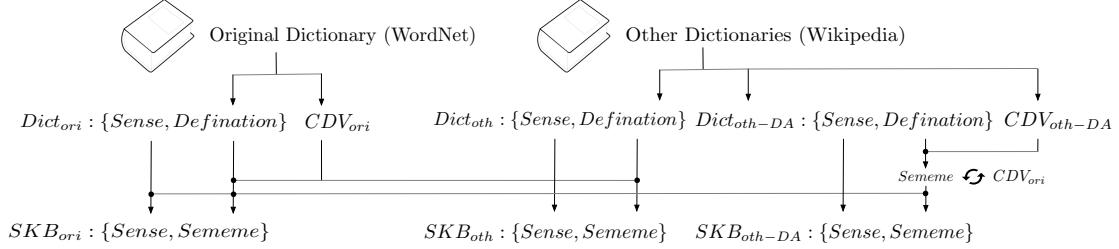
We employed the sememe search strategy of DictSKB [55]. It is intuitive to operate. Because a good sememe can represent the essential meaning of a word, it is most straightforward to extract the sememe from the word’s definition. This method starts with finding the highest frequency  $m$  words in the dictionary definition to cover as many dictionary words as possible (Previous experience:  $m \approx 2k$ ). Unlike the specialized dictionaries employed by DictSKB, we utilized WordNet<sup>1</sup> and Wikipedia<sup>2</sup> as the base corpus for building SKB. WordNet contains 0.2 *million* pairs of word senses and definitions, while Wikipedia contains 6 *million* pairs of words and detailed explanations. Our approach is divided into two modules. Since the word definitions in WordNet are shorter and more precise, we first find the initial Sememe from WordNet and construct a WordNet-based SKB by matching the annotated words in WordNet. Then we expanded it to Wikipedia

---

<sup>1</sup><https://wordnet.princeton.edu/>

<sup>2</sup><https://dumps.wikimedia.org/>

based on the WordNet-based SKB (like Figure 3.7).



**Figure 3.7: SKB Expansion Flow Chart:** We use WordNet as the original dictionary and lexical expansion through Wikipedia, sharing a sememe set (As  $CDV_{ori}$ ) between them. From the right side of the illustration, we used the look-up table method to replaced the *Definition* of  $Dict_{oth-DA}$  with the *Sememe* of  $SKB_{ori}$ , which is a straightforward operation.

## WordNet-based SKB

First, we remove the stop words<sup>3</sup> and meaningless characters and used the entity linking tool TAGME<sup>4</sup> to find the  $2k$  most frequent topic words from HowNet word definitions as the base sememes. TAGME can identify meaningful short phrases in an unstructured text and link them to a relevant Wikipedia page. This annotation process has implications that go far beyond the enrichment of the text with explanatory links because it concerns contextualization and, in some way, the understanding of the text. A case in point, the definition “*A husband is a male in a marital relationship.*” is semantically enriched by the relations with the entities “*husband*”, “*male*” and “*marital relationship*”. We then used the *link\_probability* parameter provided by TAGME to rank the entities in the word’s definition and kept only the first four entities with the highest probability as sememes. We compared the filtered results with DicSKB and HowNet (In Table 3.3.1).

<sup>3</sup><https://code.google.com/archive/p/stop-words/>

<sup>4</sup><https://sobigdata.d4science.org/group/tagme/>

SKB	#Word	#Sense	#Sememe	#AvgSem
HowNet	50,879	111,519	2,187	2.26
DictSKB+	70,218	105,160	2,046	6.03
DictSKB	70,218	105,160	1,682	2.04
WordNetSKB	121,697	163,340	2,000	1.83
WikiSKB	385,336	423,249	1,807	2.12
WikiSKB-DA	697,754	800,458	1,992	3.46
WikiSKB-DA+	697,754	800,458	1,992	5.73

Table 3.1: Statistics of WordNetSKB, WikiSKB & WikiSKB-DA. WikiSKB-DA is a data augmented version of WikiSKB, and compared with HowNet and DictSKB. The gray font represents the previous SKB results (The same is true for the following tables). #AvgSem denotes the average Sememe number per sense, and “+” represents this average over four.

## Wikipedia-based SKB

The Wikipedia-based SKB construction is roughly the same as WordNetSKB. The difference is that Wikipedia’s explanation is too detailed, and in addition, Wikipedia does not have semantic sense concepts, which significantly increases the noise of constructing SKB. To solve these problems, we took the following trick,

- Only the first two sentences of each entry in Wikipedia are adopted.
- Traverse each word in the Wikipedia entry and use NLTK to terms lemmatization.
- Use TF-IDF to remove words below the threshold from the definition. (We set the lowe bound = 4)
- Use the polysemantic annotations in Wikipedia as the sense.
- Delete the senses of polysemous words containing the meaning associated with “*film*”, “*novel*”, “*album*”, “*song*”, “*band*”, “*name*”, “*ep*”, “*game*”,

“*surname*” and “*tv series*”.

- Keep only Wikipedia entries with a one-word title.

The WikiSKB was then constructed using the same  $2k$  sememes as WordNetSKB. The statistics in Table 3.3.1. Since WordNet-based sememes do not cover the Wikipedia entry definitions perfectly, we reconstructed the entries that were not covered. In detail, we first used TAGME to extract the entities of each entry and adopted the  $2k$  most frequent entities as sememes according to the same method as WordNetSKB. Then we found these words with the same entities from WordNetSKB and replaced them with the sememes of these words. We also put the statistics of the augmented version for WikiSKB in Table 3.3.1.

## Sememe-based Graph Embedding

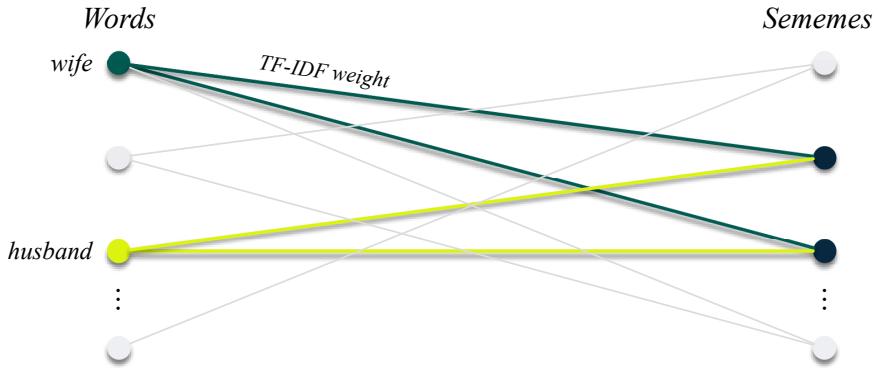
Our idea about the evaluation of sememe is to construct a bipartite graph by linking words with sememe to learn the embedding representation of words and then evaluate the quality of sememe by assessing the quality of word embedding (like Figure 3.8).

We employ second-order similarity of LINE [57] to train the node vectors regarding the graph embedding model, which is fast and intuitive. In detail, the probability of generating a neighbor node  $v_j$  given a node  $v_i$  can be expressed in the following form:

$$p(v_j | v_i) = \frac{\exp(\vec{u}'_j^T \cdot \vec{u}_i)}{\sum_{k=1}^{|V|} \exp(\vec{u}'_k^T \cdot \vec{u}_i)} \quad (3.8)$$

Where  $\vec{u}$  and  $\vec{u}'$  are denoted as the vector of  $v$  itself and  $v$  when it is a neighbor, respectively. The empirical probability as  $\hat{p}(v_j | v_i) = \frac{w_{ij}}{d_i}$ , where  $w_{ij}$  is the weight of the edge  $i, j$ ; and  $d_i$  is the degree of vertex  $i$ . If we define the importance factor  $d_i$  of the node, the loss function can be defined as

$$-\sum_{i \in V} d_i \text{KL-DIVERGENCE}(\hat{p}(\cdot | v_i), p(\cdot | v_i)). \quad (3.9)$$



**Figure 3.8:** Sememe based bipartite graph: We only learn the embedding representation of words by sememe, which means there is no line between words; we use TF-IDF for the weights of edges. The figure shows that “*wife*” and “*husband*” contain the same sememe, which should have approximate embedding representations. Note that here we do not consider the embedding representation of the sememe, so we use index instead of the word of the sememe itself.

### 3.3.2 Evaluations

This section uses two methods to evaluate our SKB: a collaborative filtering method [58] in subsection 3.3.2 and an approach based on constructing sememe graphs in subsection 3.3.2.

#### Evaluate on Consistency Check of Sememe Annotations

This method is motivated by the idea that semantically close senses should have similar sememes. It actually implements a sememe prediction process that predicts sememes for a small proportion of senses according to the sememe annotations of the other senses. We have evaluated our SKB using open source code<sup>5</sup>. For hyperparameters, we set the same hyperparameters as the original paper [55], the number of evaluating epochs is 10, the threshold is 0.8, and the descending confidence factor is 0.93. The evaluation results are in

---

<sup>5</sup><https://github.com/thunlp/DictSKB/tree/main/ConsistencyCheck>

Table 3.3.2. We discovered that the accuracy decreases with the increase of the dictionary lexicon. It is intuitive that the dictionary has many synonyms, while the sememe is static.

SKB	MAP	F1
HowNet	0.93	0.91
DictSKB+	0.88	0.86
DictSKB	0.95	0.91
WordNetSKB	<b>0.96</b>	0.87
WikiSKB	0.95	0.86
WikiSKB-AD	0.93	0.90

Table 3.2: The results on Consistency Check of Sememe Annotations: MAP score of WordNetSKB exceeds the DictSKB, which we indicated in boldface.

## Evaluate on Sememe Graph

Our ultimate goal is to build a large SKB, so we combined the knowledge of both SKBs and evaluated them on some test sets (Shown in Table 3.3.2). We first performed node embedding training using LINE<sup>6</sup>, where the size of the node embedding is 200 dimensions, the total number of training samples is 100 million, the starting value of the learning rate is 0.025, the number of negative samples is 5, and we only used second-order proximity for training. SKB is sense-based, but each word in Word Similarity Tasks has only one meaning. Since the sense of words in Wikipedia is huge, we keep only *disambiguation* and *noun* sense for each word, and found that a higher number of sememe for a sense is a more accurate representation of the meaning. It is undoubtedly. Since we use only about four sememes to represent the meaning of a sense, it may lose some semantics, but the

---

<sup>6</sup><https://github.com/tangjianpku/LINE>

Similarity Tasks	WordNet&WikiSKB	WordNet&WikiSKB+
WS-353-ALL [59]	36.23	40.00
WS-353-SIM	41.66	45.12
WS-353-REL	22.70	30.26
MC-30 [60]	26.32	31.19
RG-65 [61]	28.71	32.57

Table 3.3: The results on the Sememe graph: We merged WordNetSKB and WikiSKB. These tasks provide human scoring of the relationship between two words, thus assessing the degree of positive word relatedness. The method is to first calculate the cosine similarity of the two words and then compare them with the manual tags to calculate the Spearman correlation coefficient for scoring.

essential meaning can be kept. We released the data to reproduce the results.<sup>7</sup> Note that since the previous SKBs (As HowNet&DictSKB) did not provide a weight parameter for each sememe, we cannot compare the previous SKBs. However, to demonstrate that our SKB can better represent specialized words we extracted some specialized words in the Ohsumed dataset<sup>8</sup> for comparison (Shown in Table 3.3.2). Note that we boost the number of sememe to 5,000 to maximize the lexicon, other parameters are the same as before, and the final lexicon size of our SKB-DA is 910,369.

---

<sup>7</sup><https://github.com/SauronLee/SKB-DA>

<sup>8</sup>Ohsumed dataset: it includes medical abstracts from the cardiovascular diseases.

<http://disi.unitn.it/moschitti/corpora.htm>

Word	SKB	Sememe
clostridium	DictSKB	{cause, illness}
	SKB-DA	{bacterial, cell, swollen}
colitis	DictSKB	{cause, illness}
	SKB-DA	{colon, inflammation}
pediatric	DictSKB	non
	SKB-DA	{care, child, medical}

Table 3.4: Sememe comparison on the Ohsumed dataset. Where “*non*” means no sememe of the word, we have merged WordNetSKB and WikiSKB+ as SKB-DA. More examples can be found in Appendix Table A2.

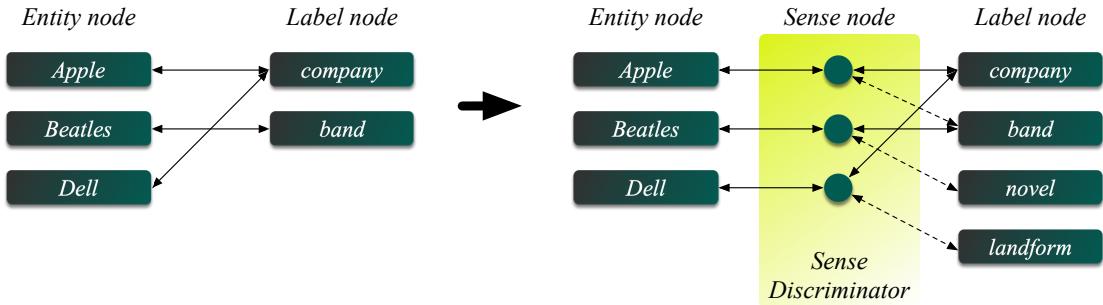
## 3.4 Modeling Heterogeneous Graph Neural Networks with Sememe Knowledge

The most important part of analyzing short text in NLP is correctly classifying them for downstream tasks, For example, sentiment analysis [62], dialogue understanding [63], news categorization [64], query intent classification [65], and user intent understanding [66].

However, compared to long texts, due to limited length, short texts lack context information and strict syntactic structure, which are necessary to text understanding. One approach is to construct a heterogeneous information network by referencing external entity knowledge base information [19] or using topic models to discover latent topics information in the original corpus [20].

However, the introduction of this external knowledge has a general disadvantage because, with the increasing amount of information, the senses of an entity go far beyond its definition. For example, “Apple” may be a multinational technology company, a rock band, or even a person’s name when it is an entity, so it is necessary to distinguish the sense of

the entity (like Figure 3.9).



**Figure 3.9:** Using Sememe Knowledge Bases to construct entity networks. We have added some sense nodes in the illustration on the right side. Before connecting the sememe nodes should first determine the correct sense nodes via sense discriminator. Note, the sense nodes will be discarded after helping us to learn the weights of the edges between the entity nodes and the sememe nodes.

Therefore we proposed an alternative way to construct entity networks, using Sememe Knowledge Bases (SKBs) to construct entity network connections. Our motivation is that introducing sememe can better uncover the semantic relationships between short texts, which move from the word level to the Sense level. Additionally, since the size of the sememe space is fixed, it can significantly reduce the size of the whole network and thus save computational resources. However, due to the lack of sememe annotation of entities in previous SKBs, We have extended the SKB using the method of [67], which uses the automatic construction of SKBs [55]. Specifically, we utilize WordNet as the base lexicon and Wikipedia as the expansion lexicon, noting that they share a Controlled Defining Vocabulary (CDV) [16] set between them.

