Abstract  Linguistic Knowledge Graphs (e.g., WordNet and HowNet) describe linguistic knowledge in formal and structural language, which can be easily incorporated in modern natural language processing systems. In this chapter, we focus on the research about HowNet. We first briefly introduce the background and basic concepts of HowNet and sememe. Next, we introduce the motivations of sememe representation learning and existing approaches. At the end of this chapter, we review important applications of sememe representation.

6.1 Introduction

In the field of Natural Language Processing (NLP), words are generally the smallest objects of study because they are considered as the smallest meaningful units that can stand by themselves of human languages. However, the meanings of words can be further divided into smaller parts. For example, the meaning of man can be considered as the combination of the meanings of human, male and adult, and the meaning of boy is composed of the meanings of human, male, and child. In linguistics, the minimum indivisible units of meaning, i.e., semantic units, are defined as sememes [8]. And some linguists believe that meanings of all the words can be composed of a limited closed set of sememes.

However, sememes are implicit and as a result, it is hard to intuitively define the set of sememes and determine which sememes a word can have at a glance. Therefore, some researchers spend tens of years sifting sememes from all kinds of dictionaries and linguistic Knowledge Bases (KBs), and annotating words with these selected sememes to construct sememe-based linguistic KB. WordNet and HowNet [17] are the two most famous ones of such KBs. In this section, we focus on the representation of linguistic knowledge in HowNet.
6.1.1 Linguistic Knowledge Graphs

6.1.1.1 WordNet

WordNet is a large lexical database for the English language and could also be viewed as a KG containing multi-relational data. It was first started in 1985, and created under the direction of George Armitage Miller, a psychology professor in the Cognitive Science Laboratory of Princeton University. Nowadays, WordNet is becoming the most popular lexicon dictionary in the world that could be available through the Web for free and is widely used in NLP applications such as text analysis, information retrieval, and relation extraction. There is also a Global WordNet Association aiming to provide a public and noncommercial platform for WordNets of all languages in the world.

Based on meanings, WordNet groups English nouns, verbs, adjectives, and adverbs into synsets (i.e., sets of cognitive synonyms), which represent a distinct concept. Each synset possesses a brief description, and in most cases, there are even some short sentences functioning as examples illustrating the use of words in this synset. The conceptual-semantic and lexical relations link the synsets and words. The main relation among words is synonymy, which indicates that the words share similar meanings and could be replaced by others in some contexts, while the main relation among synsets is hyperonymy/hyponymy (i.e., the ISA relation), which indicates the relationship between a more general synset and a more specific synset. There are also hierarchical structures for verb synsets, and the antonymy is describing the relation between adjectives with opposite meanings. To sum up, all WordNets’ 117,000 synsets are linked to each other by a small number of conceptual relations.

6.1.1.2 HowNet

HowNet was initially designed and constructed by Zhendong Dong and his son Qiang Dong in the 1990s. And it has been kept frequently updated since it was published in 1999.

The sememe set of HowNet is determined by extracting, analyzing, merging, and filtering semantics of thousands of Chinese characters. And the sememe set can also be adjusted or expanded in the subsequent process of annotating words. Each sememe in HowNet is represented by a word or phrase in Chinese and English such as (human | { 人 }) and (ProperName | { 姓 }).

HowNet also builds a taxonomy for the sememes. All the sememes of HowNet can be classified as one of the following types: Thing, Part, Attribute, Time, Space, Attribute Value, and Event. In addition, to depict the semantics of words more precisely, HowNet incorporates relations between sememes, which are called “dynamic roles”, into the sememe annotations of words.

Considering the polysemy, HowNet differentiates diverse senses of each word in the sememe annotations. And each sense is also expressed in both Chinese and
word

sense

sememe

Fig. 6.1 An example of word annotated with sememes in HowNet

Table 6.1 Statistics of HowNet

| Type                  | Count  |
|-----------------------|--------|
| Sense                 | 229,767|
| Distinct Chinese word | 127,266|
| Distinct English word | 104,025|
| Sememe                | 2,187  |

English. An example of sememe annotation for a word is illustrated in Fig. 6.1. We can see from the figure that the word apple has four senses including apple(computer), apple(phone), apple(fruit), and apple(tree), and each sense is the root node of a “sememe tree” where each pair of father and son sememe nodes is multi-relational. Additionally, HowNet annotates the POS tag for each sense, and adds sentiment category as well as some usage examples for certain senses.

The latest version of HowNet was published in January 2019 and the statistics are shown in Table 6.1.

Since HowNet was published, it has attracted wide attention. People use HowNet and sememe in various NLP tasks including word similarity computation [40], word sense disambiguation [70], question classification [62], and sentiment analysis [16, 20]. Among these researches, [40] is one of the most influential works, in which the similarity of given two words is computed by measuring the degree of resemblance of their sememe trees.

Recent years also witnessed some works incorporating sememes into neural network models. Reference [49] proposes a novel word representation learning model named SST that reforms Skip-gram [43] by adding contextual attention to senses of
the target word, which are represented with combinations of corresponding sememes’ embeddings. Experimental results show that SST can not only improve the quality of word embeddings but also learn satisfactory sense embeddings to do word sense disambiguation.

Reference [23] incorporates sememes into the decoding phase of language modeling where sememes are predicted first, and then senses and words are predicted in succession. The proposed model shows enhancement in the perplexity of language modeling and the performance of the downstream task headline generation.

Besides, HowNet is also utilized in lexicon expansion [68], semantic rationality evaluation [41], etc.

Considering that human annotation is time-consuming and labor-intensive, some works attempt to employ machine learning methods to predict sememes for new words automatically. Reference [66] proposes the task firstly and presents two simple but effective models: SPWE, which is based on collaborative filtering, and SPSE, which is based on matrix factorization. Reference [30] further takes the internal information of words into account when predicting sememes and achieves a considerable boost of performance. And [38] takes advantage of definitions of words to predict sememes. As for [56], they propose the task of cross-lingual lexical sememe prediction and present a bilingual word representation learning and alignment-based model, which demonstrates effectiveness in predicting sememes for cross-lingual words.

### 6.2 Sememe Knowledge Representation

Word Representation Learning (WRL) is a fundamental and critical step in many NLP tasks such as language modeling [4] and neural machine translation [64]. There have been a lot of researches for learning word representations, among which Word2vec [43] achieves a nice balance between effectiveness and efficiency. In Word2vec, each word corresponds to one single embedding, ignoring the polysemy of most words. To address this issue, [29] introduces a multi-prototype model for WRL, conducting unsupervised word sense induction and embeddings according to context clusters. Reference [13] further utilizes the synset information in WordNet to instruct word sense representation learning.

These previous studies demonstrate that word sense disambiguation is critical for WRL, and the sememe annotation of word senses in HowNet can provide necessary semantic regularization for these tasks [63]. To explore its feasibility, we introduce the Sememe-Encoded Word Representation Learning (SE-WRL) model, which detects word senses and learns representations simultaneously. More specifically, this framework regards each word sense as a combination of its sememes, and iteratively performs word sense disambiguation according to their contexts and learns representations of sememes, senses, and words by extending Skip-gram in Word2vec [43]. In this framework, an attention-based method is proposed to select appropriate word senses according to contexts automatically. To take full advantage...
of sememes, we introduce three different learning and attention strategies SSA, SAC, and SAT for SE-WRL, which will be described in the following paragraphs.

### 6.2.1 Simple Sememe Aggregation Model

The Simple Sememe Aggregation model (SSA) is a straightforward idea based on Skip-gram model. For each word, SSA considers all sememes in all senses of the word together, and represents the target word using the average of all its sememe embeddings. Formally, we have

$$w = \frac{1}{m} \sum_{s_i \in S^{(w)}} \sum_{x_j \in X_i^{(w)}} x_j^{(s_i)},$$

(6.1)

which means the word embedding of $w$ is composed by the average of all its sememe embeddings. Here, $S^{(w)}$ is the sense set of $w$ and $X_i^{(w)}$ is the sememe set of the $i$th sense of $w$. $m$ stands for the overall number of sememes belonging to $w$.

This model follows the assumption that the semantic meaning of a word is composed of the semantic units, i.e., sememes. As compared to the conventional Skip-gram model, since sememes are shared by multiple words, this model can utilize sememe information to encode latent semantic correlations between words. In this case, similar words that share the same sememes may finally obtain similar representations.

### 6.2.2 Sememe Attention over Context Model

The SSA Model replaces the target word embedding with the aggregated sememe embeddings to encode sememe information into word representation learning. However, each word in the SSA model still has only one single representation in different contexts, which cannot deal with the polysemy of most words. It is intuitive that we should construct distinct embeddings for a target word according to specific contexts, with the favor of word sense annotation in HowNet.

To address this issue, the Sememe Attention over Context model (SAC) is proposed. SAC utilizes the attention scheme to automatically select appropriate senses for context words according to the target word. That is, SAC conducts word sense disambiguation for context words to learn better representations of target words. The structure of the SAC model is shown in Fig. 6.2.

More specifically, SAC utilizes the original word embedding for target word $w$, and uses sememe embeddings to represent context word $w_c$ instead of the original context word embeddings. Suppose a word typically demonstrates some specific senses in one sentence. Here, the target word embedding is employed as attention
to select the most appropriate senses to make up the context word embeddings. The context word embedding $w_c$ can be formalized as follows:

$$w_c = \sum_{j=1}^{|S(w_c)|} \text{Att}(s_j^{(w_c)})s_j^{(w_c)}, \quad (6.2)$$

where $s_j^{(w_c)}$ stands for the $j$th sense embedding of $w_c$, and $\text{Att}(s_j^{(w_c)})$ represents the attention score of the $j$th sense with respect to the target word $w$, defined as follows:

$$\text{Att}(s_j^{(w_c)}) = \frac{\exp(w \cdot \hat{s}_j^{(w_c)})}{\sum_{k=1}^{|S(w_c)|} \exp(w \cdot \hat{s}_k^{(w_c)})}. \quad (6.3)$$

Note that, when calculating attention, the average of sememe embeddings is used to represent each sense $s_j^{(w_c)}$:

$$\hat{s}_j^{(w_c)} = \frac{1}{|X_j^{(w_c)}|} \sum_{k=1}^{|X_j^{(w_c)}|} x_k^{(s_j)}. \quad (6.4)$$

The attention strategy assumes that the more relevant a context word sense embedding is to the target word $w$, the more this sense should be considered when building context word embeddings. With the favor of attention scheme, each context word
can be represented as a particular distribution over its sense. This can be regarded as soft WSD and it helps learn better word representations.