Ultimately, we use the state-of-the-art model [51] in short text classification to learn a heterogeneous graph neural network containing the Sememe network. After extensive experiments, our method outperforms the original model by 0.57 percentage points in accuracy and 0.08 percentage points in F1 score.

In the following subsections will be described how to construct an entity SKB (In subsection 2.1) and build a graph network model based on SKB (In subsection 2.2).

### 3.4.1 Construct An Entity Knowledge Base

We use open source code for SKB-based data augmentation<sup>9</sup> [67]. It is intuitive to operate. Because a good sememe can represent the essential meaning of a word, it is most straightforward to extract the sememe from the word definition. First, we utilize WordNet<sup>10</sup> as the base lexicon, extract the words tagged by TagMe<sup>11</sup> in the word definition of WordNet and keep the set of  $m$  words with the highest frequency as sememes, then expanded to 6 *million* Wikipedia<sup>12</sup> entries based on these sememes. To cover the maximum number of entities, we set  $m = 5000$ . Statistics showed in table 3.4.1.

SKB	#Word	#Sense	#Sememe	#AvgSen	#AvgSem
HowNet <sup>13</sup>	50,879	111,519	2,187	2.19	2.26
DictSKB <sup>14</sup>	70,218	105,160	2,046	1.49	6.03
SKB-Entity	152,661	6,312,591	5,000	41.35	6.74

Table 3.5: Statistics of SKB-Entity, and compared with HowNet and DictSKB. Note, #AvgSen denotes the average sense number per word, #AvgSem denotes the average sememe number per sense.

---

<sup>9</sup><https://github.com/SauronLee/SKB-DA>

<sup>10</sup><https://wordnet.princeton.edu/>

<sup>11</sup><https://sobigdata.d4science.org/group/tagme/>

<sup>12</sup><https://dumps.wikimedia.org/>

<sup>13</sup><https://github.com/thunlp/OpenHowNet>

<sup>14</sup><https://github.com/thunlp/DictSKB>

### 3.4.2 Model Construction

In the modeling part, we use the state-of-the-art graph-based STC model SHINE [51]. SHINE constructs word-level, POS-level, and entity-level graphs (NELL [68]) denote as  $\mathcal{G}_w$ ,  $\mathcal{G}_p$ , and  $\mathcal{G}_e$ . We replace the entity graph  $\mathcal{G}_e$  with sememe graph  $\mathcal{G}_s$ .

Specifically,  $\mathcal{G} = \{\mathcal{V}, \mathbf{A}\}$  where  $\mathcal{V}$  is a set of nodes and  $\mathbf{A} \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$  is the adjacency matrix. Each node  $v^i \in \mathcal{V}$ . Define the feature matrix of  $\mathcal{V}$  as  $\mathbf{X} \in \mathbb{R}^{|\mathcal{V}| \times dim}$ , then the feature of  $v^i$  as  $\mathbf{x}^i \in \mathbb{R}^{dim}$ .

- Construct  $\mathcal{G}_w$ :  $[\mathbf{A}_w]_{ij} = \max(\text{PMI}(v_w^i, v_w^j), 0)$  PMI denotes the point-wise mutual information between words [49].  $\mathbf{x}_w^i \in \mathbb{R}^{|\mathcal{V}_w| + word\_dim}$  is concat by one-hot encoding and Glove word embedding. Define the word to short text adjacency matrix as  $[\mathbf{A}_{w2st}]_{ij} = \text{TF-IDF}(v_w^i, v_{st}^j)$ , where  $\mathbf{A}_{w2st} \in \mathbb{R}^{|\mathcal{V}_w| \times |\mathcal{V}_{st}|}$  and  $v_{st}^j \in \mathcal{V}_{st}$  denotes each short text node.
- Construct  $\mathcal{G}_p$ : Use NLTK<sup>15</sup> to convert word nodes to POS tag nodes, and set  $[\mathbf{A}_p]_{ij} = \max(\text{PMI}(v_p^i, v_p^j), 0)$ .  $\mathbf{x}_p^i \in \mathbb{R}^{|\mathcal{V}_p|}$  is a one-hot vector. Set the POS tag to short text adjacency matrix as  $[\mathbf{A}_{p2st}]_{ij} = \text{TF-IDF}(v_p^i, v_{st}^j)$ , where  $\mathbf{A}_{p2st} \in \mathbb{R}^{|\mathcal{V}_p| \times |\mathcal{V}_{st}|}$ .
- Construct  $\mathcal{G}_s$ : We use sememe as entity nodes, which means that the number of entity nodes is fixed no matter the dataset. In constructing the edge between the  $\mathcal{V}_{st}$  and the  $\mathcal{V}_s$ , we consider the senses of the word. Specifically, in Figure 3.10.

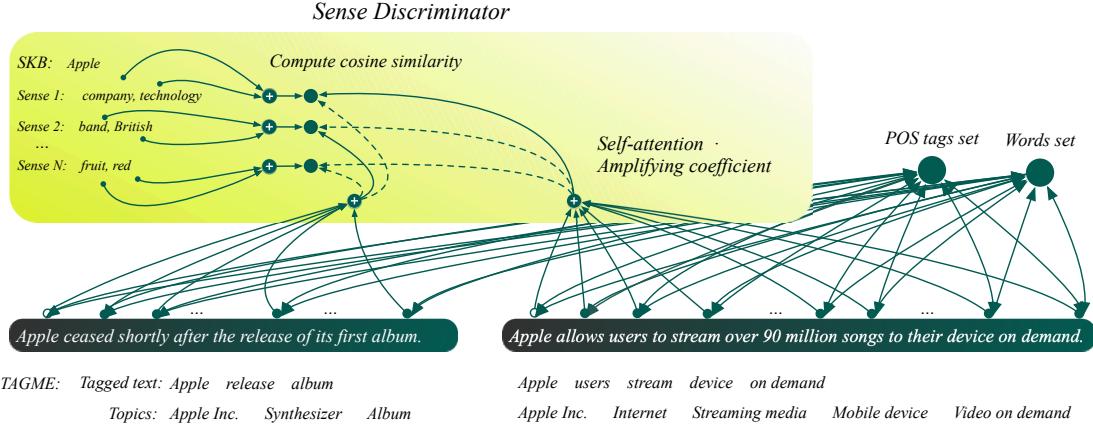
Our approach is first to compute the embedding representation of the word in the current text using a self-attentive mechanism, as

$$\mathbf{x}_{local\_word}^i = \text{SOFTMAX} \left( \mathbf{x}_w^i \cdot [\mathbf{X}_w]_t^\top \times e \right) \cdot [\mathbf{X}_w]_t, \quad (3.10)$$

where  $[\mathbf{X}_w]_t \in \mathbb{R}^{word\_dim \times |\mathcal{V}_{st}^t|}$  means the word embedding matrix in  $\mathcal{V}_{st}^t$ ,  $\mathcal{V}_{st}^t \in \mathcal{V}_{st}$  means the  $t$ th sort text. Since sentence length affects the bias of words in a sentence, we cite the method of [54], where an amplifying coefficient  $e \in \mathbb{R}$ ; and  $e > 0$  is

---

<sup>15</sup><https://www.nltk.org/api/nltk.tag.html>



**Figure 3.10:** Construction of sememe-based edges. We utilize the sense discriminator to select the correct sense. “Apple” is an entity in both sentences, the left side indicates the band, and the right side indicates the company; however, it is not well differentiated for TagMe. Note that the dotted line indicates that this sense will be discarded.

multiplied after Softmax to enhance the bias ratio of similar semantic words, we set  $e = 4$ .

Then, use the same method to calculate each sense vector

$$\mathbf{x}_{local\_sense}^{ik} = \text{SOFTMAX} \left( \mathbf{x}_w^i \cdot [\mathbf{X}_s]^T_k \times e \right) \cdot [\mathbf{X}_s]_k, \quad (3.11)$$

and returns the sememe nodes as  $\mathcal{V}_s^i$  for the sense most relevant to the  $i$ th word.

$$\begin{aligned} \mathcal{V}_s^i = \max & \left( \text{Cos}(\mathbf{x}_{local\_word}^i, \mathbf{x}_{local\_sense}^{i1}), \right. \\ & \left. \dots, \text{Cos}(\mathbf{x}_{local\_word}^i, \mathbf{x}_{local\_sense}^{ik}) \right). \end{aligned} \quad (3.12)$$

Next, we define the adjacency matrix of text nodes to sememe nodes as

$$[A_{s2st}]_j = \left( (\mathcal{E}_{w2sen}^1 \odot \mathcal{E}_{sen2sem}) \cup, \right. \\ \left. \dots, \cup (\mathcal{E}_{w2sen}^i \odot \mathcal{E}_{sen2sem}) \right)^T, \quad (3.13)$$

where

$$\mathcal{E}_{w2sen}^i = \text{Cos} \left( \mathbf{x}_{local\_word}^i, \mathbf{x}_{local\_sense}^{ik} \right), \quad (3.14)$$

and

$$\mathcal{E}_{sen2sem}^i = \text{Cos}(\mathbf{x}_w^i, [\mathbf{X}_s]^i_k), \quad (3.15)$$

$i$  means the number of words in the current short text. In order to include more words, and  $[A_{s2st}]_j$  denotes the relationship of each  $\mathcal{V}_{st}$  to  $\mathcal{V}_s$ . We use sense nodes as weight parameters to help learn the weights of words to sememe. We use the public GloVe word embeddings<sup>16</sup> as the original vector of words with sememes. In the representation of the relationship between sememes,

$$[\mathbf{A}_s]_{ij} = \max(\text{Cos}(\mathbf{x}_s^i, \mathbf{x}_s^j), 0), \quad (3.16)$$

$\mathbf{x}_s^i, \mathbf{x}_s^j \in \mathbf{X}_s$ ; and  $\mathbf{X}_s \in \mathbb{R}^{m \times \text{word\_dim}}$ , we tried to use the Glove word embedding and the TransE method [69]. For the details of TransE, we used the open source code<sup>17</sup> and set the number of training times to 1,000 and the dimension to 300. the result is that Glove word embedding performs better.

The learning node embedding part is consistent with SHINE, composed of two layers of a graph convolutional network (GCN) [46].

### 3.4.3 Experiments

All experiments were conducted on an NVIDIA Tesla K80 GPU and AMD Ryzen 7 2700 Processor. We perform experiments on Snippets<sup>18</sup>. The Snippets is a dataset of web search snippets returned by Google Search [70]. It contains 12,340 texts, with an average token count of 14.5 per text, divided into eight categories.

In setting hyperparameters, we set the sliding window size of PMI as 5, set the threshold as 2.7, the learning rate as  $5^{-3}$ , the dropout rate is set as 0.5, and train the model 1000 epochs.

We find that the average number of sememe for senses significantly impacts the

---

<sup>16</sup><https://nlp.stanford.edu/data/glove.840B.300d.zip>

<sup>17</sup><https://github.com/thunlp/OpenKE>

<sup>18</sup><http://acube.di.unipi.it/tmn-dataset/>

model, like in Figure 3.11. Note that we set an upper bound  $\delta \in \mathbb{N}^* \leq 10$  on the number of sememe per sense. The method calculates the cosine similarity between the current word and the sememes and returns the  $\delta$  largest sememes.

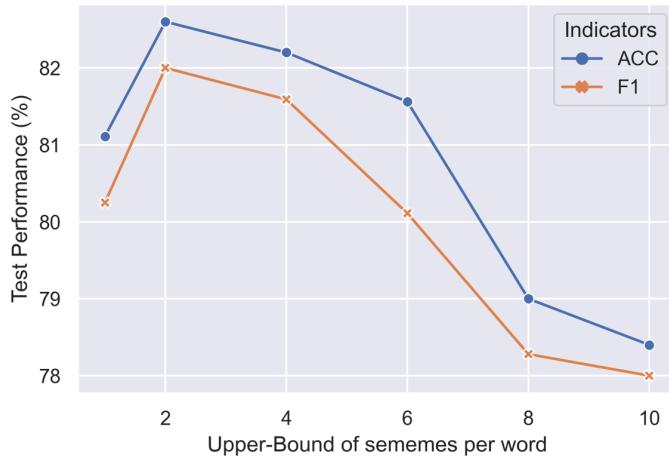


Figure 3.11: Analyze the effect of the average sememe number for senses on the model.

Following [51] and [71], we randomly sampled 40 labeled short texts from each category. Half of them form the training set, and the other half form the validation set for hyperparameter tuning. Experimentally, our method performs better than the baseline in both accuracy (ACC) and F1 score (In Table 3.6).

## 3.5 Conclusion

This work has pioneered an attempt to use SKB as an entity network applied to short text classification and has good performance. Surprisingly, lowering the average sememe number of senses significantly improves the performance of the model, probably due to the redundancy of our SKB construction. In addition, we believe that this does not fully exploit the performance of SKBs. We will continue this work to optimize the graph net-

Model	Snippets	
	ACC	F1
TFIDF+SVM	64.70	59.17
LDA+SVM	62.54	56.40
PTE	63.10	58.96
BERT-avg	79.31	78.47
BERT-CLS	81.53	79.03
CNN-rand	48.34	42.12
CNN-pre	77.09	69.28
LSTM-rand	30.74	25.04
LSTM-pre	75.07	67.31
TLGNN	70.25	63.18
TextING	71.13	70.71
HyperGAT	70.89	63.42
TextGCN	77.82	71.95
TensorGCN	74.38	73.96
STCKA	68.96	61.27
HGAT	82.36	74.44
STGCN	70.01	69.93
SHINE (Tesla v100)	82.39	81.62
SHINE (CPU)	<u>82.03</u>	<u>81.92</u>
<b>SHINE+SKB (CPU)</b>	<b>82.60</b>	<b>82.00</b>

Table 3.6: Test performance (%) measured on Snippets dataset. Our model is shown in **bold**, and it boosts the ACC and F1 score of the current best performing model (marked in underline) by 0.57 and 0.08 percentage points, respectively.

work for SKBs and try to employ SKBs on more graph neural networks in the future.

In the data augmentation section, we utilize a more straightforward method of extracting SKBs from word definitions instead of deep clustering. We extend the original SKB dictionary to allow for broader application to downstream tasks.

## 4. Sentence Representation

We found that graph-based text encoding methods tend to ignore the sequential structure (syntactic features) of the text. So we further discuss how to add heterogeneous information to the Sequence to Sequence (Seq2Seq) model, thereby keeping the syntactic features. Moreover, we demonstrate the effectiveness of our method through extensive experiments.

The difficulty of sentence representation learning is how to express the semantic information of the text effectively. Especially in question-answering systems, understanding the semantics of a sentence is often more important than what topic the sentence belongs to. History tells us that deep learning can accomplish artificial intelligence tasks that require highly abstract features such as Figure 4.1.

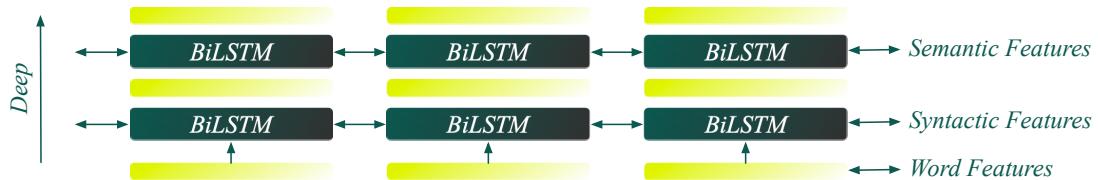


Figure 4.1: Learning Hierarchical Representations

This chapter will introduce two essential language models for learning sentence representation: LSTM and BERT in section 4.1.1, which are introduced in the chapter on word representation learning, and which can also represent sentence information by changing slightly. Their shortcomings are explored in section 4.2, and our approach to addressing these problems is presented in section 4.3. Section 4.3 also presents our proposed paper-finding tool, which helps researchers quickly find the needed papers.

## 4.1 Related Work

### 4.1.1 Bidirectional Long Short-Term Memory Networks

LSTMs or RNNs have many application scenarios. Several design approaches for them are shown in Figure 4.2 (refer to Dr. Andrej Karpathy's blog<sup>1</sup>), and this figure shows five different architectures:

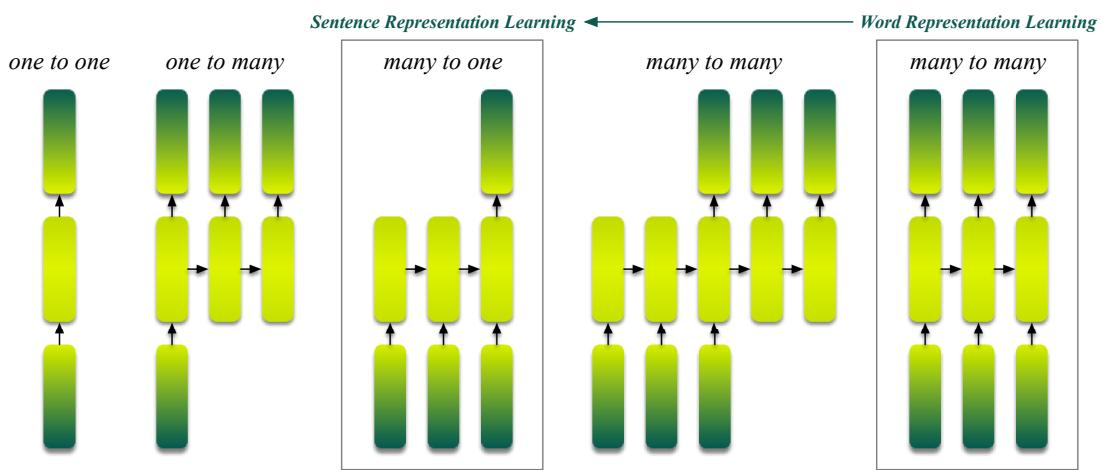


Figure 4.2: Encoding Types of RNNs or LSTMs. Each rectangle is a vector and arrows represent functions (e.g. matrix multiply)

- **One to One:** The first architecture is not a temporal model and does not contain any temporal information. It is suitable for tasks like image classification.
- **One to Many:** This subillustration has one input at the moment  $t = 1$  but one output at each moment. This architecture is suitable for tasks like image captioning. In other words, it is given a picture to generate the corresponding text.
- **Many to One:** This illustration has an input at every moment but an output only at the last moment. It is suitable for tasks like sentiment classification.

---

<sup>1</sup><http://karpathy.github.io/2015/05/21/rnn-effectiveness/>

- **Many to Many:** Typical machine translation models
- **Many to Many:** The language model we introduced before. It can also be used for Named-entity recognition tasks.

Since RNNs suffer from vanishing gradients, which causes them not to capture the previous information and is caused by long series of multiplications of small values, diminishing the gradients and causing the learning process to degenerate [72]. In contrast, LSTMs use units called cells in each hidden layer instead of multiplication operations. The long term dependencies and relations are encoded in the cell state vectors and it is the cell state derivative that can prevent the LSTM gradients from vanishing. BiLSTMs compensate for these drawbacks. A Bidirectional LSTM, or BiLSTM consists of two LSTMs: one taking the input in a forward direction, and the other in a backwards direction. Moreover, A Bidirectional LSTM, or BiLSTM, consists of two LSTMs. One taking the input in a forward direction, and the other in a backwards direction, which effectively increases the amount of information available to the network and improves the context available to the algorithm. We will introduce the BiLSTM model in detail in Section 4.3 and employ it as a baseline model to construct BiLSTM models with embedded heterogeneous information.

### 4.1.2 Bidirectional Encoder Representations from Transformers

Coincidentally, there are also many different constructs for Bidirectional Encoder Representations from Transformers (BERTs), which can be adapted to 11 downstream tasks of natural language processing, and they can be roughly grouped into four main categories. These four types of tasks are: sentence pair classification tasks, single-sentence classification tasks, question answer tasks, and single-sentence tagging tasks, and the modifications to the network for the four tasks are shown in the following figure 4.3.

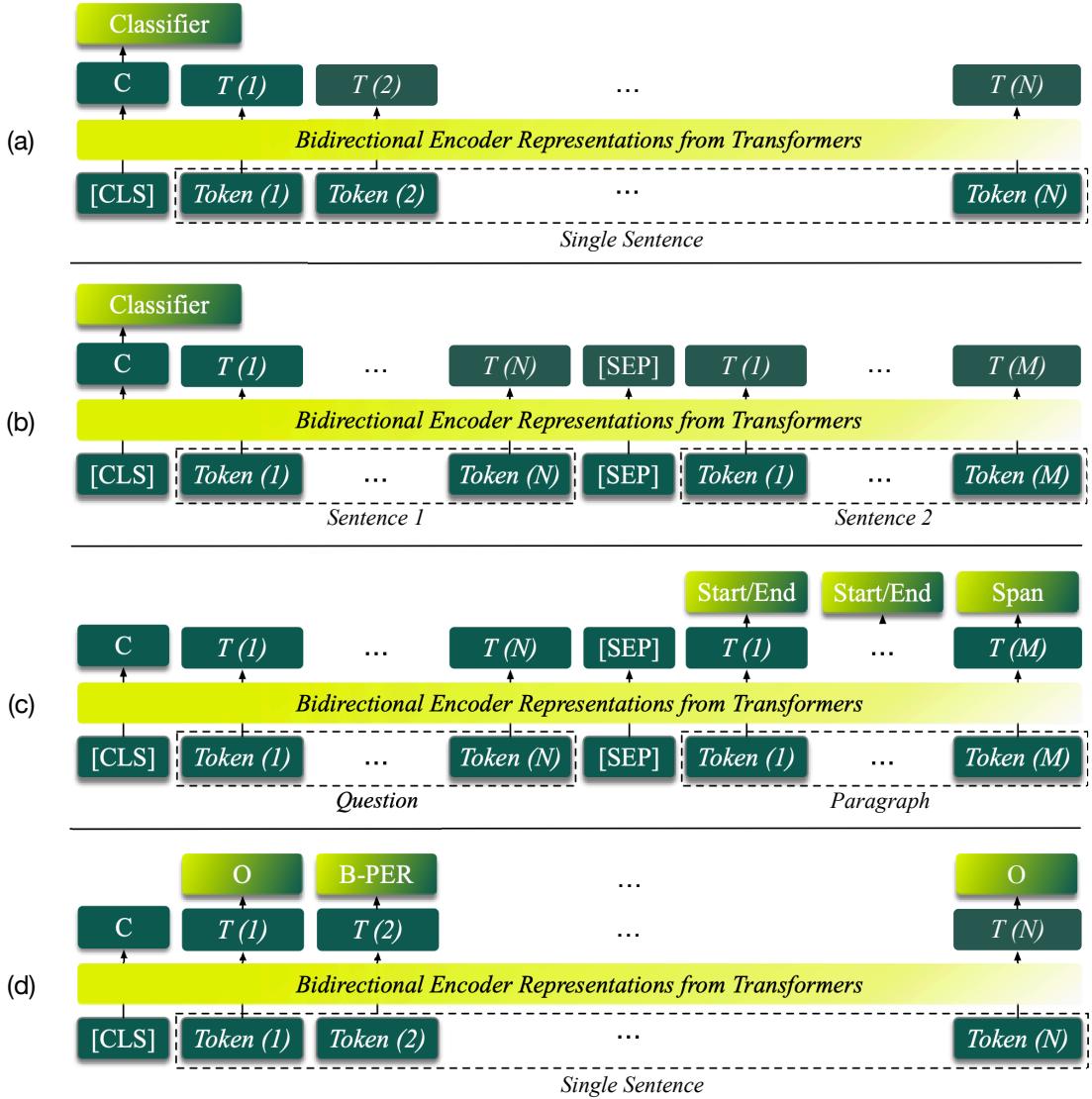


Figure 4.3: Encoding Types of BERTs. The subillustrations (a), (b), (c) and (d) represent Sentence Pair Classification Tasks, Single Sentence Classification Tasks, Question Answer Tasks, and Single-sentence Tagging Tasks, respectively. Note, the final hidden vector of the special [CLS] token as  $C$ , and the final hidden vector for the  $i^{th}$  input token as  $T(i)$

### ■ Sentence Pair Classification Tasks:

1. Sentence pair classification tasks are sequence-level tasks (for each sequence, only the loss of one output in the Bert model is calculated), and solving the sentence pair classification task using the Bert model requires the following adjustments to the Bert model.
2. A classification layer (fully connected layer + softmax layer) is added after the output corresponding to the [CLS] position to calculate the final classification probability of the output.

### ■ Single Sentence Classification Tasks:

1. Similarly, a classification layer is added after the [CLS] output to calculate the classification probability.

### ■ Question Answer Tasks:

1. A classification layer (fully connected layer + softmax layer) is added after the output positions of all Bert answer tokens to output the probability that each position is the beginning and end of the answer. Finally, the mean loss of the start and end losses is calculated.

### ■ Single-sentence Tagging Tasks:

1. A classification layer (fully connected layer + softmax layer) is added after all Bert outputs for outputting the probabilities of the final labeled categories.

We extensively utilize BERT-based pre-trained language models for generating datasets on the Paper Recommendation System task in Section 4.3.

## 4.2 The Completeness of Sentence Representations

The traditional approach of fine-tuning downstream tasks using pre-trained language models ignores some heterogeneous features in sentences because they are difficult to distinguish in the same vector representation. We suppose these heterogeneous features should be learned separately by the language model and combined to improve the overall sentence expression (like Figure 4.4).

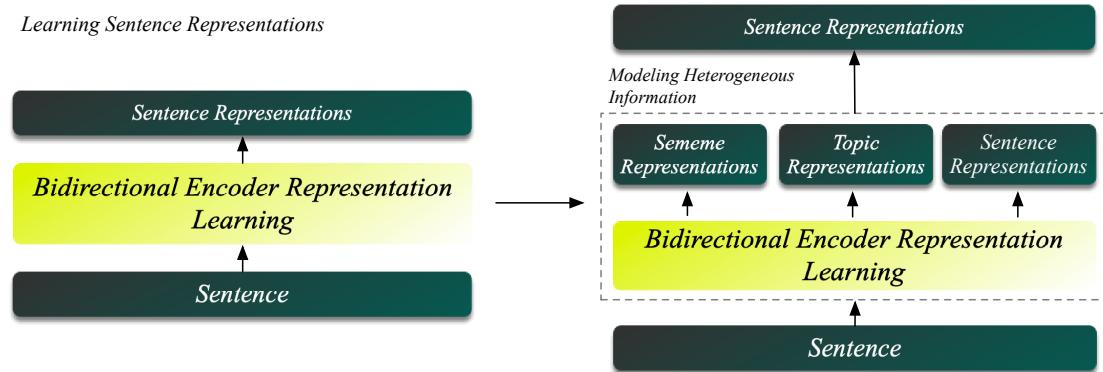


Figure 4.4: Sentence Representations Learning with Heterogeneous Information

In the next section, we propose a method for question generation based on unsupervised multi-hop question answering to adapt to the beginner’s questioning style, using NLP papers as an example. Specifically, we employ a keyphrase extraction pre-training model to generate questions on the pre-trained Q&A model utilizing the keyphrases as answers. Next, construct an extensive keyphrase dictionary by converting questions into descriptions operating linguistic rules. Finally, we augment the questions by replacing the keyphrase in the questions with their definitions. Moreover, to implement the paper recommendation task, we constructed a simple neural network recommendation model to predict the order of paper relevance. We collected paper abstracts from NLP’s top venues published from 2017 to 2022 for training. In extensive experiments, our method shows

comparable performance.

Furthermore, Section 4.3.2 uses the paper recommendation model proposed in this thesis as the baseline model and adds more heterogeneous information (e.g., author, title, sememe) to the original one. After experiments, our model has been further improved and substantially outperforms the baseline model in accuracy.



### 4.3 NLP Paper Recommender: MIYU

Researchers quickly find and understand the articles relevant to their research remains challenging due to the rapid iteration of technologies and the ever-increasing volume of scientific articles. In this work, we propose a method for question generation based on unsupervised multi-hop question answering to adapt to the beginner’s questioning style, using Natural Language Processing papers as an example. Specifically, we employ a keyphrase extraction pre-training model to generate questions on the pre-trained Q&A model utilizing the keyphrases as answers. Next, construct an extensive keyphrase dictionary by converting questions into descriptions operating linguistic rules. Finally, we augment the questions by replacing the keyphrase in the questions with their definitions (the detail in Subsection 4.3.1). Moreover, to implement the paper recommendation task, we constructed a simple neural network recommendation model, namely MIYU (みゆ) in Figure 4.5, to predict the order of paper relevance, which incorporates the title, author, and sememe information of the paper. We collected paper abstracts from top venues of NLP published from 2017 to 2022 for training. In extensive experiments, our method shows comparable performance. (the detail in Subsection 4.3.2).