### 6.2.3 Sememe Attention over Target Model

The Sememe Attention over Context Model can flexibly select appropriate senses and sememes for context words according to the target word. The process can also be applied to select appropriate senses for the target word by taking context words as attention. Hence, the Sememe Attention over Target model (SAT) is proposed, which is shown in Fig. 6.3.

Different from the SAC model, SAT learns the original word embeddings for context words and sememe embeddings for target words. Then SAT applies context words to perform attention over multiple senses of the target word \( w \) to build the embedding of \( w \), formalized as follows:

\[
    w = \sum_{j=1}^{\left|S^{(w)}\right|} \text{Att}(s^{(w)}_j) s^{(w)}_j, \tag{6.5}
\]

and the context-based attention is defined as follows:
\[ \text{Att}(s_j^{(w)}) = \frac{\exp(w_c' \cdot \hat{s}_j^{(w)})}{\sum_{k=1}^{(|S|)} \exp(w_c' \cdot \hat{s}_k^{(w)})}, \quad (6.6) \]

where the average of sememe embeddings \( \hat{s}_j^{(w)} \) is also used to represent each sense \( s_j^{(w)} \). Here, \( w_c' \) is the context embedding, consisting of a constrained window of word embeddings in \( C(w_i) \). We have

\[ w_c' = \frac{1}{2K'} \sum_{k=i-K'}^{i+K'} w_k, \quad k \neq i. \quad (6.7) \]

Note that, since in experiments, the sense selection of the target word is found to be only dependent on more limited context words for calculating attention, hence a smaller \( K' \) is selected as compared to \( K \).

Recall that SAC only uses one target word as attention to select senses of context words whereas SAT uses several context words together as attention to select appropriate senses of target words. Hence SAT is expected to conduct more reliable WSD and result in more accurate word representations, which is explored in experiments.

### 6.3 Applications

In the previous section, we introduce HowNet and sememe representation. In fact, linguistic knowledge graphs such as HowNet contain rich information which could effectively help downstream applications. Therefore, in this section, we will introduce the major applications of sememe representation, including sememe-based word representation, linguistic knowledge graph construction, and language modeling.

#### 6.3.1 Sememe-Guided Word Representation

Sememe-Guided word representation is intended for improving word embeddings for sememe prediction by introducing the information of sememe-based linguistic KBs of the source language. Qi et al. [56] present two methods of the sememe-guided word representation.

##### 6.3.1.1 Relation-Based Word Representation

A simple and intuitive method is to let words with similar sememe annotations tend to have similar word embeddings, which is named as word relation-based approach. To begin with, a synonym list is constructed from sememe-based linguistic KBs,
where words sharing a certain number of sememes are regarded as synonyms. Next, synonyms are forced to have closer word embeddings.

Formally, let $w_i$ be the original word embedding of $w_i$ and $\hat{w}_i$ be its adjusted word embedding. And let $\text{Syn}(w_i)$ denote the synonym set of word $w_i$. Then the loss function is defined as

$$L_{\text{sememe}} = \sum_{w_i \in V} \left[ \alpha_i \| w_i - \hat{w}_i \|^2 + \sum_{w_j \in \text{Syn}(w_i)} \beta_{ij} \| \hat{w}_i - \hat{w}_j \|^2 \right],$$

(6.8)

where $\alpha$ and $\beta$ control the relative strengths of the two terms. It should be noted that the idea of forcing similar words to have close word embeddings is similar to the state-of-the-art retrofitting approach [19]. However, the retrofitting approach cannot be applied here because sememe-based linguistic KBs such as HowNet cannot directly provide its needed synonym list.

6.3.1.2 Sememe Embedding-Based Word Representation

Simple and effective as the word relation-based approach is, it cannot make full use of the information of sememe-based linguistic KBs because it disregards the complicated relations between sememes and words as well as relations between different sememes. To address this limitation, the sememe embedding-based approach is proposed, which learns both sememe and word embeddings jointly.

In this approach, sememes are represented with distributed vectors as well and place them into the same semantic space as words. Similar to SPSE [66], which learns sememe embeddings by decomposing the word-sememe matrix and sememe-sememe matrix, the method utilizes sememe embeddings as regularizers to learn better word embeddings. Different from SPSE, the model described in [56] does not use pretrained word embeddings. Instead, it learns word embeddings and sememe embeddings simultaneously. More specifically, a word-sememe matrix $M$ can be extracted from HowNet, where $M_{ij} = 1$ indicates word $w_i$ is annotated with sememe $x_j$, otherwise $M_{ij} = 0$. Hence by factorizing $M$, the loss function can be defined as

$$L_{\text{sememe}} = \sum_{w_i \in V, x_j \in X} (w_i \cdot x_j + b_i + b'_j - M_{ij})^2,$$

(6.9)

where $b_i$ and $b'_j$ are the biases of $w_i$ and $x_j$, and $X$ denotes sememe set.

In this approach, word and sememe embeddings are obtained in a unified semantic space. The sememe embeddings bear all the information about the relationships between words and sememes, and they inject the information into word embeddings. Therefore, the word embeddings are expected to be more suitable for sememe prediction.
6.3.2 Sememe-Guided Semantic Compositionality Modeling

Semantic Compositionality (SC) is defined as the linguistic phenomenon that the meaning of a syntactically complex unit is a function of meanings of the complex unit’s constituents and their combination rule [50]. Some linguists regard SC as the fundamental truth of semantics [51]. In the field of NLP, SC has proved effective in many tasks including language modeling [47], sentiment analysis [42, 61], syntactic parsing [59], etc.

Most literature on SC pays attention to using vector-based distributional models of semantics to learn representations of Multiword Expressions (MWEs), i.e., embeddings of phrases or compounds. Reference [46] conducts a pioneering work which introduces a general framework to formulate this task:

\[
p = f(w_1, w_2, R, K), \tag{6.10}
\]

where \( f \) is the compositionality function, \( p \) denotes the embedding of an MWE, \( w_1 \) and \( w_2 \) represent the embeddings of the MWE’s two constituents, \( R \) stands for the combination rule, and \( K \) refers to the additional knowledge which is needed to construct the semantics of the MWE.

Most of the proposed approaches ignore \( R \) and \( K \), centering on reforming compositionality function \( f \) [3, 21, 60, 61]. Some try to integrate combination rule \( R \) into SC models [7, 35, 65, 71]. A few works consider external knowledge \( K \). Reference [72] tries to incorporate task-specific knowledge into an LSTM model for sentence-level SC.

Reference [55] proposes a novel sememe-based method to model semantic compositionality. They argue that sememes are beneficial to modeling SC. To verify this, they first design a simple SC degree (SCD) measurement experiment and find that the SCDs of MWEs computed by simple sememe-based formulae are highly correlated with human judgment. This result shows that sememes can finely depict meanings of MWEs and their constituents, and capture the semantic relations between the two sides. Moreover, they propose two sememe-incorporated SC models for learning embeddings of MWEs, namely Semantic Compositionality with Aggregated Sememe (SCAS) model and Semantic Compositionality with Mutual Sememe Attention (SCMSA) model. When learning the embedding of an MWE, the SCAS model concatenates the embeddings of the MWE’s constituents and their sememes, while the SCMSA model considers the mutual attention between a constituent’s sememes and the other constituent. Finally, they integrate the combination rule, i.e., \( R \) in Eq. (6.10), into the two models. Their models achieve significant performance over the MWE similarity computation task and sememe prediction task compared with baseline methods.

In this section, we focus on the work conducted by [55]. We will first introduce sememe-based SC Degree (SCD) computation formulae, and then expand their Sememe-incorporated SC models.

\footnote{This formula only applies to two-word MWEs but can be easily extended to longer MWEs.}
6.3.2.1 Sememe-Based SCD Computation Formulae

Although SC widely exists in MWEs, not every MWE is fully semantically compositional. In fact, different MWEs show different degrees of SC. Reference [55] believes that sememes can be used to measure SCD conveniently.

To this end, based on the assumption that all the sememes of a word accurately depict the word’s meaning, they intuitively design a set of SCD computation formulae, which are consistent with the principle of SCD. The formulae are illustrated in Table 6.2. They define four SCDs denoted by numbers 3, 2, 1, and 0, where larger numbers mean higher SCDs. $S_p$, $S_{w_1}$, and $S_{w_2}$ represent the sememe sets of an MWE, its first and second constituent, respectively. Here is a brief explanation for their SCD computation formulae:

1. For SCD 3, the sememe set of an MWE is identical to the union of the two constituents’ sememe sets, which means the meaning of the MWE is exactly the same as the combination of the constituents’ meanings. Therefore, the MWE is fully semantically compositional and should have the highest SCD.

2. For SCD 0, an MWE has totally different sememes from its constituents, which means the MWE’s meaning cannot be derived from its constituents’ meanings. Hence the MWE is completely non-compositional, and its SCD should be the lowest.

3. As for SCD 2, the sememe set of an MWE is a proper subset of the union of its constituents’ sememe sets, which means the meanings of the constituents cover the MWE’s meaning but cannot precisely infer the MWE’s meaning.

4. Finally, for SCD 1, an MWE shares some sememes with its constituents, but both the MWE itself and its constituents have some unique sememes.