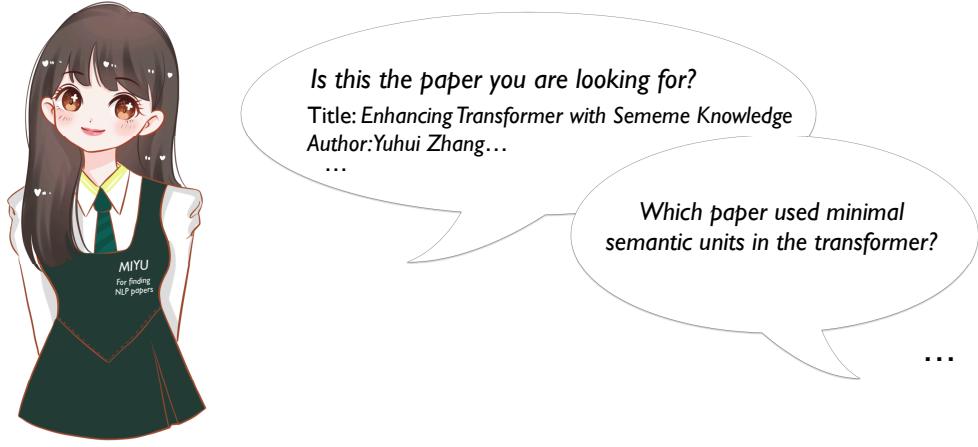


Figure 4.5: NLP Paper Recommender: MIYU

### 4.3.1 Finding NLP Papers by Asking a Multi-hop Question

In recent years, NLP has seen rapid growth with the impact of deep learning, then many researchers have flocked to this research boom. However, due to the massive increase in scientific papers per year, it is challenging for NLP researchers (NLPers) to find papers that are helpful to them quickly. This has led to numerous duplicate studies (such as reinventing the wheel) syndrome. Therefore, many scholars believe paper reading is crucial for NLPers and even more significant than coding ability. Moreover, papers often incorporate multiple terminologies. Specifically, when researchers enter a new field, they cannot search the relevant papers precisely based on terminologies. Nevertheless, this issue has not been considered in the extant studies [73].

A more intuitive solution is to convert the academic style text in the paper into informal text that non-specialists can understand and then ask questions from the informal text style paper to improve the search efficiency. Regarding this aspect, the Text Style Transfer (TST) [74] task is better-known among the related studies, while the layman style transfer is a subtask of TST, also commonly dubbed the Simplicity (*Complicated → Simple*) task.

One of the successful efforts has been to employ medical data to perform text conversions between expert and laymen styles, namely Expertise Style Transfer (EST) [75]. This work employs the terminology definitions from the health reference Merck Manuals as a bridge to EST. However, numerous already constructed artificial lexicons are unlikely to exist in the emerging field, making EST extremely challenging.

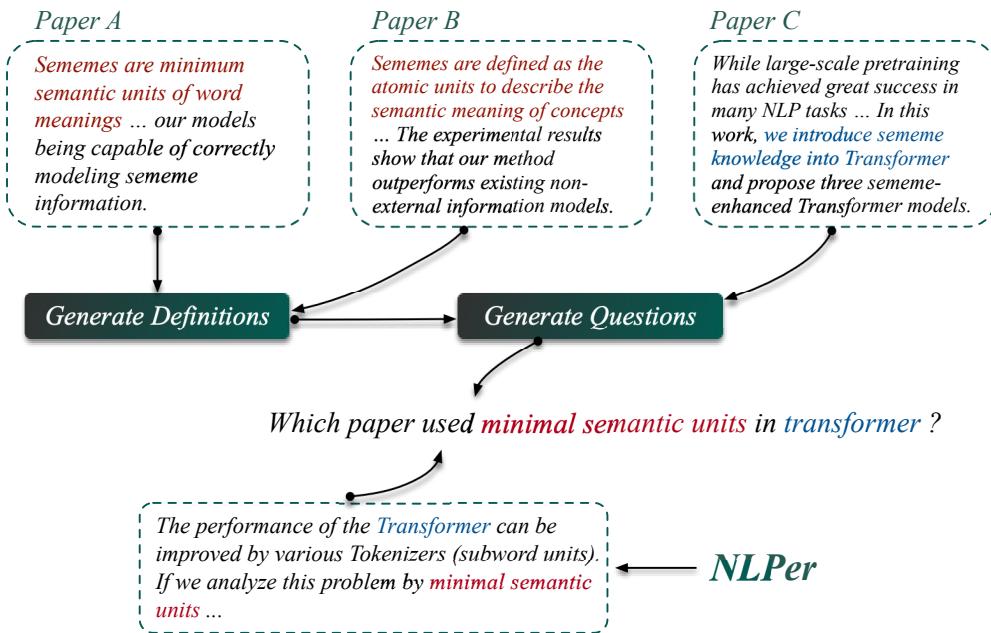


Figure 4.6: Automatic generation of terminology dictionary-based questions from unstructured text.

This paper proposes a method to extract terminology definitions from papers to solve the terminology dictionary problem (Figure 1). Our motivation is that since the term is defined in the paper, it is most straightforward to extract the definition of the term from the paper. Specifically, we employ an automatic keyphrase extraction [76] model to extract keyphrases as terms and use the pre-trained language model [77] fine-tuned on the Q&A dataset to generate questions utilizing the terms as answers automatically. Following this, we constructed an extensive keyphrase

dictionary as a terminology dictionary by converting questions into descriptions operating linguistic rules [78].

Moreover, instead of rewriting sentences in papers to address the terminology problem in paper searches, we opted to rewrite the questioner’s question via building a multi-hop question. Specifically, we treat the relationship extraction task as an answer extraction task, ask questions about papers using manually defined entity relationships and minister their answers as entities associated with the question type. Finally, we employed an unsupervised multi-hop question answering technique [79] to rewrite the questions by treating terminologies as bridge entities.

Furthermore, to implement the paper recommendation task, we constructed a Bidirectional Long Short Term Memory (BiLSTM) [43] network to predict the order of paper relevance. Moreover, we provide the inference API of the model on the Hugging Face<sup>2</sup>.

Our main contributions can be summarized as:

- We propose a method to extract terminology definitions from papers to solve the terminology dictionary problem.
- We employed an unsupervised multi-hop question answering technique to rewrite the questions by treating terminologies as bridge entities. Afterward, we constructed a sizeable single-round Q&A dataset for paper retrieval.
- We constructed a BiLSTM network to predict the order of paper relevance.

In the following subsections will divide into three parts: terminology dictionary construction (subsection 4.3.1), question generation (subsection 4.3.1), and paper prediction model (subsection 4.3.1).

## Terminology Dictionary Construction

We employ the method of unsupervised problem generation [79] as the basic framework. Subsequently, adding the terminology extraction module and terminology normalization

---

<sup>2</sup>[https://huggingface.co/SauronLee/BiLSTM\\_Finding\\_NLP\\_Papers](https://huggingface.co/SauronLee/BiLSTM_Finding_NLP_Papers)

module. Moreover, augment the data by employing multi-pretrained models.

Specifically, as in Figure 4.7, we used two pre-trained models to extract terms. First, We use the pretrained Google T5 model [80] fine-tuned on SQuAD [81] to automatically generate questions & answers and use the answers as terms.<sup>3</sup> Secondly, We use Keyphrase Boundary Infilling with Replacement (KBIR) [82] as its base model<sup>4</sup> and fine-tune it on the Inspec dataset [83] to extract keyphrases as terms.

To avoid including common words or terms with repeated meanings in the term dictionary, we removed words that appeared on Wikipedia with a word frequency<sup>5</sup> greater than 2,000. Then, we solve problems like “2Seq” (*Seq2Seq*) by replacing incomplete words according to the terminology dictionary.

In the question generation part, we use the T5 model fine-tuned on SQuAD [84] to generate questions based on the terms as answers. Finally, the questions are converted into declarative sentences using straightforward word replacement [78] and fill mask based on XLM-RoBERTa [85] base-sized language model<sup>6</sup>.

## Question Generation

This subsection will introduce the question design and slot filling.

We manually defined the relationship structure of the entities in the paper and designed the one-hop and multi-hop questions based on the entity relationships. Specifically, We annotated and took notes<sup>7</sup> on numerous NLP papers through the entity annotation tool

---

<sup>3</sup>[https://github.com/teacherpeterpan/Unsupervised-Multi-hop-QA/blob/main/MQA\\_QG/Operators/T5\\_QG.py](https://github.com/teacherpeterpan/Unsupervised-Multi-hop-QA/blob/main/MQA_QG/Operators/T5_QG.py)

<sup>4</sup><https://huggingface.co/ml6team/keyphrase-extraction-kbir-inspec>

<sup>5</sup><https://github.com/IlyaSemenov/wikipedia-word-frequency/blob/master/results/enwiki-20190320-words-frequency.txt>

<sup>6</sup><https://huggingface.co/xlm-roberta-base>

<sup>7</sup><https://github.com/SauronLee/Paper-KG>

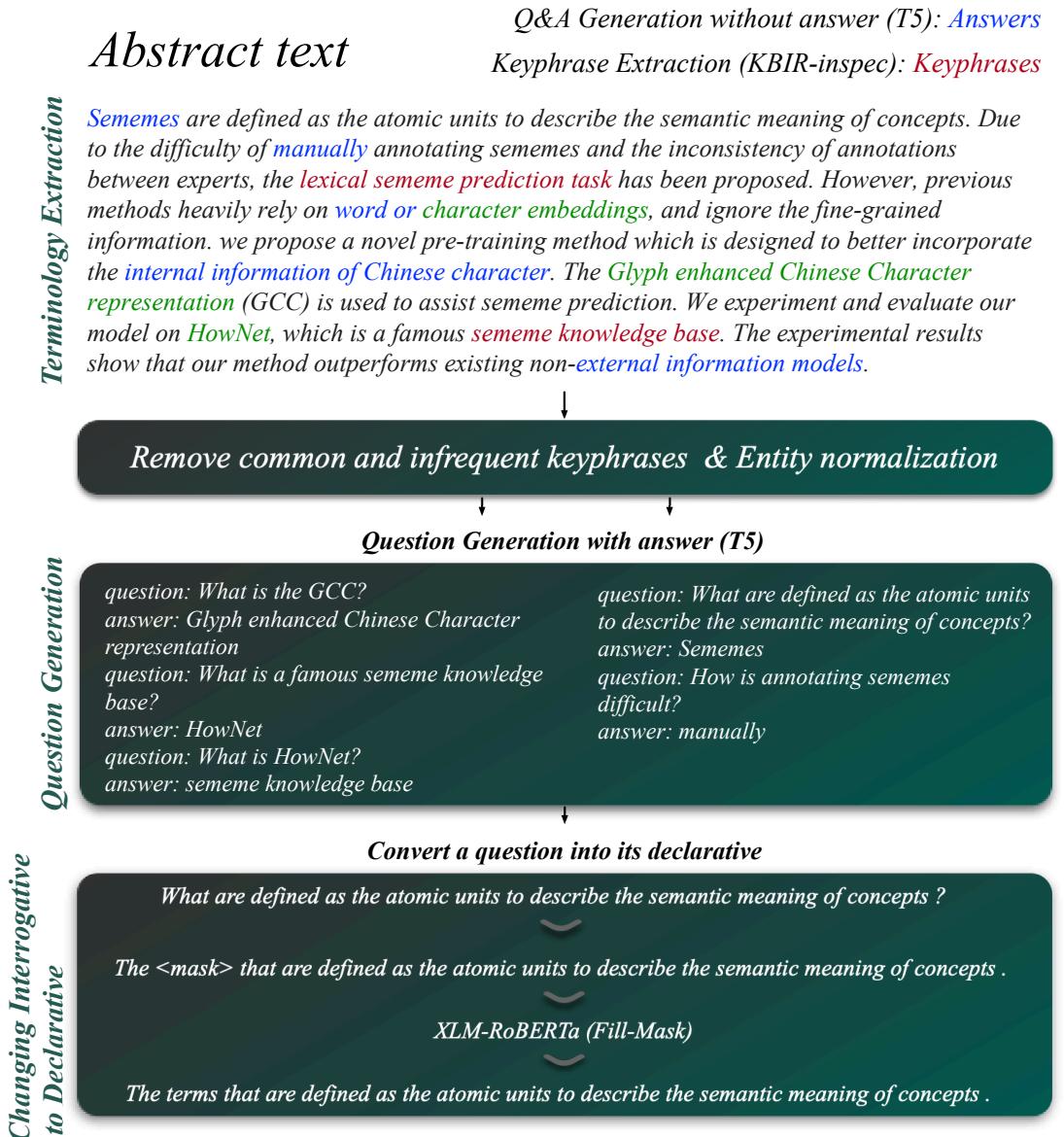


Figure 4.7: The pipeline of terminology dictionary construction: In Terminology Extraction and Question Generation, The blue font represents terminologies extracted by the Q&A generation, the red font represents the keyphrase extraction results, and the green font represents the shared terminologies. Finally, the questions are converted into declarative sentences using straightforward word replacement and fill mask.

Brat<sup>8</sup> [86]. We summarized eight types (like Figure 4.8) of entities with 19 entity relationships (like Figure 4.9).

<i>Entities definition</i>	
<i>Field</i>	<i>Defined as a broad concept.</i>
<i>Theorie</i>	<i>A theory can also be a conclusion. It may have been validated or may not have been validated yet.</i>
<i>Method</i>	<i>A defined method or model.</i>
<i>Language</i>	<i>It generally refers to the language on which the dataset or examination results are based.</i>
<i>Issue</i>	<i>An unresolved issue.</i>
<i>Task</i>	<i>(1) A new task requires a new evaluation dataset or measure. (2) Near the beginning of the paragraph, followed by the challenge.</i>
<i>Dataset</i>	<i>Datasets or corpora.</i>
<i>Achievement</i>	<i>Metrics for evaluation based on some benchmarks.</i>

Figure 4.8: One-hop and multi-hop questions building: Entity Declaration

Then, we defined 17 possible questioning styles for the questioner based on entity relations. (including eight types of one-hop questions and nine types of multi-hop questions). Note that in the question design part, we just concatenated “*which paper*” and “*relationship*” or “*entity*” as shown in Figure 4.10.

It may not follow the grammatical structure; therefore, we have employed the T5 model<sup>9</sup> finetuned on the JFLEG dataset [87] to make Grammatical Error Correction (GEC) on the spliced sentences.

In the slot filling part, we designed questions based on the entity definition (shown in Table 4.1) and using the XLM-RoBERTa base-sized language model<sup>10</sup> fine-tuned on SQuAD 2.0 [88] to ask these questions about the paper and thus obtain the corresponding entities  $\mathcal{E}$ . Then we filter out entities that do not appear in the terminology dictionary  $\mathcal{T}$  (such as  $\mathcal{E} = \mathcal{E} \cap \mathcal{T}$ ).

Moreover, to reduce the terminology in the question, we use term definitions to re-

---

<sup>8</sup><https://brat.nlplab.org/>

<sup>9</sup><https://huggingface.co/vennify/t5-base-grammar-correction>

<sup>10</sup><https://huggingface.co/deepset/xlm-roberta-base-squad2>

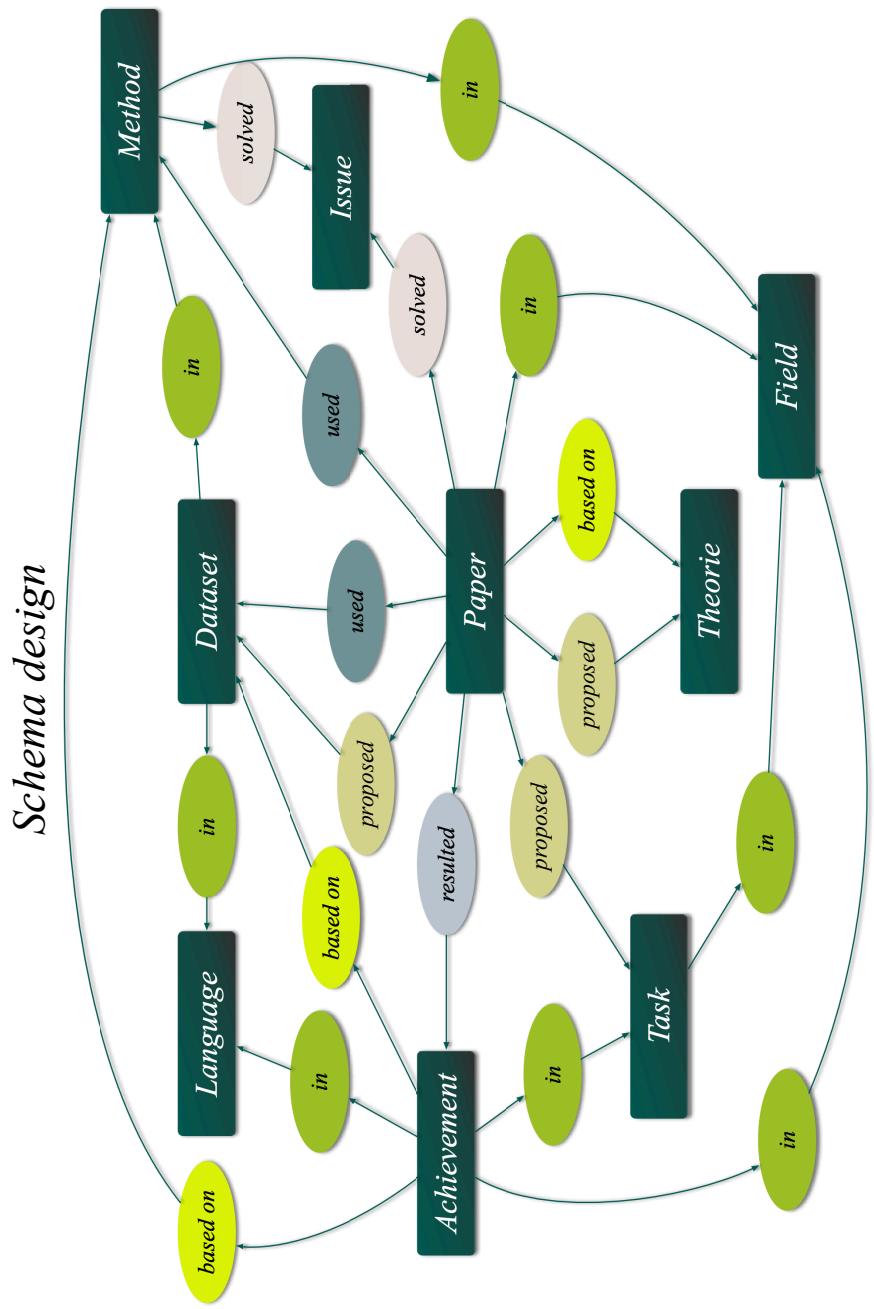


Figure 4.9: One-hop and multi-hop questions building: Schema. Where rectangles represent entities and ellipses represent entity relationships.

Entity	Question
Theorie	What theories are presented In this thesis? What theories are this paper based on?
Field	What field does this paper belong to?
Language	What language is the data for this paper based on? What languages are used In this thesis?
Issue	What problem does this article solve? What is the current unresolved issue?
Method	What methods are used In this thesis? What technologies are used in this article? What new methods are proposed In this thesis?
Task	What task does this article belong to? What tasks are used In this thesis? What new tasks are proposed In this thesis?
Dataset	Which datasets are used In this thesis? What new datasets are presented In this thesis?
Result	What are the results of this paper? What are the conclusions of this paper?

Table 4.1: Building questions based on the entity definition for entity extraction.

## Questions design

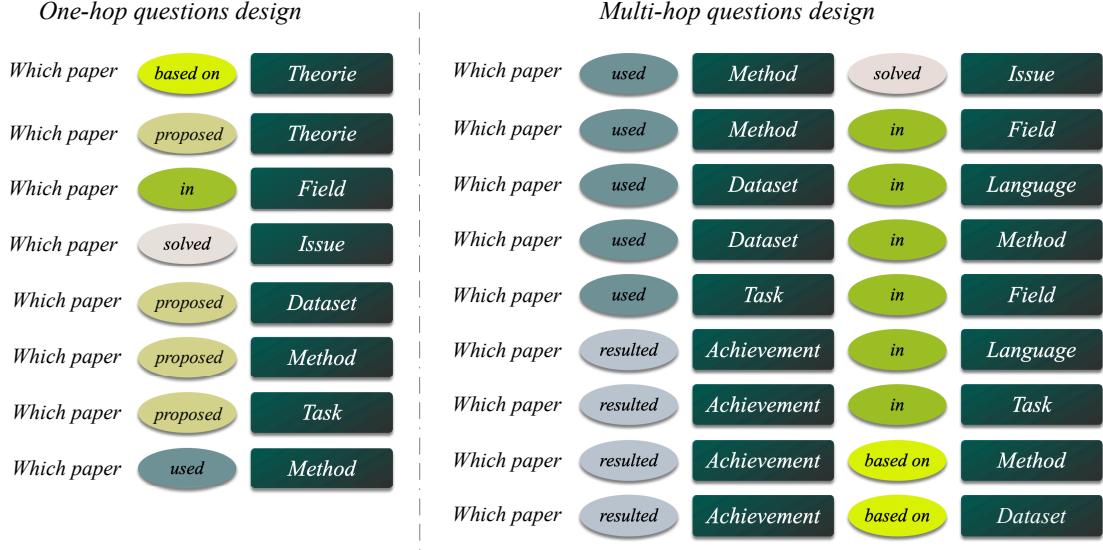


Figure 4.10: One-hop and multi-hop questions building: Template Definition. We defined 17 possible questioning styles for the questioner based on the schema.

place the related terms. Primarily, we calculated the cosine similarity between the slot type and the “*mask word*” in the term definition. We excluded the term definitions with a similarity of less than 0.8 due to the existence of renaming of different types of terms or inconsistency between the term definition and the slot definition.

## Paper Prediction Model

In this subsection, we are motivated by the reverse dictionary model [89]. We first encode questions and papers, i.e., questions and answers, into vectors and then utilize a superficial linear layer to map the question vector to the paper vectors space and update the network by computing the cross entropy loss. Finally, the papers are ranked by similarity (Figure 4.14).

Regarding the coded paper vector, we constructed a bipartite graph based on the rela-

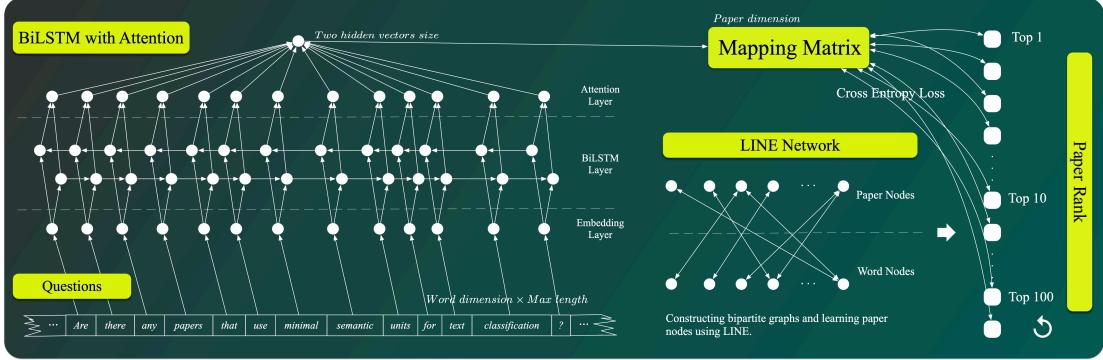


Figure 4.11: BiLSTM Mapping Flowchart: The whole network architecture consists of three parts: sentence representation learning, paper representation learning, and paper prediction.

tionship between the words that appear in the paper and the paper itself. Then we employ the LINE graph embedding model [57] to train the node vectors and employ the second-order similarity as paper vectors.

To encode the problem vector, we used Bidirectional Long Short Term Memory (BiLSTM) [43] as the base network. Formally, for an input question  $\mathcal{Q} = \{q_1, \dots, q_{|\mathcal{Q}|}\}$ , we replace the words in the question with word embeddings and pass them to BiLSTM. The output of the model is a question vector after the attention mechanism.

$$v_{question} = \text{Attention}(\text{Encode}_{BiLSTM}(\mathcal{Q})), \quad (4.1)$$

Where the outputs of  $\text{Encode}_{BiLSTM}(\mathcal{Q})$  are two sequences of bi-directional hidden vectors

$$\{\overrightarrow{h}_1, \dots, \overrightarrow{h}_{|\mathcal{Q}|}\}, \{\overleftarrow{h}_1, \dots, \overleftarrow{h}_{|\mathcal{Q}|}\},$$

where  $\overrightarrow{h}_i, \overleftarrow{h}_i \in \mathbb{R}^h$  denote the hidden layer states passed forward and backward,  $h$  is the dimension of directional hidden states. Thus the formula can be rewritten as:

$$v_{question} = \sum_{i=1}^{|\mathcal{Q}|} \left( C(\overrightarrow{h}_{|\mathcal{Q}|}, \overleftarrow{h}_1) \cdot C(\overrightarrow{h}_i, \overleftarrow{h}_i) \right) \times C(\overrightarrow{h}_i, \overleftarrow{h}_i), \quad (4.2)$$

where  $C(\cdot, \cdot)$  is a simple concatenate function. Then we map the  $v_{question}$  to the paper vectors space by the following equation:

$$v_{paper} = \mathbf{W}_{paper} v_{question} + b, \quad (4.3)$$

where  $\mathbf{W}_{paper} \in \mathbb{R}^{p \times 2h}$  is a weight matrix,  $b \in \mathbb{R}^p$  is a bias vector. where  $p$  is the dimension of paper vectors as  $v_{paper_{LINE}}$  from the LINE model. Finally, calculate the confidence score of each paper using the dot product:

$$score_{base} = v_{paper} \cdot v_{paper_{LINE}}, \quad (4.4)$$

### 4.3.2 Improving Finding NLP Papers via Heterogenous Information

In this subsection, we introduce heterogeneous information to improve the expressiveness of sentences in specific downstream tasks (Note that this study only focuses on paper recommendation system design). Compared to the previous section, we collect more information about the paper, such as the title, author and sememe, and use this information to improve the recommendation capability. Our motivation comes from the reverse dictionary model. The input to the model requires a question from the finder. The model simultaneously predicts three heterogeneous features (title, author, and sememe) of the paper based on the question and finds the most relevant paper based on each feature. Finally, all the heterogeneous information is combined with the correlation coefficients of the papers, and the papers are ranked such as 4.12. We present the modeling of title, author and sememe information in subsubsection 4.3.2.1, subsubsection 4.3.2.2 and subsubsection 4.3.2.3, respectively, and present the experiments and results in subsection 4.3.3.

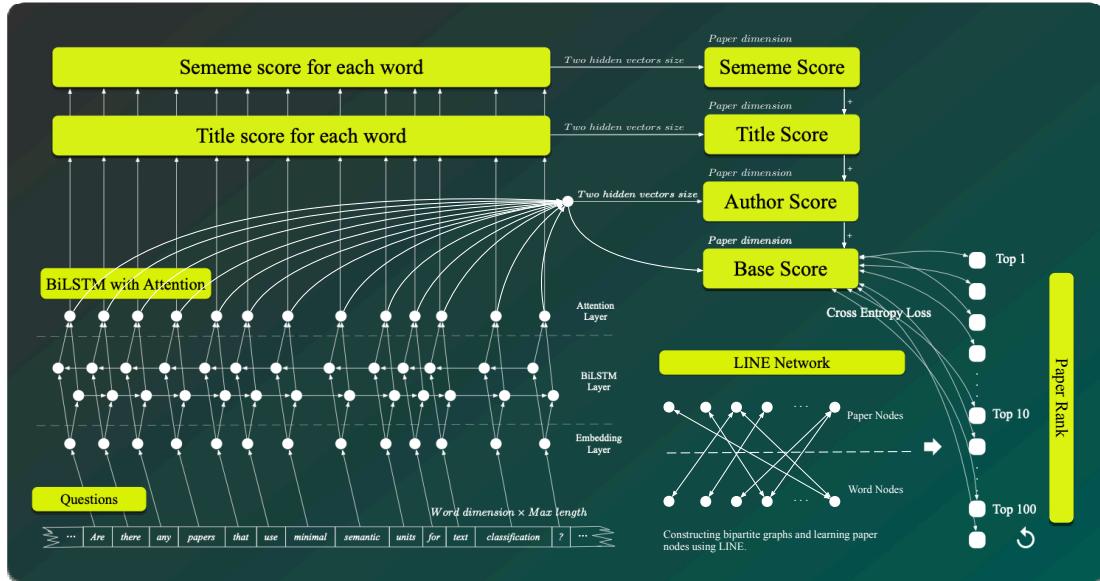


Figure 4.12: Modeling Heterogeneous Information for Paper Recommendation Systems

## Modeling Title Information

Since the words in the paper title may appear in the finder's query, encoding the paper title information into the model is necessary. First, we create a set of paper title words  $\mathbb{T}_w = \{wot_1, \dots, wot_{|\mathbb{T}_w|}\}$  and then extract each input word's hidden layer representation in a single-layer BiLSTM to predict the closest paper title words. Specifically, we use a single-layer perceptron to calculate the score of the current input word and each title word

$$score_{title}^i = \mathbf{W}_{title} h_i + b_{title}, \quad (4.5)$$

where  $score_{title}^i \in \mathbb{R}^{|\mathbb{T}|}$ ,  $\mathbf{W}_{title} \in \mathbb{R}^{|\mathbb{T}| \times 2l}$  is a weight matrix, and  $b_{title} \in \mathbb{R}^{|\mathbb{T}|}$  is a bias vector. Then keep only the title word with the highest score for each input word as

$$score_{title} = \max_{i \leq |\mathcal{D}|} (score_{title}^i). \quad (4.6)$$

Finally, the score of the paper predicted using the title words is calculated by summing:

$$score_{title,paper} = \sum_{wot \in \mathbb{T}_w} score_{title}^{index(wot)}, \quad (4.7)$$

where  $\text{index}(wot)$  returns the title word index of  $wot$ .