There is an example for each SCD in Table 6.2, including a Chinese MWE, its two constituents, and their sememes. 2

6.3.2.2 Evaluating SCD Computation Formulae

To evaluate their sememe-based SCD computation formulae, [55] constructs a human-annotated SCD dataset. They ask several native speakers to label SCDs for 500 Chinese MWEs, where there are four degrees to choose. Before labeling an MWE, they are shown the dictionary definitions of both the MWE and its constituents.

Each MWE is labeled by 3 annotators, and the average of the 3 SCDs given by them is the MWE’s final SCD.

Eventually, they obtain a dataset containing 500 Chinese MWEs together with their human-annotated SCDs.

Then they evaluate the correlativity between SCDs of the MWEs in the dataset computed by sememe-based rules and those given by humans. They find Pearson’s correlation coefficient is up to 0.75, and Spearman’s rank correlation coefficient is

---

2 In Chinese, most MWEs are words consisting of more than two characters which are actually single-morpheme words.
**Table 6.2** Sememe-based semantic compositionality degree computation formulae and examples. Bold sememes of constituents are shared with the constituents’ corresponding MWE

| SCD | Our Computation Formulae | MWEs and Constituents | Examples |
|-----|--------------------------|-----------------------|----------|
| 3   | $S_p = S_{w_1} \cup S_{w_2}$ | 农民起义 (peasant uprising) 职位 | 事情 | fact, 职位 | occupation, 政 | politics, 暴动 | uprise, 人 | human, 农 | agricultural |
|     |                           | 农民 (peasant) 起义 (uprising) |           | 数学 | math, 图像 | image |
| 2   | $S_p \subseteq (S_{w_1} \cup S_{w_2})$ | 几何图形 (geometric figure) 几何 (geometry; how much) | 数学 | math, 知识 | knowledge, 疑问 | question, 功能词 | funcword |
|     |                           | 图形 (figure) | 图像 | image |
| 1   | $S_p \cap (S_{w_1} \cup S_{w_2}) \neq \emptyset \wedge S_p \not\subseteq (S_{w_1} \cup S_{w_2})$ | 应考 (engage a test) 应 (deal with; echo; agree) | 考试 | exam, 从事 | engage |
|     |                           | (quiz; check) | 处理 | handle, 回应 | respond, 同意 | agree, 遵循 | obey, 功能词 | funcword, 姓 | surname |
|     |                           | 考试 | exam, 查 | check |
| 0   | $S_p \cap (S_{w_1} \cup S_{w_2}) = \emptyset$ | 画句子 (end) 画 (draw) | 完字 | finish |
|     |                           | 画句号 (period) | 符号 | symbol, 语文 | text |
These results manifest remarkable capability of sememes to compute SCDs of MWEs and provide a proof that sememes of a word can finely represent the word’s meaning.

### Sememe-Incorporated SC Models

In this section, we first introduce two basic sememe-incorporated SC models in detail, namely Semantic Compositionality with Aggregated Sememe (SCAS) and Semantic Compositionality with Mutual Sememe Attention (SCMSA). SCAS model simply concatenates the embeddings of the MWE’s constituents and their sememes, while the SCMSA model takes account of the mutual attention between a constituent’s sememes and the other constituent. Then we describe how to integrate combination rules into the two basic models.

**Incorporating Sememes Only.** Following the notations in Eq. (6.10), for an MWE \( p = \{w_1, w_2\} \), its embedding can be represented as

\[
p = f(w_1, w_2, \mathcal{K}),
\]

where \( p, w_1, w_2 \in \mathbb{R}^d \), and \( d \) is the dimension of embeddings, \( \mathcal{K} \) denotes the sememe knowledge here, and we assume that we only know the sememes of \( w_1 \) and \( w_2 \), considering that MWEs are normally not in the sememe KBs. \( X \) indicates the set of all the sememes and \( X_w = \{x_1, ..., x_{|X_w|}\} \subset X \) to signify the sememe set of \( w \). In addition, \( x \in \mathbb{R}^d \) denotes the embedding of sememe \( x \).

1. **SCAS Model** The first model we introduce is the SCAS model, which is illustrated in Fig. 6.4. The idea of the SCAS model is straightforward, i.e., simply

![Fig. 6.4 The architecture of SCAS model](image_url)
concatenating word embedding of a constituent and the aggregation of its sememes’ embeddings. Formally, we have

$$w'_1 = \sum_{x_i \in X_w_1} x_i, \quad w'_2 = \sum_{x_j \in X_w_2} x_j,$$

(6.12)

where \(w'_1\) and \(w'_2\) represent the aggregated sememe embeddings of \(w_1\) and \(w_2\), respectively. Then \(p\) can be obtained by

$$p = \tanh(W_c[w_1 + w'_1; w_2 + w'_2] + b_c),$$

(6.13)

where \(W_c \in \mathbb{R}^{d \times 2d}\) is the composition matrix and \(b_c \in \mathbb{R}^d\) is a bias vector.

(2) SCMSA Model

The SCAS model simply uses the sum of all the sememes’ embeddings of a constituent as the external information. However, a constituent’s meaning may vary with the other constituent, and accordingly, the sememes of a constituent should have different weights when the constituent is combined with different constituents (there is an example in the case study).

Correspondingly, we introduce the SCMSA model (Fig. 6.5), which adopts the mutual attention mechanism to dynamically endow sememes with weights. Formally, we have

$$e_1 = \tanh(W_a w_1 + b_a),$$

$$a_{2,i} = \frac{\exp(s_i \cdot e_1)}{\sum_{x_j \in X_w_2} \exp(x_j \cdot e_1)},$$

$$w'_2 = \sum_{x_i \in X_w_2} a_{2,i} x_i,$$

(6.14)

where \(W_a \in \mathbb{R}^{d \times d}\) is the weight matrix and \(b_a \in \mathbb{R}^d\) is a bias vector. Similarly, \(w'_1\) can be calculated. Then they still use Eq. (6.13) to obtain \(p\).

Integrating Combination Rules. Reference [55] further integrates combination rules into their sememe-incorporated SC models. In other words,

$$p = f(w_1, w_2, K, R).$$

(6.15)

We can use totally different composition matrices for MWEs with different combination rules:

$$W_c = W'_c, \quad r \in R_s,$$

(6.16)

where \(W'_c \in \mathbb{R}^{d \times 2d}\) and \(R_s\) refers to combination rule set containing syntax rules of MWEs, e.g., adjective-noun and noun-noun.

However, there are many different combination rules, and some rules have sparse instances which are not enough to train the corresponding composition matrices
with $d \times 2d$ parameters. In addition, we believe that the composition matrix should contain common compositionality information except the combination rule-specific compositionality information. Hence, they let composition matrix $W_c$ be the sum of a low-rank matrix containing combination rule information and a matrix containing common compositionality information:

$$W_c = U_1^r U_2^r + W_c^c,$$  \hspace{1cm} (6.17)

where $U_1^r \in \mathbb{R}^{d \times d_r}$, $U_2^r \in \mathbb{R}^{d_r \times 2d}$, and $d_r \in \mathbb{N}_+$ is a hyperparameter and may vary with the combination rule, and $W_c^c \in \mathbb{R}^{d \times 2d}$.

### 6.3.3 Sememe-Guided Language Modeling

Language Modeling (LM) aims to measure the probability of a word sequence, reflecting its fluency and likelihood as a feasible sentence in a human language. Language Modeling is an essential component in a wide range of natural language
processing (NLP) tasks, such as machine translation [9, 10], speech recognition [34], information retrieval [5, 24, 45, 54], and document summarization [2, 57].

A probabilistic language model calculates the conditional probability of the next word given their contextual words, which are typically learned from large-scale text corpora. Taking the simplest language model, for example, $n$-gram estimates the conditional probabilities according to maximum likelihood over text corpora [31]. Recent years have witnessed the advances of Recurrent Neural Networks (RNNs) as the state-of-the-art approach for language modeling [44], in which the context is represented as a low-dimensional hidden state to predict the next word (Fig. 6.6).

Those conventional language models, including neural models, typically assume words as atomic symbols and model sequential patterns at the word level. However, this assumption does not necessarily hold to some extent. Consider the following example sentence for which people want to predict the next word in the blank,

The U.S. trade deficit last year is initially estimated to be 40 billion____.

People may first realize a unit should be filled in, then realize it should be a currency unit. Based on the country this sentence is talking about, the U.S., one may confirm it should be an American currency unit and predict the word dollars. Here, the unit, currency, and American, which are basic semantic units of the word dollars, are also the sememes of the word dollars. However, this process has not been explicitly taken into consideration by conventional language models. That is in most cases, words are atomic language units, words are not necessarily atomic semantic units for language modeling. Thus, explicit modeling of sememes could improve both the performance and the interpretability of language models. However, as far as we know, a few efforts have been devoted to exploring the effectiveness of sememes in language models, especially neural language models.

It is nontrivial for neural language models to incorporate discrete sememe knowledge, as it is not compatible with continuous representations in neural models. In this part, Sememe-Driven Language Model (SDLM) is proposed to leverage lexical sememe knowledge. In order to predict the next word, SDLM utilizes a novel
sememe-sense-word generation process: (1) First, SDLM estimates sememes’ distribution according to the context. (2) Regarding these sememes as experts, SDLM employs a sparse product of expert method to select the most probable senses. (3) Finally, SDLM calculates the distribution of words by marginalizing out the distribution of senses.

SDLM is composed of three modules in series: Sememe Predictor, Sense Predictor, and Word Predictor (Fig. 6.6). The Sememe Predictor first takes the context vector as input and assigns a weight to each sememe. Then each sememe is regarded as an expert and makes predictions about the probability distribution over a set of senses in the Sense Predictor. Finally, the probability of each word is obtained in the Word Predictor.