## Modeling Author Information

Likewise, we define  $\mathbb{A}_w = \{woa_1, \dots, woa_{|\mathbb{A}_w|}\}$  as the author set. For the author information, we directly capture the query representation through the attention mechanism to calculate the score of the current query and each author, like

$$\text{score}_{\text{author}} = \mathbf{W}_{\text{author}} v_{\text{question}} + b_{\text{author}}, \quad (4.8)$$

Similarly, the scores for each paper are obtained by adding up the author scores. The formula can be written as

$$\text{score}_{\text{author}, \text{paper}} = \sum_{woa \in \mathbb{A}_w} \text{score}_{\text{author}}^{\text{index}(woa)}, \quad (4.9)$$

## Modeling Sememe Information

A sememe is a semantic language unit of meaning, and it can also be applied to any scene that needs to express meaning. We assume that a thesis is composed of some basic meanings, then we can find the paper using some sememes. We constructed the sememe set following the method based on the Wikipedia expansion as in Section 3.3. Similarly, we guess that words similar to the sememe of the paper might appear in the finder's query, so we utilize the hidden layer vector output from BiLSTM to calculate the sememe scores of each paper separately and then predicted the most relevant paper by summing the sememe scores of each paper.

First we define the sememe set  $\mathbb{S}_w = \{wos_1, \dots, wos_{|\mathbb{S}_w|}\}$ , then calculate the sememe score past a single-layer perceptron:

$$\text{score}_{\text{sememe}}^i = \mathbf{W}_{\text{sememe}} h_i + b_{\text{sememe}}, \quad (4.10)$$

where  $\text{score}_{\text{sememe}}^i \in \mathbb{R}^{|\mathbb{S}|}$ ,  $\mathbf{W}_{\text{sememe}} \in \mathbb{R}^{|\mathbb{S}| \times 2l}$  and  $b_{\text{sememe}} \in \mathbb{R}^{|\mathbb{T}|}$  are defined as a trainable weight matrix and a bias vector, respectively. Then the sememe set of the query is calcu-

lated by max pooling as:

$$score_{sememe} = \max_{i \leq |\mathcal{D}|} (score_{sememe}^i), \quad (4.11)$$

Find the corresponding set of sememe by the index of sememe of each paper and sum them up:

$$score_{sememe,paper} = \sum_{wos \in \mathbb{S}_w} score_{sememe}^{index(wos)}, \quad (4.12)$$

The final paper score is obtained by summing up the paper prediction scores of the four features with the formula

$$score_{paper} = \sum_{type \in \{base, title, author, sememe\}} score_{type,paper}. \quad (4.13)$$

### 4.3.3 Experiments

In this subsection, we introduce our dataset, the specific statistics, and the model's hyperparameters and show the probability that our model succeeded in predicting the paper. All experiments were conducted on an NVIDIA A100-SXM4-40GB GPU.

### Settings

We collected paper abstracts from NLP's top conferences (ACL, EMNLP, NAACL)<sup>11</sup> published from 2017 to 2022 for training and showed the statistics for terminology extraction in Table 4.2 and for making the training set and the test set in Table 4.3.

Primarily, we extracted 2,000 Q&A pairs as the test set. In addition, we have taken some examples from the dataset based on each question type and presented them in Table 4.4. Note that there are cases in the one-hop type dataset where a single question points to multiple papers.

For the BiLSTM model, the dimension of non-directional hidden states is  $300 \times 2$ ,

---

<sup>11</sup><https://aclanthology.org/>

#Paper	#Trem	#Trem_wiki	#Trem_norm
13,38	58,157	57,077	55,962

**Table 4.2:** Statistics about removing common terminology and terminology normalization. Note, #trem\_wiki denotes the number of terms after removing common words in Wikipedia. #trem\_norm denotes the number of terms after normalization.

#Q&A	#Merged	#Deduped	#Split
557,067 (One-hop)	691,317	432,885	430,885 (train)
115,934 (Multi-hop)			2,000 (test)

**Table 4.3:** Q&A dataset Statistics: Represent the number of Q&A, the number after merging, the number after de-duplication, and the number after splitting into training and test sets, respectively.

Type	Question
One-hop	(1) Which paper proposed the tools that can improve the prediction of the difficulty and response time parameters for a high-stakes medical exam ?
	(2) Which paper solved issues are investigates the effectiveness of pre-training for few-shot intent classification ?
Multi-hop	(1) Which paper used the method that is one of the state-of-the-art deep learning methods used in this study to solve learns content selection and summary generation in an end-to-end fashion ?
	(2) Which paper used method is external knowledge bases (KBs) to solve issues are to improve recurrent neural networks for machine reading ?

**Table 4.4:** Examples of each question type.

the dropout rate is 0.5. we use the 300 dimensional word embeddings pretrained on Wikipedia-2014 and Gigaword-5 with GloVe<sup>12</sup>, and the word embeddings are fixed during training. For training, we adopt Adam as the optimizer with initial learning rate 0.001, the batch size is 128, and trained 100 epochs.

For the LINE model, where the size of the node embedding is 300 dimensions, the total number of training samples is 100 million, the starting value of the learning rate is 0.025, the number of negative samples is 5, and we only used second-order proximity for training. Regarding the performance of the model.

## Ablation Experiment

In evaluation. We extract the papers satisfying given prior knowledge (Node vectors of papers pre-trained on LINE) from the top 100 results of our model to evaluate the performance (Table 4.5). Note that the evaluation metric indicates the probability that the ground truth will occur within the top 100 of the model’s predicted papers.

We trained 100 epochs and plotted Figures 5 and 6 to show the model’s performance on various test sets. We find that the model flattens out after the 20th epoch, indicating that the model successfully learns valuable knowledge from the problem and can predict papers with high similarity.

In more detail, we compared two test sets, where the test data appeared in the training set (Seen) and the test set that did not appear in the training set (Unseen). It is moot to test on data that has already been seen, although we have evaluated this based on previous research. Moreover, we did ablation experiments for each group of training data separately, which are the “*base*” model with only BiLSTM, “*base + title*” model with added title information, “*base + author*” model with added author information, “*base + sememe*” model with added sememe information, and “*all*” model with fused all information. We find that the performance of the models is significantly improved after adding sememe information.

---

<sup>12</sup><https://nlp.stanford.edu/data/glove.6B.zip>

Test data	Model	Accuracy@1	Accuracy@10	Accuracy@100
Unseen	<i>base</i>	62.56	74.58	82.62
	<i>base + title</i>	64.97	78.36	86.81
	<i>base + author</i>	63.66	75.75	83.51
	<i>base + sememe</i>	67.03	80.49	89.83
Seen	<i>all</i>	67.37	80.21	90.24
	<i>base</i>	65.38	77.40	84.89
	<i>base + title</i>	68.33	81.45	88.66
	<i>base + author</i>	65.45	77.88	84.89
	<i>base + sememe</i>	69.98	84.54	93.26
<i>all</i>		70.26	84.95	94.36

Table 4.5: The performance of the BiLSTM model on all test sets. Accuracy@1/10/100 denotes the accuracy that target papers appear in top 1/10/100 (higher better)

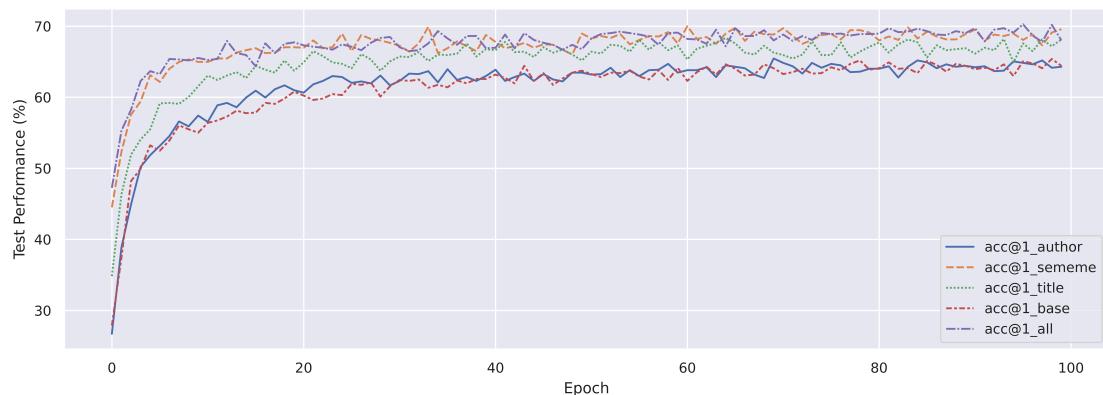


Figure 4.13: The accuracy@1 performances by using the BiLSTM model on *Seen* test set.

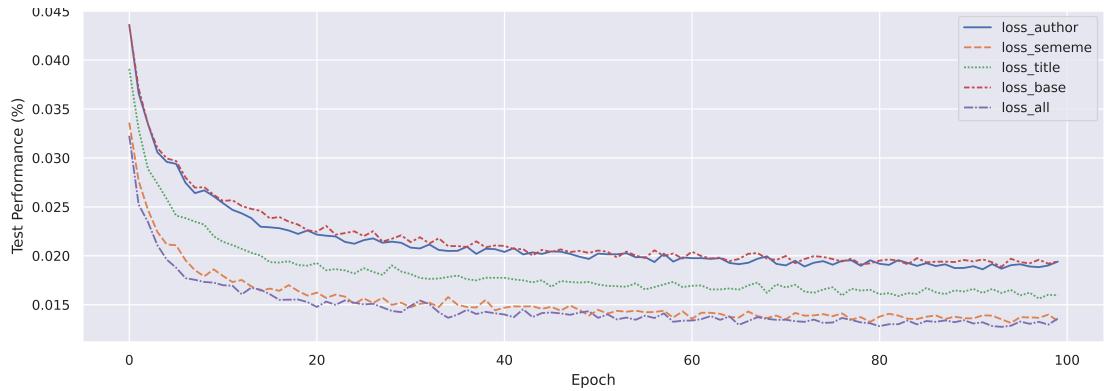


Figure 4.14: The decreasing trend of the model’s losses on *Seen* test set.

## 4.4 Conclusion

In this work, we have implemented a simple paper recommendation system from data collection, entity relationship definition, and multi-hop problem construction to the construction of a paper recommendation model, and we have confirmed the effectiveness of our model. Due to the lack of manually labeled data, we make extensive use of the knowledge provided by the pre-trained model to serve as a priori knowledge, which is an attempt to construct a knowledge base using the pre-trained model. Furthermore, we utilize the same model to predict different information to achieve information fusion. The effectiveness of our approach has been experimentally demonstrated.

## 5. Conclusion

This study focuses on leveraging an external knowledge base to enhance the model's performance. Our ultimate goal is to embed knowledge into the model, and this thesis considers only the most straightforward and primitive model structure, which is only the beginning of the research. We believe that the current "version answer" of Text Representation Learning is conducted over knowledge to pre-training directly. Naturally, this is my main research content for the next three to five years.

Moreover, the research on our paper recommendation system is still ongoing. In the future, I will focus on multi-round dialogue and cross-language recommendation systems to serve researchers better. I am convinced that this will be the topic of my doctoral dissertation.



Figure 5.1: Toward Cross-lingual and Multi-round Conversational MIYU.

Data is not information, information is not knowledge. We have always believed that the path to advanced intelligence is something other than piling up data. Although in recent years, large-scale language models have shown humans that the ceiling of machines is so close to human intelligence, this is just a greedy exploration of the limits of artificial intelligence under Moore's Law.

*“Information is not knowledge”*

– Albert Einstein

First, the quality of the data is too important. Whether it is two hundred billion or two trillion parameters of the model, they are trained data that does not filter out false information or information that is not helpful to us as humans. After all, we have seen only some of the world's information for so long. Secondly, I read a particularly poignant quote “*When monkeys looked up at the stars, humans were born.*” Perhaps at this stage of AI, there is no such advanced intelligence as thinking and self-awareness, but research on how to make machines learn to reason for themselves or "causal inference" is already underway, and it is not unattainable.

Therefore, pre-training trillion parameter models can solve 90% of the current AI problems, but this is not the optimal solution. There will be more changes before the arrival of advanced intelligence.

# Acknowledgments

臘月寒冬雪滿江，夢寐太白盡雕霜。重逢親朋眉歡時，驟醒不覺已扶桑。  
邇來倏忽將五載，尚未訴於幾還鄉。憶昔家慈傾家竭，十八相送讀書郎。  
雖餐風雨登萬樓，孤身一人仍舊樣。而立之年而未立，不知何時面親娘。  
唯有手捧拙碩論，道出恩情世難忘。初踏計算語言路，江源先生引入行。  
幸遇惠恩高野師，給予仁愛予慈祥。孜孜以求前後繼，誨爾諄諄載吾航。  
數式難解多青嶂，水野先生似明光。高山仰止行無愧，景行行止做坦蕩。  
友次亦友器注視，猶於青雲氣自強。三春之寵應衆友，畫書夜勤影印窗。  
苦盡未甘窮末路，待到澱粉千金賞。前世修得何因果，方得今生恩無量。  
積年學術綿薄力，欲立功勳助後長。才疏學淺不自量，淺書略談愚感想。  
成功失敗終相伴，唯獨後者是家常。若訴苦難何如渡，卻把衆人分兩幫。  
一幫枯朽潰難軍，一幫奮起戰四方。研究應當不懼苦，直面焦瘦滿瘡瘍。  
遍體鱗傷命使然，事後受惠愈強壯。提筆不覺寅時已，玄英天幕必將降。  
汝若此刻深於夢，怎曉昨夜人惆悵。千秋萬代皆如此，崎嶇難曉勝名揚。  
故無無心柳成蔭，漣漪終匯千層浪。徑身舒展勞疲恙，朝暉良久緩神慌。  
回首舉目窗外望，湧上心頭兩茫茫。八十一難天不負，萬字懸河終成章。

李笑然 於袋井

二〇二二年十二月十八日

---

The full explanation of the poem is on the following page.

(The no-frills meaning of the above poem) Words cannot express my gratitude to my supervisor Prof. Toshiaki Takano for his invaluable patience and feedback. I also could not have undertaken this journey without my co-supervisors, Prof. Shinya Mizuno, and Prof. Katsuko Tomotsugu, who generously provided knowledge and expertise. Additionally, this endeavor would not have been possible without the generous support from the Tokai Denpun International Scholarship, who generously financed my research. I would like to thank all the members in Takano Lab. for your zealous support and prompt advice for my research. Without your help, this thesis could not have been forwarded. Lastly, I would be remiss in not mentioning my family, especially my mother Ms. Binglan Zhang, who supports my studying abroad and provides guidance for my lifetime.