**Sememe Predictor.** The Sememe Predictor takes the context vector \( \mathbf{g} \in \mathbb{R}^{H_1} \) as input and assigns a weight to each sememe. Assume that given the context \( w_1, w_2, \ldots, w_{t-1} \), the events that word \( w_t \) contains sememe \( x_k \) (\( k \in \{1, 2, \ldots, K\} \)) are independent, since the sememe is the minimum semantic unit and there is no semantic overlap between any two different sememes. For simplicity, the superscript \( t \) is ignored. The Sememe Predictor is designed as a linear decoder with the sigmoid activation function. Therefore, \( p_k \), the probability that the next word contains sememe \( x_k \), is formulated as

\[
p_k = P(x_k | \mathbf{g}) = \text{Sigmoid}(\mathbf{g} \cdot \mathbf{v}_k + b_k), \quad (6.18)
\]

where \( \mathbf{v}_k \in \mathbb{R}^{H_1}, b_k \in \mathbb{R} \) are trainable parameters, and \( \text{Sigmoid}(\cdot) \) denotes the sigmoid activation function.

**Sense Predictor and Word Predictor.** The architecture of the Sense Predictor is motivated by Product of Experts (PoE) [25]. Each sememe is regarded as an expert that only makes predictions on the senses connected with it. Let \( S^{(x_k)} \) denote the set of senses that contain sememe \( x_k \), the \( k \)th expert. Different from conventional neural language models, which directly use the inner product of the context vector \( \mathbf{g} \in \mathbb{R}^{H_1} \) and the output embedding \( \mathbf{w} \in \mathbb{R}^{H_2} \) for word \( w \) to generate the score for each word, Sense Predictor uses \( \phi^{(k)}(\mathbf{g}, \mathbf{w}) \) to calculate the score given by expert \( x_k \). And a bilinear function parameterized with a matrix \( \mathbf{U}_k \in \mathbb{R}^{H_1 \times H_2} \) is chosen as a straight implementation of \( \phi^{(k)}(\cdot, \cdot) \):

\[
\phi^{(k)}(\mathbf{g}, \mathbf{w}) = \mathbf{g}^\top \mathbf{U}_k \mathbf{w}. \quad (6.19)
\]

The score of sense \( s \) provided by sememe expert \( x_k \) can be written as \( \phi^{(k)}(\mathbf{g}, s) \). Therefore, \( P^{(x_k)}(s | \mathbf{g}) \), the probability of sense \( s \) given by expert \( x_k \), is formulated as

\[
P^{(x_k)}(s | \mathbf{g}) = \frac{\exp(q_k C_{k,s} \phi^{(k)}(\mathbf{g}, s))}{\sum_{s' \in S^{(x_k)}} \exp(q_k C_{k,s'} \phi^{(k)}(\mathbf{g}, s'))}, \quad (6.20)
\]

where \( C_{k,s} \) is a normalization constant because sense \( s \) is not connected to all experts (the connections are sparse with approximately \( \lambda N \) edges, \( \lambda < 5 \)). Here we can
choose either $C_{k,s} = 1/|X^{(s)}|$ \textit{(left normalization)} or $C_{k,s} = 1/\sqrt{|X^{(s)}||S^{(x_k)}}$ \textit{(symmetric normalization)}.

In the Sense Predictor, $q_k$ can be viewed as a gate which controls the magnitude of the term $C_{k,s}\phi^{(k)}(g, s)$, thus controlling the flatness of the sense distribution provided by sememe expert $x_k$. Consider the extreme case when $p_k \rightarrow 0$, the prediction will converge to the discrete uniform distribution. Intuitively, it means that the sememe expert will refuse to provide any useful information when it is not likely to be related to the next word.

Finally, the predictions can be summarized on sense $s$ by taking the product of the probabilities given by relevant experts and then normalize the result; that is to say, $P(s|g)$, the probability of sense $s$, satisfies

$$P(s|g) \propto \prod_{x_k \in X^{(i)}} P^{(x_k)}(s|g).$$ \hspace{1cm} (6.21)

Using Eqs. 6.19 and 6.20, $P(s|g)$ can be formulated as

$$P(s|g) = \frac{\exp(\sum_{x_k \in X^{(i)}} q_k C_{k,s}g^T U_k s)}{\sum_{s'} \exp(\sum_{x_k \in X^{(i')}} q_k C_{k,s'}g^T U_k s')}.$$ \hspace{1cm} (6.22)

It should be emphasized that all the supervision information provided by HowNet is embodied in the connections between the sememe experts and the senses. If the model wants to assign a high probability to sense $s$, it must assign a high probability to some of its relevant sememes. If the model wants to assign a low probability to sense $s$, it can assign a low probability to its relevant sememes. Moreover, the prediction made by sememe expert $x_k$ has its own tendency because of its own $\phi^{(k)}(\cdot, \cdot)$. Besides, the sparsity of connections between experts and senses is also determined by HowNet itself.

As illustrated in Fig. 6.7, in the Word Predictor, $P(w|g)$, the probability of word $w$ is calculated by summing up probabilities of corresponding $s$ given by the Sense Predictor, that is

$$P(w|g) = \sum_{s \in S^{(w)}} P(s|g).$$ \hspace{1cm} (6.23)

### 6.3.4 Sememe Prediction

The manual construction of HowNet is actually time-consuming and labor-intensive, e.g., HowNet has been built for more than 10 years by several linguistic experts. However, as the development of communications and techniques, new words and phrases are emerging, the semantic meanings of existing words are also dynamically evolving. In this case, sustained manual annotation and updates are becoming much more overwhelmed. Moreover, due to the high complexity of sememe ontology and
word meanings, it is also challenging to maintain annotation consistency among experts when they collaboratively annotate lexical sememes.

To address the issues of inflexibility and inconsistency of manual annotation, the automatic lexical sememe prediction task is proposed, which is expected to assist expert annotation and reduce manual workload. Note that for simplicity, most works introduced in this part do not consider the complicated hierarchies of word sememes, and simply group all annotated sememes of each word as the sememe set for learning and prediction.

The basic idea of sememe prediction is that those words of similar semantic meanings may share overlapped sememes. Hence, the key challenge of sememe prediction is how to represent semantic meanings of words and sememes to model the semantic relatedness between them. In this part, we will focus on introducing the sememe prediction word accomplished by Xie et al. [66]. In their work, they propose to model the semantics of words and sememes using distributed representation learning [26]. Distributed representation learning aims to encode objects into a low-dimensional semantic space, which has shown its impressive capability of modeling semantics of human languages, e.g., word embeddings [43] have been widely studied and utilized in various tasks of NLP.

As shown in previous work [43], it is effective to measure word similarities using cosine similarity or Euclidean distance of their word embeddings learned from a large-scale text corpus. Hence, a straightforward method for sememe prediction is
that, given an unlabeled word, we find its most related words in HowNet according
to their word embeddings, and recommend the annotated sememes of these related
words to the given word. The method is intrinsically similar to collaborative filtering
[58] in recommendation systems, capable of capturing semantic relatedness between
words and sememes based on their annotation co-occurrences.

Word embeddings can also be learned with techniques of matrix factorization
[37]. Inspired by the successful practice of matrix factorization for personalized
recommendation [36], a new model which factorizes the word-sememe matrix from
HowNet and obtains sememe embeddings is proposed. In this way, the relatedness of
words and sememes can be measured directly using dot products of their embeddings,
according to which we could recommend the most related sememes to an unlabeled
word.

The two methods are named as Sememe Prediction with Word Embeddings
(SPWE) and with Sememe Embeddings (SPSE/SPASE), respectively.

6.3.4.1 Sememe Prediction with Word Embeddings

Given an unlabeled word, it is straightforward to recommend sememes according to
its most related words, assuming that similar words should have similar sememes.
This idea is similar to collaborative filtering in the personalized recommendation, for
in the scenario of sememe prediction words can be regarded as users and sememes
as the items/products to be recommended. Inspired by this, Sememe Prediction
with Word Embeddings (SPWE) model is proposed, which uses similarities of word
embeddings to judge user distances.

Formally, the score function \( P(x_j|w) \) of sememes \( x_j \) given a word \( w \) is defined
as

\[
P(x_j|w) = \sum_{w_i \in V} \cos(w, w_i) M_{ij} c^{r_{i}},
\]

where \( \cos(w, w_i) \) is the cosine similarity between word embeddings of \( w \) and \( w_i \)
pretrained by GloVe. \( M_{ij} \) indicates the annotation of sememe \( x_j \) on word \( w_i \), where
\( M_{ij} = 1 \) indicates the word \( w_i \) which has the sememe \( x_j \) in HowNet and otherwise
has not. Higher the score function \( P(x_j|w) \) is, more possible the word \( w \) should be
recommended with \( x_j \).

Differing from classical collaborative filtering in recommendation systems, only
the most similar words should be concentrated when predicting sememes for new
words since irrelevant words have totally different sememes which may be noises
for sememe prediction. To address this problem, a declined confidence factor \( c^{r_{i}} \) is
assigned for each word \( w_i \), where \( r_{i} \) is the descend rank of word similarity \( \cos(w, w_i) \),
and \( c \in (0, 1) \) is a hyperparameter. In this way, only a few top words that are similar
to \( w \) have strong influences on predicting sememes.

SPWE only uses word embeddings for word similarities and is simple and effect-
ive for sememe prediction. It is because, differing from the noisy and incomplete
user-item matrix in most recommender systems, HowNet is carefully annotated by
human experts, and thus the word-sememe matrix is with high confidence. Therefore, the word-sememe matrix can be confidently applied to collaboratively recommend reliable sememes based on similar words.

### 6.3.4.2 Sememe Prediction with Sememe Embeddings

Sememe Prediction with Word Embeddings model follows the assumption that the sememes of a word can be predicted according to its related words’ sememes. However, simply considering sememes as discrete labels may inevitably neglect the latent relations between sememes. To take the latent relations of sememes into consideration, Sememe Prediction with Sememe Embeddings (SPSE) model is proposed, which projects both words and sememes into the same semantic vector space, learning sememe embeddings according to the co-occurrences of words and sememes in HowNet.

Similar to GloVe [53] which decomposes co-occurrence matrix of words to learn word embeddings, sememe embeddings can be learned by factorizing word-sememe matrix and sememe-sememe matrix simultaneously. These two matrices are both constructed from HowNet. As for word embeddings, similar to SPWE, SPSE uses word embeddings pretrained from a large-scale corpus and fixes them during factorizing of the word-sememe matrix. With matrix factorization, both sememe and word embeddings can be encoded into the same low-dimensional semantic space, and then computed the cosine similarity between normalized embeddings of words and sememes for sememe prediction.

More specifically, similar to M, a sememe-sememe matrix \( C \) can also be extracted, where \( C_{jk} \) is defined as point-wise mutual information that 
\[
C_{jk} = \text{PMI}(x_j, x_k)
\]
to indicate the correlations between two sememes \( x_j \) and \( x_k \). Note that, by factorizing \( C \), two distinct embeddings for each sememe \( s \) will be obtained, denoted as \( x \) and \( \bar{x} \), respectively. The loss function of learning sememe embeddings is defined as follows:

\[
\mathcal{L} = \sum_{w_i \in W, x_j \in X} (w_i \cdot (x_j + \mathbf{N}_j) + b_i + b'_j - M_{ij})^2 + \lambda \sum_{x_j, x_k \in X} (x_j \cdot \bar{x}_k - C_{jk})^2,
\]

(6.25)

where \( b_i \) and \( b'_j \) denote the bias of \( w_i \) and \( x_j \). These two parts correspond to the losses of factorizing matrices \( M \) and \( C \), adjusted by the hyperparameter \( \lambda \). Since the sememe embeddings are shared by both factorizations, our SPSE model enables jointly encoding both words and sememes into a unified semantic space.

Since each word is typically annotated with 2–5 sememes in HowNet, most elements in the word-sememe matrix are zeros. If all zero elements and nonzero elements are treated equally during factorization, the performance will be much worse. To address this issue, different factorization strategies are assigned for zero and nonzero elements. For each zero element, the model chooses to factorize them with a small probability like 0.5%, and otherwise, the model chooses to ignore. While for nonzero elements, the model always chooses to factorize them. With the help of this strategy, the model can pay more attention to those annotated word-sememe pairs.
In SPSE, sememe embeddings are learned accompanying with word embeddings via matrix factorization into the unified low-dimensional semantic space. Matrix factorization has been verified as an effective approach in the personalized recommendation, because it can accurately model relatedness between users and items, and is highly robust to noises in user-item matrices. Using this model, we can flexibly compute semantic relatedness of words and sememes, which provides us an effective tool to manipulate and manage sememes, including but not limited to sememe prediction.

### 6.3.4.3 Sememe Prediction with Aggregated Sememe Embeddings

Inspired by the characteristics of sememes, we assume that the word embeddings are semantically composed of sememe embeddings. In the word-sememe joint space, we can simply implement semantic composition as additive operations that each word embedding is expected to be the sum of its all sememes’ embeddings. Following this assumption, Sememe Prediction with Aggregated Sememe Embeddings (SPASE) model is proposed. SPASE is also based on matrix factorization, and is formally denoted as

$$ w_i = \sum_{x_j \in X_{w_i}} M'_{ij} x_j, \quad (6.26) $$

where $X_{w_i}$ is the sememe set of the word $w_i$ and $M'_{ij}$ represents the weight of sememe $x_j$ for word $w_i$, which only has value on nonzero elements of word-sememe labeled matrix $M$. To learn sememe embeddings, we attempt to decompose the word embedding matrix $V$ into $M'$ and sememe embedding matrix $X$, with pretrained word embeddings fixed during training, which could also be written as $V = M'X$.

The contribution of SPASE is that it complies with the definition of sememes in HowNet that sememes are the semantic components of words. In SPASE, each sememe can be regarded as a tiny semantic unit, and all words can be represented by composing several semantic units, i.e., sememes, which make up an interesting semantic regularity. However, SPASE is difficult to train because word embeddings are fixed, and the number of words is much larger than the number of sememes. In the case of modeling complex semantic compositions of sememes into words, the representation capability of SPASE may be strongly constrained by limited parameters of sememe embeddings and excessive simplification of additive assumption.

### 6.3.4.4 Lexical Sememe Prediction with Internal Information

In the previous section, we introduce the automatic lexical sememe prediction proposed by Xie et al. [66]. These methods ignore the internal information within words (e.g., the characters in Chinese words), which is also significant for word understanding, especially for words which are of low frequency or do not appear in the corpus
6.3 Applications

Wors embedding

External information

Internal information

HostOf Relate To domain

word

sense

sememe

Fig. 6.8 Sememes of the word 铁匠 (ironsmith) in HowNet, where occupation, human, and industrial can be inferred by both external (contexts) and internal (characters) information, while metal is well-captured only by the internal information within the character 铁 (iron)

at all. In this section, we introduce the work of Jin et al. [30], which takes Chinese as an example and explores methods of taking full advantage of both external and internal information of words for sememe prediction.

In Chinese, words are composed of one or multiple characters, and most characters have corresponding semantic meanings. As shown by [67], more than 90% of Chinese characters in modern Chinese corpora are morphemes. Chinese words can be divided into single-morpheme words and compound words, where compound words account for a dominant proportion. The meanings of compound words are closely related to their internal characters as shown in Fig. 6.8. Taking a compound word 铁匠 (ironsmith), for instance, it consists of two Chinese characters: 铁 (iron) and 匠 (craftsman), and the semantic meaning of 铁匠 can be inferred from the combination of its two characters (iron + craftsman → ironsmith). Even for some single-morpheme words, their semantic meanings may also be deduced from their characters. For example, both characters of the single-morpheme word 徘徊 (hover) represent the meaning of hover or linger. Therefore, it is intuitive to take the internal character information into consideration for sememe prediction.

Reference [30] proposes a novel framework for Character-enhanced Sememe Prediction (CSP), which leverages both internal character information and external context for sememe prediction. CSP predicts the sememe candidates for a target word from its word embedding and the corresponding character embeddings. Specifically, following SPWE and SPSE as introduced by [66] to model external information, Sememe Prediction with Word-to-Character Filtering (SPWCF) and Sememe Prediction with Character and Sememe Embeddings (SPCSE) are proposed to model internal character information.

Sememe Prediction with Word-to-Character Filtering. Inspired by collaborative filtering [58], Jin et al. [30] propose to recommend sememes for an unlabeled word according to its similar words based on internal information. And words are considered as similar if they contain the same characters at the same positions.
In Chinese, the meaning of a character may vary according to its position within a word [14]. Three positions within a word are considered: Begin, Middle, and End. For example, as shown in Fig. 6.9, the character at the Begin position of the word 火车站 (railway station) is 火 (fire), while 车 (vehicle) and 站 (station) are at the Middle and End position, respectively. The character 站 usually means station when it is at the End position, while it usually means stand at the Begin position like in 站立 (stand), 站岗哨兵 (standing guard), and 站起来 (stand up).

Formally, for a word \( w = c_1c_2...c_{|w|} \), we define \( \pi_B(w) = \{c_1\} \), \( \pi_M(w) = \{c_2, ..., c_{|w|-1}\} \), \( \pi_E(w) = \{c_{|w|}\} \), and

\[
P_p(x_j|c) \sim \frac{\sum_{w_i \in W \land c \in \pi_p(w_i)} M_{ij}}{\sum_{w_i \in W \land c \in \pi_p(w_i)} |X_{w_i}|},
\]

that represents the score of a sememe \( x_j \) given a character \( c \) and a position \( p \), where \( \pi_p \) may be \( \pi_B \), \( \pi_M \), or \( \pi_E \). \( M \) is the same matrix used in SPWE. Finally, the score function \( P(x_j|w) \) of sememe \( x_j \) given a word \( w \) is defined as

\[
P(x_j|w) \sim \sum_{p \in \{B,M,E\}} \sum_{c \in \pi_p(w)} P_p(x_j|c).
\]
Fig. 6.10 An example of adopting multiple-prototype character embeddings. The numbers are the cosine distances. The sememe 金属 (metal) is the closest to one embedding of 铁 (iron) and even low-frequency words are usually composed of common characters, it is feasible to use pretrained character embeddings to represent rare words. During factorizing of the word-sememe matrix, the character embeddings are fixed.

$N_c$ is set as the number of embeddings for each character, and each character $c$ has $N_c$ embeddings $c^1, ..., c^{N_c}$. Given a word $w$ and a sememe $x$, the embedding of a character of $w$ closest to the sememe embedding by cosine distance is selected as the representation of the word $w$, as shown in Fig. 6.10. Specifically, given a word $w = c_1...c_{|w|}$ and a sememe $x_j$, we define

$$k^*, r^* = \arg \min_{k, r} \left[ 1 - \cos(c_r^k, x_j' + \bar{x}_j') \right],$$  

(6.29)

where $k^*$ and $r^*$ indicate the indices of the character and its embedding closest to the sememe $x_j$ in the semantic space. With the same word-sememe matrix $M$ and sememe-sememe correlation matrix $C$ in SPSE, the sememe embeddings are learned with the loss function:

$$\mathcal{L} = \sum_{w_i \in W, x_j \in X} (c_k^{r^*} \cdot (x'_j + \bar{x}_j') + b^{r^*}_k + b^{r^*}_j - M_{ij})^2 + \lambda' \sum_{x_j, x_q \in X} (x'_j \cdot \bar{x}_q' - C_{jq})^2,$$

(6.30)

where $x'_j$ and $\bar{x}_j'$ are the sememe embeddings for sememe $x_j$, and $c_k^{r^*}$ is the embedding of the character that is the closest to sememe $x_j$ within $w_i$. Note that, as the characters and the words are not embedded into the same semantic space, new sememe embeddings are learned instead of using those learned in SPSE, hence different notations are used for the sake of distinction. $b^r_k$ and $b^r_j$ denote the biases of $c_k$ and $x_j$, and $\lambda'$ is the hyperparameter adjusting the two parts. Finally, the score function of word $w = c_1...c_{|w|}$ is defined as

$$P(x_j|w) \sim c_k^{r^*} \cdot (x'_j + \bar{x}_j').$$  

(6.31)
Model Ensembling. SPWCF/SPCSE and SPWE/SPSE take different sources of information as input, which means that they have different characteristics: SPWCF/SPCSE only have access to internal information, while SPWE/SPSE can only make use of external information. On the other hand, just like the difference between SPWE and SPSE, SPWCF originates from collaborative filtering, whereas SPCSE uses matrix factorization. All of those methods have in common that they tend to recommend the sememes of similar words, but they diverge in their interpretation of similar.