Generally, Achievability in research is always temporary; failure lasts much longer. However, how to face failure divides researchers into different formations. Some researchers will be crushed, while others will get up and move on. I think that a grown researcher should not merely seek perfection but face shortcomings head-on, which is the essence of research. In other words, research always makes us black and blue, but later, those places become stronger. Just now, it is 3:00 a.m. JST, a new dawn will rises slowly, and everyone will wake up tomorrow; how could you know such suffering without that you have seen the bright lights of the library or labs at now? However, life is insensitive, and no one cares how cost effort you have put in. People notice what position you finally stand in solely, so nothing that a casually planted willow may survive; instead, everything will fall into place when only after the struggles. Then all disappointment and sadness become immaterial.

Xiaoran Li in Fukuroi

December 18, 2022

# References

- [1] Herbert A. Simon. *The Sciences of the Artificial*. MIT Press, Cambridge, Massachusetts, first edition, 1969.
  - \\" Fundamentals of computational linguistics, including some earlier articles.
- [2] Geoffrey E Hinton et al. Learning distributed representations of concepts. In *Proceedings of the eighth annual conference of the cognitive science society*, volume 1, page 12. Amherst, MA, 1986.
- [3] Yoshua Bengio. Neural net language models. *Scholarpedia*, 3(1):3881, 2008.
- [4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- [6] Leonard Bloomfield. A set of postulates for the science of language. *Language*, 2(3):153–164, 1926.
- [7] Anna Wierzbicka. *Semantics: Primes and universals: Primes and universals*. Oxford University Press, UK, 1996.
  - \\" Related research on sememe and text deep clustering.

- [8] Zhendong Dong and Qiang Dong. *Hownet and the computation of meaning (with Cd-rom)*. World Scientific, 2006.
- [9] Roberto Navigli and Simone Paolo Ponzetto. Babelnet: Building a very large multilingual semantic network. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 216–225, 2010.
- [10] Fanchao Qi, Liang Chang, Maosong Sun, Sicong Ouyang, and Zhiyuan Liu. Towards building a multilingual sememe knowledge base: Predicting sememes for babelnet synsets. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8624–8631, 2020.
- [11] Yihong Gu, Jun Yan, Hao Zhu, Zhiyuan Liu, Ruobing Xie, Maosong Sun, Fen Lin, and Leyu Lin. Language modeling with sparse product of sememe experts. *arXiv preprint arXiv:1810.12387*, 2018.
- [12] Fanchao Qi, Junjie Huang, Chenghao Yang, Zhiyuan Liu, Xiao Chen, Qun Liu, and Maosong Sun. Modeling semantic compositionality with sememe knowledge. *arXiv preprint arXiv:1907.04744*, 2019.
- [13] Lei Zheng, Fanchao Qi, Zhiyuan Liu, Yasheng Wang, Qun Liu, and Maosong Sun. Multi-channel reverse dictionary model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 312–319, 2020.
- [14] Yuan Zang, Fanchao Qi, Chenghao Yang, Zhiyuan Liu, Meng Zhang, Qun Liu, and Maosong Sun. Word-level textual adversarial attacking as combinatorial optimization. *arXiv preprint arXiv:1910.12196*, 2019.
- [15] Erxue Min, Xifeng Guo, Qiang Liu, Gen Zhang, Jianjing Cui, and Jun Long. A survey of clustering with deep learning: From the perspective of network architecture. *IEEE Access*, 6:39501–39514, 2018.
- [16] Sidney I Landau. Bt sue atkins and michael rundell. the oxford guide to practical lexicography., 2009.

- [17] Ge Song, Yunming Ye, Xiaolin Du, Xiaohui Huang, and Shifu Bie. Short text classification: A survey. *Journal of multimedia*, 9(5):635, 2014.
- [18] Lalita Sharma and Anju Gera. A survey of recommendation system: Research challenges. *International Journal of Engineering Trends and Technology (IJETT)*, 4(5):1989–1992, 2013.
- [19] Jin Wang, Zhongyuan Wang, Dawei Zhang, and Jun Yan. Combining knowledge with deep convolutional neural networks for short text classification. In *IJCAI*, volume 350, 2017.
- [20] Jichuan Zeng, Jing Li, Yan Song, Cuiyun Gao, Michael R Lyu, and Irwin King. Topic memory networks for short text classification. *arXiv preprint arXiv:1809.03664*, 2018.  
\\ Basic research on word embedding representation learning.
- [21] Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.
- [22] Geoffrey E Hinton. Distributed representations. 1984.
- [23] DE Rumelhard, GE Hinton, and RJ Willianms. Learnin, g, internal, re, p, resentations b, y, back, p, ro, p, a, g, atin, g, errors [j]. *Nature*, 323(533-536):807, 1986.
- [24] Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. *Advances in neural information processing systems*, 27, 2014.
- [25] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.
- [26] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.
- [27] Luke Vilnis and Andrew McCallum. Word representations via gaussian embedding. *arXiv preprint arXiv:1412.6623*, 2014.

- [28] Imre Csiszár. I-divergence geometry of probability distributions and minimization problems. *The annals of probability*, pages 146–158, 1975.
- [29] Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. A neural probabilistic language model. *Advances in neural information processing systems*, 13, 2000.
- [30] ME Peters M Neumann, M Iyyer, M Gardner, C Clark, K Lee, and L Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.
- [31] Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. What do neural machine translation models learn about morphology? *arXiv preprint arXiv:1704.03471*, 2017.
- [32] Oren Melamud, Jacob Goldberger, and Ido Dagan. context2vec: Learning generic context embedding with bidirectional lstm. In *Proceedings of the 20th SIGNLL conference on computational natural language learning*, pages 51–61, 2016.
- [33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.  
\\ Research on deep clustering networks and autoencoders.
- [34] Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. Linear algebraic structure of word senses, with applications to polysemy. *Transactions of the Association for Computational Linguistics*, 6:483–495, 2018.
- [35] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.
- [36] Bo Yang, Xiao Fu, Nicholas D Sidiropoulos, and Mingyi Hong. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In *international conference on machine learning*, pages 3861–3870. PMLR, 2017.
- [37] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. Stacked denoising autoencoders: Learning useful rep-

resentations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(12), 2010.

- [38] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.
  - [39] Taku Kudo and John Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*, 2018.
  - [40] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.
  - [41] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5:135–146, 2017.
  - [42] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.
  - [43] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997.
  - [44] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.
- \ Related Research on Graph Neural Networks.
- [45] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.
  - [46] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

- [47] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29, 2016.
- [48] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.
- [49] Liang Yao, Chengsheng Mao, and Yuan Luo. Graph convolutional networks for text classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 7370–7377, 2019.
- [50] Tianchi Yang, Linmei Hu, Chuan Shi, Houye Ji, Xiaoli Li, and Liqiang Nie. Hgat: Heterogeneous graph attention networks for semi-supervised short text classification. *ACM Transactions on Information Systems (TOIS)*, 39(3):1–29, 2021.
- [51] Yaqing Wang, Song Wang, Quanming Yao, and Dejing Dou. Hierarchical heterogeneous graph representation learning for short text classification. *arXiv preprint arXiv:2111.00180*, 2021.
- [52] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.
- [53] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.
- \ Related research on the expansion of sememe knowledge base.
- [54] Xiaoran Li and Toshiaki Takano. The analysis about building cross-lingual sememe knowledge base based on deep clustering network. In *Special Interest Group on Spoken Language Understanding and Dialogue Processing 92th (2021/9)*, page 06, 2021.

- [55] Fanchao Qi, Yangyi Chen, Fengyu Wang, Zhiyuan Liu, Xiao Chen, and Maosong Sun. Automatic construction of sememe knowledge bases via dictionaries. *arXiv preprint arXiv:2105.12585*, 2021.
- [56] LIU Zhiyuan SUN Maosong LIU Yangguang, QI Fanchao. Research on consistency check of sememe annotations in hownet. 35(4):23, 2021.
- [57] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*, pages 1067–1077, 2015.
- [58] Ruobing Xie, Xingchi Yuan, Zhiyuan Liu, and Maosong Sun. Lexical sememe prediction via word embeddings and matrix factorization. In *IJCAI*, pages 4200–4206, 2017.
- [59] Eneko Agirre, Enrique Alfonsena, Keith Hall, Jana Kravalova, Marius Pasca, and Aitor Soroa. A study on similarity and relatedness using distributional and wordnet-based approaches. 2009.
- [60] George A Miller and Walter G Charles. Contextual correlates of semantic similarity. *Language and cognitive processes*, 6(1):1–28, 1991.
- [61] Herbert Rubenstein and John B Goodenough. Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633, 1965.  
\\ Related research on short text classification.
- [62] Fang Wang, Zhongyuan Wang, Zhoujun Li, and Ji-Rong Wen. Concept-based short text classification and ranking. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 1069–1078, 2014.
- [63] Ji Young Lee and Franck Dernoncourt. Sequential short-text classification with recurrent and convolutional neural networks. *arXiv preprint arXiv:1603.03827*, 2016.
- [64] Charu C Aggarwal and ChengXiang Zhai. A survey of text classification algorithms. In *Mining text data*, pages 163–222. Springer, 2012.

- [65] Yu Meng, Jiaming Shen, Chao Zhang, and Jiawei Han. Weakly-supervised neural text classification. In *proceedings of the 27th ACM International Conference on information and knowledge management*, pages 983–992, 2018.
  - [66] Jian Hu, Gang Wang, Fred Lochovsky, Jian-tao Sun, and Zheng Chen. Understanding user’s query intent with wikipedia. In *Proceedings of the 18th international conference on World wide web*, pages 471–480, 2009.
  - [67] Xiaoran Li and Toshiaki Takano. A data augmentation method for building sememe knowledge base via reconstructing dictionary definitions. *The Association for Natural Language Processing*, 2022.
  - [68] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka, and Tom M Mitchell. Toward an architecture for never-ending language learning. In *Twenty-Fourth AAAI conference on artificial intelligence*, 2010.
  - [69] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26, 2013.
  - [70] Xuan-Hieu Phan, Le-Minh Nguyen, and Susumu Horiguchi. Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In *Proceedings of the 17th international conference on World Wide Web*, pages 91–100, 2008.
  - [71] Hu Linmei, Tianchi Yang, Chuan Shi, Houye Ji, and Xiaoli Li. Heterogeneous graph attention networks for semi-supervised short text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4821–4830, 2019.
  - [72] Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. Visualizing and understanding neural models in nlp. *arXiv preprint arXiv:1506.01066*, 2015.
- \ Related papers on paper retrieval and text style conversion.

- [73] Monarch Parmar, Naman Jain, Pranjali Jain, P Jayakrishna Sahit, Soham Pachpande, Shruti Singh, and Mayank Singh. Nlpexplorer: exploring the universe of nlp papers. In *European Conference on Information Retrieval*, pages 476–480. Springer, 2020.
- [74] Di Jin, Zhijing Jin, Zhiting Hu, Olga Vechtomova, and Rada Mihalcea. Deep learning for text style transfer: A survey. *Computational Linguistics*, 48(1):155–205, 2022.
- [75] Yixin Cao, Ruihao Shui, Liangming Pan, Min-Yen Kan, Zhiyuan Liu, and Tat-Seng Chua. Expertise style transfer: A new task towards better communication between experts and laymen. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1061–1071, Online, July 2020. Association for Computational Linguistics.
- \ Some research on generating tasks based on pre-training language model
- [76] Zakariae Alami Merrouni, Bouchra Frikh, and Brahim Ouhbi. Automatic keyphrase extraction: a survey and trends. *Journal of Intelligent Information Systems*, 54(2):391–424, 2020.
- [77] Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. Unified language model pre-training for natural language understanding and generation. *Advances in Neural Information Processing Systems*, 32, 2019.
- [78] Dorottya Demszky, Kelvin Guu, and Percy Liang. Transforming question answering datasets into natural language inference datasets. *arXiv preprint arXiv:1809.02922*, 2018.
- [79] Liangming Pan, Wenhui Chen, Wenhan Xiong, Min-Yen Kan, and William Yang Wang. Unsupervised multi-hop question answering by question generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5866–5880, Online, June 2021. Association for Computational Linguistics.

- [80] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67, 2020.
- [81] Yanghoon Kim, Hwanhee Lee, Joongbo Shin, and Kyomin Jung. Improving neural question generation using answer separation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 6602–6609, 2019.
- [82] Mayank Kulkarni, Debanjan Mahata, Ravneet Arora, and Rajarshi Bhowmik. Learning rich representation of keyphrases from text. *arXiv preprint arXiv:2112.08547*, 2021.
- [83] Anette Hulth. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 216–223, 2003.
- [84] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.
- [85] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*, 2019.
- \\\ Text annotation tools and ranking models
- [86] Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun’ichi Tsujii. Brat: a web-based tool for nlp-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 102–107, 2012.
- [87] Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. Jfleg: A fluency corpus and benchmark for grammatical error correction. *arXiv preprint arXiv:1702.04066*, 2017.

- [88] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*, 2018.
- [89] Lei Zhang, Fanchao Qi, Zhiyuan Liu, Yasheng Wang, Qun Liu, and Maosong Sun. Multi-channel reverse dictionary model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 312–319, 2020.