Therefore, to obtain better prediction performance, it is necessary to combine these models. We denote the ensemble of SPWCF and SPCSE as the internal model, and the ensemble of SPWE and SPSE as the external model. The ensemble of the internal and the external models is the novel framework CSP. In practice, for words with reliable word embeddings, i.e., high-frequency words, we can use the integration of the internal and the external models; for words with extremely low frequencies (e.g., having no reliable word embeddings), we can just use the internal model and ignore the external model, because the external information is noisy in this case. Figure 6.11 shows model ensembling in different scenarios. For the sake of comparison, we use the integration of SPWCF, SPCSE, SPWE, and SPSE as CSP in all experiments. And two models are integrated by simple weighted addition.

6.3.4.5 Cross-Lingual Sememe Prediction

Most languages do not have sememe-based linguistic KBs such as HowNet, which prevents us from understanding and utilizing human languages to a greater extent. Therefore, it is important to build sememe-based linguistic KBs for various languages.

To address the issue of the high labor cost of manual annotation, Qi et al. [56] propose a new task, cross-lingual lexical sememe prediction (CLSP) which aims to
automatically predict lexical sememes for words in other languages. There are two critical challenges for CLSP:

(1) There is not a consistent one-to-one match between words in different languages. For example, English word “beautiful” can refer to Chinese words of either 美丽 or 漂亮. Hence, we cannot simply translate HowNet into another language. And how to recognize the semantic meaning of a word in other languages becomes a critical problem.

(2) Since there is a gap between the semantic meanings of words and sememes, we need to build semantic representations for words and sememes to capture the semantic relatedness between them.

To tackle these challenges, Qi et al. [56] propose a novel model for CLSP, which aims to transfer sememe-based linguistic KBs from source language to target language. Their model contains three modules: (1) monolingual word embedding learning which is intended for learning semantic representations of words for source and target languages, respectively; (2) cross-lingual word embedding alignment which aims to bridge the gap between the semantic representations of words in two languages; (3) sememe-based word embedding learning whose objective is to incorporate sememe information into word representations.

They take Chinese as source language and English as the target language to show the effectiveness of their model. Experimental results show that the proposed model could effectively predict lexical sememes for words with different frequencies in other languages and their model has consistent improvements on two auxiliary experiments including bilingual lexicon induction and monolingual word similarity computation by jointly learning the representations of sememes, words in source and target languages.

The model consists of three parts: monolingual word representation learning, cross-lingual word embedding alignment, and sememe-based word representation learning. Hence, they define the objective function of our method corresponding to the three parts:

$$L = L_{\text{mono}} + L_{\text{cross}} + L_{\text{sememe}}.$$  \hspace{1cm} (6.32)

Here, the monolingual term $L_{\text{mono}}$ is designed for learning monolingual word embeddings from nonparallel corpora for source and target languages, respectively. The cross-lingual term $L_{\text{cross}}$ aims to align cross-lingual word embeddings in a unified semantic space. And $L_{\text{sememe}}$ can draw sememe information into word representation learning and conduct to better word embeddings for sememe prediction. In the following paragraphs, we will introduce the three parts in detail.

**Monolingual Word Representation.** Monolingual word representation is responsible for explaining regularities in monolingual corpora of source and target languages. Since the two corpora are nonparallel, $L_{\text{mono}}$ comprises two monolingual submodels that are independent of each other:

$$L_{\text{mono}} = L_{\text{mono}}^{S} + L_{\text{mono}}^{T},$$  \hspace{1cm} (6.33)

where the superscripts $S$ and $T$ denote source and target languages, respectively.
As a common practice, the well-established Skip-gram model is chosen to obtain monolingual word embeddings. The Skip-gram model is aimed at maximizing the predictive probability of context words conditioned on the centered word. Formally, taking the source side, for example, given a training word sequence \( \{ w_s^1, \ldots, w_s^n \} \), Skip-gram model intends to minimize

\[
\mathcal{L}_{\text{mono}}^S = - \sum_{c=K+1}^{n-K} \sum_{-K \leq k \leq K, k \neq 0} \log P(w_c^{S+k} | w_c^S), \tag{6.34}
\]

where \( K \) is the size of the sliding window. \( P(w_c^{S+k} | w_c^S) \) stands for the predictive probability of one of the context words conditioned on the centered word \( w_c^S \), formalized by the following softmax function:

\[
P(w_c^{S+k} | w_c^S) = \frac{\exp(\mathbf{w}_c^{S+k} \cdot \mathbf{w}_c^S)}{\sum_{\mathbf{w}_s^s \in V^s} \exp(\mathbf{w}_s^s \cdot \mathbf{w}_c^S)}, \tag{6.35}
\]

in which \( V^s \) indicates the word vocabulary of source language. \( \mathcal{L}_{\text{mono}}^T \) can be formulated similarly.

**Cross-lingual Word Embedding Alignment.** Cross-lingual word embedding alignment aims to build a unified semantic space for the words in source and target languages. Inspired by [69], the cross-lingual word embeddings are aligned with signals of a seed lexicon and self-matching.

Formally, \( \mathcal{L}_{\text{cross}} \) is composed of two terms including alignment by seed lexicon \( \mathcal{L}_{\text{seed}} \) and alignment by matching \( \mathcal{L}_{\text{match}} \):

\[
\mathcal{L}_{\text{cross}} = \lambda_s \mathcal{L}_{\text{seed}} + \lambda_m \mathcal{L}_{\text{match}}, \tag{6.36}
\]

where \( \lambda_s \) and \( \lambda_m \) are hyperparameters for controlling relative weightings of the two terms.

1. **Alignment by Seed Lexicon**

The seed lexicon term \( \mathcal{L}_{\text{seed}} \) encourages word embeddings of translation pairs in a seed lexicon \( \mathcal{D} \) to be close, which can be achieved via an \( L_2 \) regularizer:

\[
\mathcal{L}_{\text{seed}} = \sum_{(w_s^S, w_t^T) \in \mathcal{D}} \| w_s^S - w_t^T \|^2, \tag{6.37}
\]

in which \( w_s^S \) and \( w_t^T \) indicate the words in source and target languages in the seed lexicon, respectively.

2. **Alignment by Matching Mechanism**

As for the matching process, it is found on the assumption that each target word should be matched to a single source word or a special empty word, and vice versa. The goal of the matching process is to find the matched source (target) word for each
target (source) word and maximize the matching probabilities for all the matched word pairs. The loss of this part can be formulated as

$$L_{\text{match}} = L_{T2S} + L_{S2T},$$  \hspace{1cm} (6.38)

where $L_{\text{match}}$ is the term for target-to-source matching and $L_{\text{match}}$ is the term for source-to-target matching.

Next, a detailed explanation of target-to-source matching is given, and the source-to-target matching is defined in the same way. A latent variable $m_t \in \{0, 1, \ldots, |V_S|\}$ ($t = 1, 2, \ldots, |V_T|$) is first introduced for each target word $w^T_t$, where $|V_S|$ and $|V_T|$ indicate the vocabulary size of source and target languages, respectively. Here, $m_t$ specifies the index of the source word that $w^T_t$ matches with, and $m_t = 0$ signifies the empty word is matched. Then we have $m = \{m_1, m_2, \ldots, m_{|V_T|}\}$, and can formalize the target-to-source matching term:

$$L_{T2S} = -\log P(C^T|C^S) = -\log \sum_m P(C^T, m|C^S),$$  \hspace{1cm} (6.39)

where $C^T$ and $C^S$ denote the target and source corpus, respectively. Here, they simply assume that the matching processes of target words are independent of each other. Therefore, we have

$$P(C^T, m|C^S) = \prod_{w^T \in C^T} P(w^T, m|C^S) = \prod_{t=1}^{|V_T|} P(w^T_t | w^S_{m_t}) c(w^T_t),$$  \hspace{1cm} (6.40)

where $w^S_{m_t}$ is the source word that $w^T_t$ matches with, and $c(w^T_t)$ is the number of times $w^T_t$ occurs in the target corpus.

### 6.3.5 Other Sememe-Guided Applications

#### 6.3.5.1 Chinese LIWC Lexicon Expansion

Linguistic Inquiry and Word Count (LIWC) [52] has been widely used for computerized text analysis in social science. Not only can LIWC be used to analyze text for classification and prediction, but it has also been used to examine the underlying psychological states of a writer or speaker. In the beginning, LIWC was developed to address content analytic issues in experimental psychology. Nowadays, there is an increasing number of applications across fields such as computational linguistics [22], demographics [48], health diagnostics [11], and social relationship [32].

Chinese is the most spoken language in the world, but we cannot use the original LIWC to analyze Chinese text. Fortunately, Chinese LIWC [28] has been released
to fill the vacancy. In this part, we mainly focus on Chinese LIWC and using LIWC to stand for Chinese LIWC if not otherwise specified.

While LIWC has been used in a variety of fields, its lexicon only contains less than 7,000 words. This is insufficient because according to [39], there are at least 56,008 common words in Chinese. Moreover, LIWC lexicon does not consider emerging words and phrases on the Internet. Therefore, it is reasonable and necessary to expand the LIWC lexicon so that it is more accurate and comprehensive for scientific research. One way to expand LIWC lexicon is to annotate the new words manually. However, it is too time-consuming and often requires language expertise to add new words. Hence, expanding LIWC lexicon automatically is proposed.