# Published Papers

- [1] Xiaoran Li and Toshiaki Takano. The Analysis about Building Cross-lingual Sememe Knowledge Base Based on Deep Clustering Network. *Special Interest Group on Spoken Language Understanding and Dialogue Processing (SIG-SLUD)*, 92th, pages 27-32, 2021
- [2] Xiaoran Li and Toshiaki Takano. A Data Augmentation Method for Building Sememe Knowledge Base via Reconstructing Dictionary Definitions. *The 28th Annual Conference of the Association for Natural Language Processing*. pages 1802-1807, 2021
- [3] Xiaoran Li and Toshiaki Takano. Finding NLP Papers by Asking a Multi-hop Question. *Special Interest Group on Spoken Language Understanding and Dialogue Processing (SIG-SLUD)*, 95th, pages 59-46, 2022
- [4] Xiaoran Li and Toshiaki Takano. The Endeavour to Advance Short Text Classification: Using Heterogeneous Graph Neural Network via Building Sememe-relationships. *The 36th Annual Conference of the Japanese Society for Artificial Intelligence*, 2022
- [5] 李笑然 and 高野敏明. 軸言語で偽パラレルコーパスの構築による論文スタイルテキスト変換. NLP 若手の会 (YANS) 第 16 回 2021

- [6] 李笑然 and 江原遙. 単語ベクトルに基づく語彙意味の関係予測. NLP 若手の会 (YANS) 第 15 回 2020
- [7] 峯尾海成, 李笑然, 谷口ジョイ and 高野敏明. wav2vec モデルによる方言音声資料のテキスト化 Language Resources Workshop 2022

# A. Appendices

## A.1 Sememes Cluster Prediction in English

	1st	2nd	3rd	4th	5th	6th
drone	pilotless; refueling; missile (0.03559)	spacecraft; shuttlecraft; spacelab (0.03082)	rabbuh; zengi; hashimi (0.02908)	melodically; instrumentally; sonically (0.02604)	mobilizations; stationing; forces (0.01823)	gunnery; antiaircraft; airlanding (0.01649)
leaders	constitutionalists; politicize; interventionist (0.01723)	business; financial; investment (0.01462)	political; constitutionalist; leftist (0.01262)	ideological; ideology; ideologies (0.01242)	election; elections; party (0.01182)	religious; religiousness; trinitarianism (0.01142)
marijuana	medication; chronic; hospitalization (0.08371)	medication; medications; diabetic (0.03051)	arrest; suspects; retribution (0.02976)	constitutionality; constitutionally; unconstitutionality (0.0253)	recourse; overreaching; unconscionable (0.02307)	prosecution; conviction; prosecute (0.01823)
carrot	flavouring; sugared; juice (0.07528)	syrup; butter; juice (0.05256)	grasses; berries; shrubs (0.04119)	fruit; almonds; blackcurrant (0.01989)	buttered; sweetening; butter (0.01989)	scared; scaredy; crazy (0.01847)
ozone	evaporation; evaporative; contaminants (0.09809)	chemically; peroxides; nitrification (0.04774)	transpiration; particulates; contaminants (0.0434)	gaseous; vaporization; hydrogen (0.03559)	chemically; peroxides; solvents (0.02865)	protrusions; particulates; concentrically (0.02778)

Table A1: Sememes cluster prediction in English

## A.2 Sememe comparison on the Ohsumed dataset

Word	SKB	Sememe
clostridium	DictSKB	{cause, illness}

**Table A2 continued from previous page**

<b>Word</b>	<b>SKB</b>	<b>Sememe</b>
	SKB-DA	{bacterial, cell, swollen}
colitis	DictSKB	{cause, illness}
	SKB-DA	{colon, inflammation}
pediatric	DictSKB	non
	SKB-DA	{care, child, medical}
thoracic	DictSKB	{neck, part}
	SKB-DA	{chest}
empyema	DictSKB	non
	SKB-DA	{body, cavity, lung, pu}
diagnosis	DictSKB	{wrong}
	SKB-DA	{identify, nature, phenomenon}
helicobacter	DictSKB	non
	SKB-DA	{bacteria, gram, negative, shape}
infection	DictSKB	{disease, someone}
	SKB-DA	{body, invasion, microorganism, pathogenic}
salmonella	DictSKB	{make}
	SKB-DA	{Gram-negative, bioweapon, fever, food, poisoning, rod-shaped}
cerebrospinal	DictSKB	non
	SKB-DA	{brain, cord, spinal}
rhesus	DictSKB	non
	SKB-DA	{Asia, medical, southern}
mangabey	DictSKB	non
	SKB-DA	{arboreal, eyelid, limb, monkey, tail, white}
splenic	DictSKB	non
	SKB-DA	{spleen}
tissue	DictSKB	Sense1: {nose, paper, piece}
		Sense2: {paper, use, wrap}
		Sense3: {cell, form}
	SKB-DA	Sense1: {cloth, cotton, fabric, interlace, piece, strand, wool}
		Sense2: {paper, soft, translucent}
		Sense3: {cell, function, organism, structure}
itraconazole	DictSKB	non
	SKB-DA*	{fungal, infection, medication, mouth, treat}

**Table A2 continued from previous page**

<b>Word</b>	<b>SKB</b>	<b>Sememe</b>
phaeohyphomycosis	DictSKB	non
	SKB-DA*	{cell,characteristic,diverse,fungi,infection,tissue,yeast}
dermatophyte	DictSKB	non
	SKB-DA*	{chlorophyll,evolution,feed,fungi,fungus,protective}
percutaneous	DictSKB	non
	SKB-DA	{cream, form, medication, ointment, patch, skin}
venous	DictSKB	{carry}
	SKB-DA	{function, vein}
catheterization	DictSKB	non
	SKB-DA	{body, operation}
subspecialty	DictSKB	non
	SKB-DA*	{field, knowledge, medical, professional, skill, trade}
tinea	DictSKB	non
	SKB-DA	Sense1 {fungi, infection, nail, patch, skin} Sense2 {genus, moth, type}
candidiasis	DictSKB	non
	SKB-DA	{fungi, genus, infection}
immunization	DictSKB	protect
	SKB-DA*	{agent, immune, infectious, process}

Table A2: Sememe comparison on the Ohsumed dataset.

Where “*non*” means no sememe of the word, we have merged WordNetSKB and WikiSKB+ as SKB-DA, note that the SKB-DA\* with an asterisk indicates that this sememe is from WikiSKB. We extracted only the first 15 sentences of specialized words in the Ohsumed. When constructing SKB-DA, we did not purposely adjust the medical-related words, but it performed well. We found that DictSKB has insufficient vocabulary and words with similar meanings with the same sememes. e.g., “*clostridium*” and “*colitis*”. In this way, it is questioning to classify words effectively in the downstream task of natural language processing.

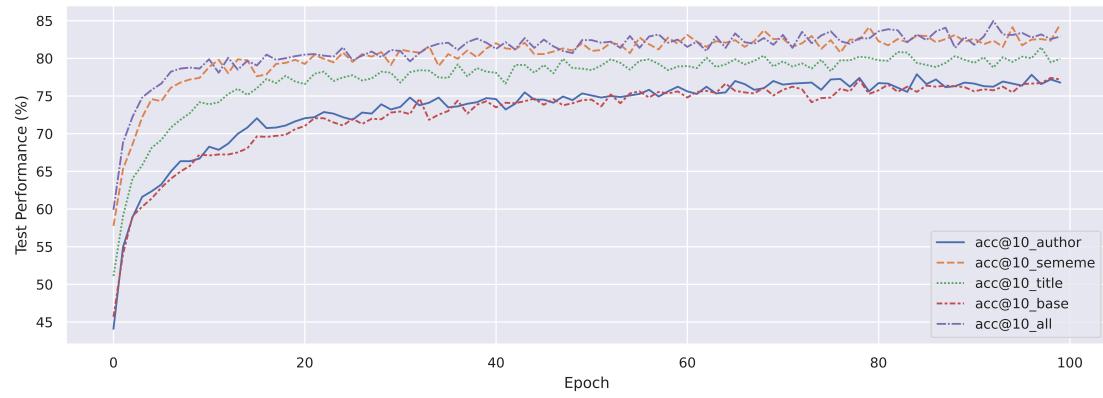


Figure B1: The accuracy@10 performances by using the BiLSTM model on *Seen* test set.

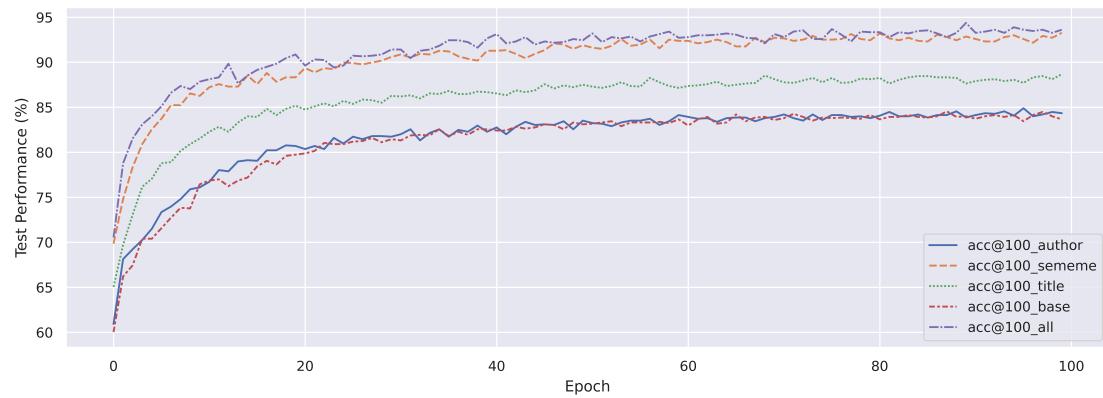


Figure B2: The accuracy@100 performances by using the BiLSTM model on *Seen* test set.

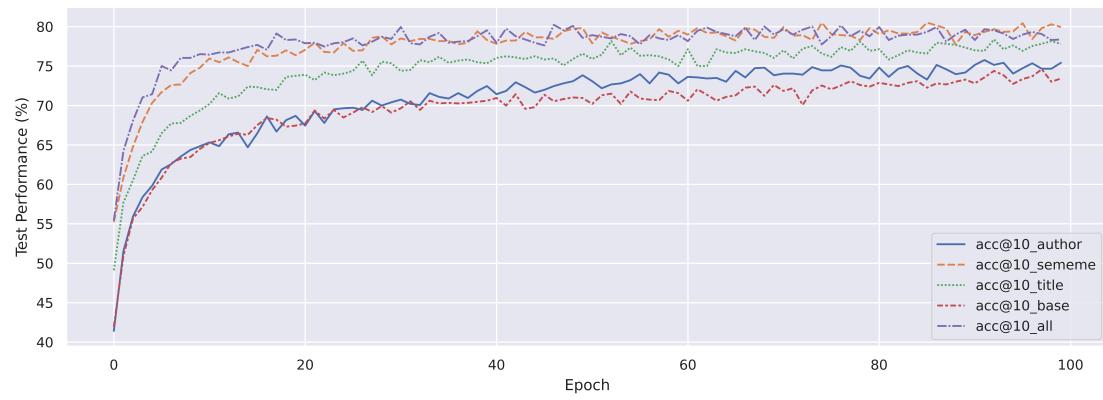


Figure B3: The accuracy@10 performances by using the BiLSTM model on *Unseen* test set.

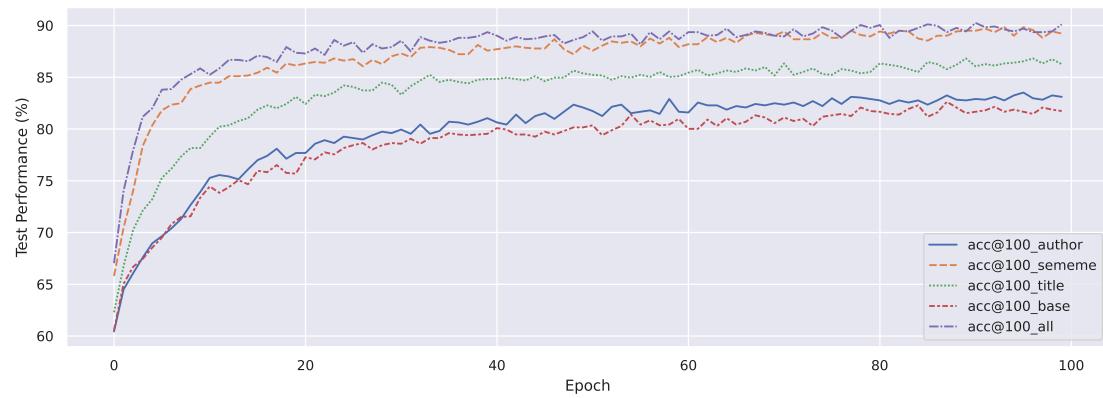


Figure B4: The accuracy@100 performances by using the BiLSTM model on *Unseen* test set.

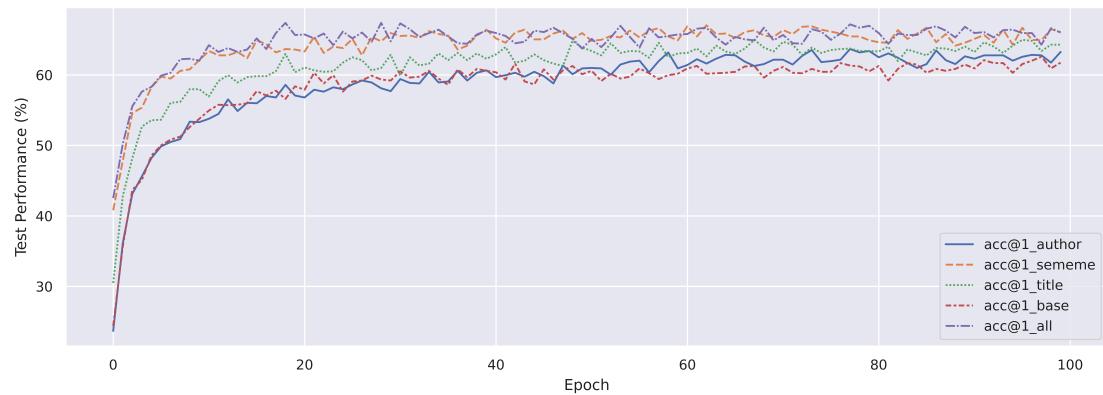


Figure B5: The accuracy@1 performances by using the BiLSTM model on *Unseen* test set.

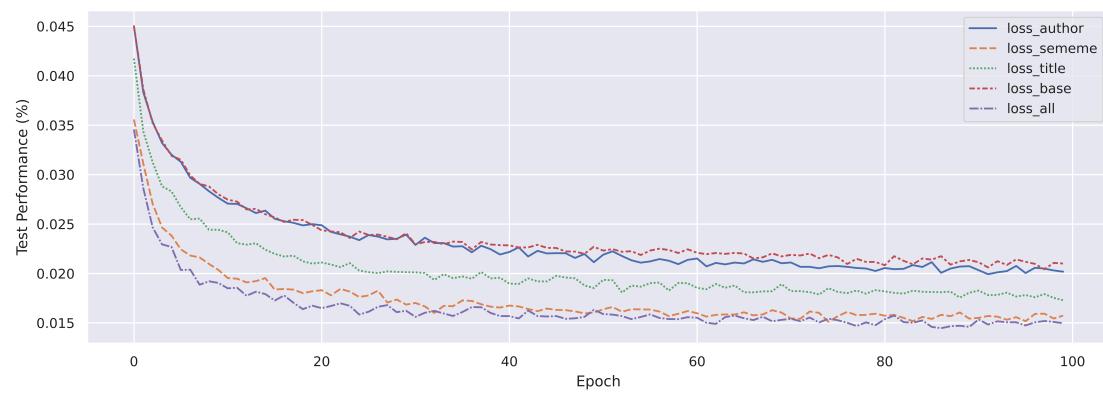


Figure B6: The decreasing trend of the model's losses on *Unseen* test set.