In LIWC lexicon, words are labeled with different categories and categories form a certain hierarchy. Therefore, hierarchical classification algorithms can be naturally applied to LIWC lexicon. Reference [15] proposes Hierarchical SVM (Support Vector Machine), which is a modified version of SVM based on the hierarchical problem decomposition approach. In [6], the authors presented a novel algorithm which can be used on both tree- and Directed Acyclic Graph (DAG)-structured hierarchies. Some recent works [12, 33] attempted to use neural networks in the hierarchical classification.

However, these methods are often too generic without considering the special properties of words and LIWC lexicon. Many words and phrases have multiple meanings and are thereby classified into multiple leaf categories. This is often referred to as polysemy. Additionally, many categories in LIWC are fine-grained, thus making it more difficult to distinguish them. To address these issues, we introduce several models to incorporate sememe information when expanding the lexicon, which will be discussed after the introduction of the basic model.

Basic Decoder for Hierarchical Classification. First, we introduce the basic model for Chinese LIWC lexicon expansion. The well-known Sequence-to-Sequence decoder [64] is exploited for hierarchical classification. The original Sequence-to-Sequence decoder is often trained to predict the next word $w_t$ with consideration of all the previously predicted words {$w_1, \ldots, w_{t-1}$}. This is a useful feature since an important difference between flat multilabel classification and hierarchical classification is that there are explicit connections among hierarchical labels. This property is utilized by transforming hierarchical labels into a sequence. Let $Y$ denote the label set and $\pi: Y \rightarrow Y$ denote the parent relationship where $\pi(y)$ is the parent node of $y \in Y$. Given a word $w$, its labels form a tree structure hierarchy. We then choose each path from the root node to the leaf node, and transform it into a sequence {$y_1, y_2, \ldots, y_L$} where $\pi(y_i) = y_{i-1}$, $\forall i \in [2, L]$ and $L$ is the number of levels in the hierarchy. In this way, when the model predicts a label $y_i$, it takes into consideration the probability of parent label sequence {$y_1, \ldots, y_{i-1}$}. Formally, the decoder defines a probability over the label sequence:

$$P(y_1, y_2, \ldots, y_L) = \prod_{i=1}^{L} P(y_i| (y_1, \ldots, y_{i-1}), w). \quad (6.41)$$
A common approach for decoder is to use LSTM [27] so that each conditional probability is computed as

\[
P(y_i|(y_1, \ldots, y_{i-1}), w) = g(y_{i-1}, s_i) = o_i \odot \tanh(h_i),
\]  

(6.42)

where

\[
\begin{align*}
h_i &= f_i \odot h_{i-1} + i_i \odot \tilde{h}_i, \\
\tilde{h}_i &= \tanh(W_h[h_{i-1}; y_{i-1}] + b_h), \\
o_i &= \text{Sigmoid}(W_o[h_{i-1}; y_{i-1}] + b_o), \\
z_i &= \text{Sigmoid}(W_z[h_{i-1}; y_{i-1}] + b_z), \\
f_i &= \text{Sigmoid}(W_f[h_{i-1}; y_{i-1}] + b_f),
\end{align*}
\]  

(6.43)

where \(\odot\) is an element-wise multiplication and \(h_i\) is the \(i\)th hidden state of the RNN. \(W_h, W_o, W_z, W_f\) are weights and \(b_h, b_o, b_z, b_f\) are biases. \(o_i, z_i,\) and \(f_i\) are known as output gate layer, input gate layer, and forget gate layer, respectively.

To take advantage of word embeddings, the initial state \(h_0 = w\) is defined where \(w\) represents the embedding of the word. In other words, the word embeddings are applied as the initial state of the decoder.

Specifically, the inputs of our model are word embeddings and label embeddings. First, raw words are transformed into word embeddings by an embedding matrix \(E \in \mathbb{R}^{|V| \times dw}\), where \(dw\) is the word embedding dimension. Then, at each time step, label embeddings \(y\) are fed to the model, which is obtained by a label embedding matrix \(Y \in \mathbb{R}^{|Y| \times dy}\), where \(dy\) is the label embedding dimension. Here word embeddings are pretrained and fixed during training.

Generally speaking, the decoder is expected to decode word labels hierarchically based on word embeddings. At each time step, the decoder will predict the current label depending on previously predicted labels.

**Hierarchical Decoder with Sememe Attention.** The basic decoder uses word embeddings as the initial state, then predicts word labels hierarchically as sequences. However, each word in the basic decoder model has only one representation. This is insufficient because many words are polysemous and many categories are fine-grained in the LIWC lexicon. It is difficult to handle these properties using a single real-valued vector. Therefore, Zeng et al. [68] propose to incorporate sememe information.

Because different sememes represent different meanings of a word, they should have different weights when predicting word labels. Moreover, we believe that the same sememe should have different weights in different categories. Take the word *apex* in Fig. 6.12, for example. The sememe *location* should have a relatively higher weight when the decoder chooses among the subclasses of *relative*. When choosing among the subclasses of *PersonalConcerns*, *location* should have a lower weight because it represents a relatively irrelevant sense *vertex*. 
To achieve these goals, the utilization of attention mechanism [1] is proposed to incorporate sememe information when decoding the word label sequence. The structure of the model is illustrated in Fig. 6.13.

Similar to the basic decoder approach, word embeddings are applied as the initial state of the decoder. The primary difference is that the conditional probability is defined as

$$P(y_i | (y_1, \ldots, y_{i-1}), w, c_i) = g([y_{i-1}; c_i], h_i),$$

(6.44)

where $c_i$ is known as context vector. The context vector $c_i$ depends on a set of sememe embeddings $\{x_1, \ldots, x_N\}$, acquired by a sememe embedding matrix $X \in \mathbb{R}^{S \times ds}$, where $ds$ is the sememe embedding dimension.

To be more specific, the context vector $c_i$ is computed as a weighted sum of the sememe embedding $x_j$:

$$c_i = \sum_{j=1}^{N} \alpha_{ij} x_j.$$  

(6.45)

The weight $\alpha_{ij}$ of each sememe embedding $x_j$ is defined as

$$\alpha_{ij} = \frac{\exp(v \cdot \tanh(W_1 y_{i-1} + W_2 x_j))}{\sum_{k=1}^{N} \exp(v \cdot \tanh(W_1 y_{i-1} + W_2 x_k))},$$

(6.46)
where $\mathbf{v} \in \mathbb{R}^a$ is a trainable parameter, $\mathbf{W}_1 \in \mathbb{R}^{a \times d_I}$ and $\mathbf{W}_2 \in \mathbb{R}^{a \times d_s}$ are weight matrices, and $a$ is the number of hidden units in attention model.

Intuitively, at each time step, the decoder chooses which sememes to pay attention to when predicting the current word label. In this way, different sememes can have different weights, and the same sememe can have different weights in different categories. With the support of sememe attention, the decoder can differentiate multiple meanings in a word and the fine-grained categories and thus can expand a more accurate and comprehensive lexicon.

### 6.4 Summary

In this chapter, we first give an introduction to the most well-known sememe knowledge base, HowNet, which uses about 2,000 predefined sememes to annotate over 100,000 Chinese and English words and phrases. Different from other linguistic knowledge bases like WordNet, HowNet is based on the minimum semantics units (sememes) and captures the compositional relations between sememes and words.

To learn the representations of sememe knowledge, we elaborate on three models, namely Simple Sememe Aggregation model (SSA), Sememe Attention over Context model (SAC), and Sememe Attention over Target model (SAT). These models not only learn the representations of sememes but also help improve the representations of words. Next, we describe some applications of sememe knowledge, including word...
representation, semantic composition, and language modeling. We also detail how to automatically predict sememes for both monolingual and cross-lingual unannotated words.

For further learning of sememe knowledge-based NLP, you can read the book written by the authors of HowNet [18]. You can also find more related papers in this paper list https://github.com/thunlp/SCPapers. You can use the open source API OpenHowNet https://github.com/thunlp/OpenHowNet to access HowNet data.

In the future, there are some research directions worth exploring:

(1) **Utilizing Structures of Sememe Annotations.** The sememe annotations in HowNet are hierarchical, and sememes annotated to a word are actually organized as a tree. However, existing studies still do not utilize the structural information of sememes. Instead, in current methods, sememes are simply regarded as semantic labels. In fact, the structures of sememes also incorporate abundant semantic information and will be helpful to the deep understanding of lexical semantics. Besides, existing sememe prediction studies also predict unstructured sememes only, and it is an interesting task to conduct structured sememe predictions.

(2) **Leveraging Sememes in Low-data Regimes.** One of the most important and typical characteristics of sememes is that limited sememes can represent unlimited semantics, which can play an important and positive role in tackling the low-data regimes. In word representation learning, the representations of low-frequency words can be improved by their sememes, which have been well learned with the high-frequency words they annotate. We believe sememes will be beneficial to other low-data regimes, e.g., low-resource language NLP tasks.

(3) **Building Sememe Knowledge Bases for Other Languages.** Original HowNet annotates sememes for only two languages: Chinese and English. As far as we know, there are not sememe knowledge bases like HowNet in other languages. Since HowNet and its sememe knowledge have been verified helpful for better understanding human languages, it will be of great significance to annotate sememes for words and phrases in other languages. In the section, we have described a study on cross-lingual sememe prediction. And we think it is promising to make efforts toward this direction.

**References**

1. Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR*, 2015.
2. Michele Banko, Vibhu O Mittal, and Michael J Witbrock. Headline generation based on statistical translation. In *Proceedings of ACL*, 2000.
3. Marco Baroni and Roberto Zamparelli. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of EMNLP*, 2010.
4. Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3(Feb):1137–1155, 2003.
5. Adam Berger and John Lafferty. Information retrieval as statistical translation. In *Proceedings of SIGIR*, 1999.
6. Wei Bi and James T Kwok. Multi-label classification on tree-and dag-structured hierarchies. In Proceedings of ICML, 2011.
7. William Blacoe and Mirella Lapata. A comparison of vector-based representations for semantic composition. In Proceedings of EMNLP-CoNLL, 2012.
8. Leonard Bloomfield. A set of postulates for the science of language. Language, 2(3):153–164, 1926.
9. Thorsten Brants, Ashok C Popat, Peng Xu, Franz J Och, and Jeffrey Dean. Large language models in machine translation. In Proceedings of EMNLP, 2007.
10. Peter F Brown, John Cocke, Stephen A Della Pietra, Vincent J Della Pietra, Fredrick Jelinek, John D Lafferty, Robert L Mercer, and Paul S Roossin. A statistical approach to machine translation. Computational Linguistics, 16(2):79–85, 1990.
11. Wilma Bucci and Bernard Maskit. Building a weighted dictionary for referential activity. Computing Attitude and Affect in Text, pages 49–60, 2005.
12. Ricardo Cerri, Rodrigo C Barros, and André CPLF De Carvalho. Hierarchical multi-label classification using local neural networks. Journal of Computer and System Sciences, 80(1):39–56, 2014.
13. Xinxiong Chen, Zhiyuan Liu, and Maosong Sun. A unified model for word sense representation and disambiguation. In Proceedings of EMNLP, 2014.
14. Xinxiong Chen, Lei Xu, Zhiyuan Liu, Maosong Sun, and Huanbo Luan. Joint learning of character and word embeddings. In Proceedings of IJCAI, 2015.
15. Yangchi Chen, Melba M Crawford, and Joydeep Ghosh. Integrating support vector machines in a hierarchical output space decomposition framework. In Proceedings of ICARSS, 2004.
16. Lei Dang and Lei Zhang. Method of discriminant for chinese sentence sentiment orientation based on hownet. Application Research of Computers, 4:43, 2010.
17. Zhendong Dong and Qiang Dong. Hownet-a hybrid language and knowledge resource. In Proceedings of NLP-KE, 2003.
18. Zhendong Dong and Qiang Dong. HowNet and the Computation of Meaning (With CD-Rom). World Scientific, 2006.
19. Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. Retrofitting word vectors to semantic lexicons. In Proceedings of NAACL-HLT, 2015.
20. Xianghua Fu, Guo Liu, Yanyan Guo, and Zhiqiang Wang. Multi-aspect sentiment analysis for chinese online social reviews based on topic modeling and hownet lexicon. Knowledge-Based Systems, 37:186–195, 2013.
21. Edward Grefenstette and Mehrnoosh Sadrzadeh. Experimental support for a categorical compositional directional model of meaning. In Proceedings of EMNLP, 2011.
22. Justin Grimmer and Brandon M Stewart. Text as data: The promise and pitfalls of automatic content analysis methods for political texts. Political analysis, 21(3):267–297, 2013.
23. 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. In Proceedings of EMNLP, pages 4642–4651, 2018.
24. Djord Hiemstra. A linguistically motivated probabilistic model of information retrieval. In Proceedings of TPDL, 1998.
25. G. E Hinton. Products of experts. In Proceedings of ICANN, 1999.
26. Geoffrey E Hinton. Learning distributed representations of concepts. In Proceedings of CogSci, 1986.
27. Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. Neural Computation, 9(8):1735–1780, 1997.
28. Chin-Lan Huang, CK Chung, Natalie Hui, Yi-Cheng Lin, Yi-Tai Seih, WC Chen, and JW Pennbaker. The development of the chinese linguistic inquiry and word count dictionary. Chinese Journal of Psychology, 54(2):185–201, 2012.
29. Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. Improving word representations via global context and multiple word prototypes. In Proceedings of ACL, 2012.
30. Huiming Jin, Hao Zhu, Zhiyuan Liu, Ruobing Xie, Maosong Sun, Fen Lin, and Leyu Lin. Incorporating chinese characters of words for lexical sememe prediction. In Proceedings of ACL, 2018.
31. Dan Jurafsky. Speech & language processing. 2000.
32. Ewa Kacewicz, James W Pennebaker, Matthew Davis, Moongee Jeon, and Arthur C Graesser. Pronoun use reflects standings in social hierarchies. *Journal of Language and Social Psychology*, 33(2):125–143, 2014.
33. Sanjeev Kumar Karn, Ulli Waltinger, and Hinrich Schütze. End-to-end trainable attentive decoder for hierarchical entity classification. *Proceedings of EACL*, 2017.
34. Slava Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35(3):400–401, 1987.
35. Thomas Kober, Julie Weeds, Jeremy Reffin, and David Weir. Improving sparse word representations with distributional inference for semantic composition. In *Proceedings of EMNLP*, 2016.
36. Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8), 2009.
37. Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. In *Proceedings of NeurIPS*, 2014.
38. Wei Li, Xuancheng Ren, Damai Dai, Yunfang Wu, Houfeng Wang, and Xu Sun. Sememe prediction: Learning semantic knowledge from unstructured textual wiki descriptions. arXiv preprint arXiv:1808.05437, 2018.
39. Xingjian Li et al. Lexicon of common words in contemporary chinese, 2008.
40. Qun Liu. Word similarity computing based on hownet. *Computational linguistics and Chinese language processing*, 7(2):59–76, 2002.
41. Shu Liu, Jingjing Xu, Xuancheng Ren, and Xu Sun. Evaluating semantic rationality of a sentence: A sememe-word-matching neural network based on hownet. 2019.
42. Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of ACL*, 2011.
43. Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *Proceedings of ICLR*, 2013.
44. Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernocký, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Proceedings of InterSpeech*, 2010.
45. David RH Miller, Tim Leek, and Richard M Schwartz. A hidden markov model information retrieval system. In *Proceedings of SIGIR*, 1999.
46. Jeff Mitchell and Mirella Lapata. Vector-based models of semantic composition. In *Proceedings of ACL*, 2008.
47. Jeff Mitchell and Mirella Lapata. Language models based on semantic composition. In *Proceedings of EMNLP*, 2009.
48. Matthew L Newman, Carla J Groom, Lori D Handelman, and James W Pennebaker. Gender differences in language use: An analysis of 14,000 text samples. *Discourse Processes*, 45(3):211–236, 2008.
49. Yilin Niu, Ruobing Xie, Zhiyuan Liu, and Maosong Sun. Improved word representation learning with sememes. In *Proceedings of ACL*, 2017.
50. Francis Jeffry Pelletier. The principle of semantic compositionality. *Topoi*, 13(1):11–24, 1994.
51. Francis Jeffry Pelletier. *Semantic Compositionality*, volume 1. 2016.
52. James W Pennebaker, Roger J Booth, and Martha E Francis. Linguistic inquiry and word count: Liwc [computer software]. *Austin, TX: liwc.net*, 2007.
53. Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of EMNLP*, 2014.
54. Jay M Ponte and W Bruce Croft. A language modeling approach to information retrieval. In *Proceedings of SIGIR*, 1998.
55. Fanchao Qi, Junjie Huang, Chenghao Yang, Zhiyuan Liu, Xiao Chen, Qun Liu, and Maosong Sun. Modeling semantic compositionality with sememe knowledge. In *Proceedings of ACL*, 2019.
56. Fanchao Qi, Yankai Lin, Maosong Sun, Hao Zhu, Ruobing Xie, and Zhiyuan Liu. Cross-lingual lexical sememe prediction. In *Proceedings of EMNLP*, 2018.
References

57. Alexander M Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. In Proceedings of EMNLP, 2015.

58. Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In Proceedings of WWW, 2001.

59. Richard Socher, John Bauer, Christopher D Manning, and Andrew Y Ng. Parsing with compositional vector grammars. In Proceedings of ACL, 2013.

60. Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. Semantic compositionality through recursive matrix-vector spaces. In Proceedings of EMNLP-CoNLL, 2012.

61. Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D. Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In Proceedings of EMNLP, 2013.

62. Jingguang Sun, Dongfeng Cai, Dexin Lv, and Yanju Dong. Hownet based chinese question automatic classification. Journal of Chinese Information Processing, 21(1):90–95, 2007.

63. Maosong Sun and Xinxiong Chen. Embedding for words and word senses based on human annotated knowledge base: A case study on hownet. Journal of Chinese Information Processing, 30:1–6, 2016.

64. Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In Proceedings of NeurIPS, 2014.

65. David Weir, Julie Weeds, Jeremy Reffin, and Thomas Kober. Aligning packed dependency trees: a theory of composition for distributional semantics. Computational Linguistics, 42(4):727–761, December 2016.

66. Ruobing Xie, Xingchi Yuan, Zhiyuan Liu, and Maosong Sun. Lexical sememe prediction via word embeddings and matrix factorization. In Proceedings of IJCAI, 2017.

67. Binyong Yin. Quantitative research on Chinese morphemes. Studies of the Chinese Language, 5:338–347, 1984.

68. Xiangkai Zeng, Cheng Yang, Cunchao Tu, Zhiyuan Liu, and Maosong Sun. Chinese liwc lexicon expansion via hierarchical classification of word embeddings with sememe attention. In Proceedings of AAAI, 2018.

69. Meng Zhang, Haoruo Peng, Yang Liu, Huan-Bo Luan, and Maosong Sun. Bilingual lexicon induction from non-parallel data with minimal supervision. In Proceedings of AAAI, 2017.

70. Yuntao Zhang, Ling Gong, and Yongcheng Wang. Chinese word sense disambiguation using hownet. In Proceedings of ICNC.

71. Yu Zhao, Zhiyuan Liu, and Maosong Sun. Phrase type sensitive tensor indexing model for semantic composition. In Proceedings of AAAI, 2015.

72. Xiaodan Zhu, Parinaz Sobhani, and Hongyu Guo. DAG-Structured long short-term memory for semantic compositionality. In Proceedings of NAACL-HLT, 2016.

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter’s Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter’s Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.