Exploring Word Composition Knowledge in Language Usages

Yuchen Han[1], Tianyuan Liu[1], Yuqing Sun[1](✉), Tian Huang[1], Huiqian Wu[1], and Shengjun Wu[2]

[1] Shandong University, Jinan 250000, China
{hanyc,huangtian}@mail.sdu.edu.cn, sun_yuqing@sdu.edu.cn
[2] Air Force Medical University, Xi'an, China
wushj@fmmu.edu.cn

Abstract. The creativity of language is a distinct feature that sets humans apart from machines and animals, where the flexibility of word composition is the fundamental part. The patterns on words that are systematically linked together are acknowledged as the word composition knowledge. We explore this knowledge by combining the syntax information with word semantics and verify it through a series of empirical experiments on multiple datasets. From the linguistic perspective, we found that this knowledge can find the appropriate alternatives for the given phrase and generate high-quality paraphrases that satisfy both the syntax soundness and the semantic consistency with the original text. We also verify it on the questionnaires in psychological testing and find the abnormal patterns on the language usages. Compared to the large pre-trained models, this method is much more training-economic and can generate the paraphrases in an explainable way.

Keywords: Word Composition · Paraphrase · Semantic Inference

1 Introduction

The creativity of language is a distinct feature that sets humans away from machines and animals, namely expressing the same idea in totally different forms [3]. Before using sentences, in the beginning, a baby learns language from single words and gradually uses short phrases by compositing words together [6]. The flexible and diverse combinative forms of words are regarded as a milestone in using language since a baby can express many informative meanings by relatively simple words. Besides, the word composition shows the evidences of preliminary syntax, such as some nouns tending to follow a few specific verbs. Inspired by these observations, we investigate the word composition knowledge in this paper.

Y. Han and T. Liu—Contribute equally to this work.

This work was supported by the NSFC (62376138) and NSFC of Shandong (ZR2022LZH007).

C. Cao et al. (Eds.): KSEM 2024, LNAI 14887, pp. 61–72, 2024.
https://doi.org/10.1007/978-981-97-5501-1_5

There are quite a few pre-training models on learning such linguistic knowledge [8]. The widely adopted word embeddings are the semantic knowledge about words [15], which are learned based on the distributed hypothesis, namely the words often appearing in the same context are modeled as the closer vectors than others [7,16]. These embeddings have appealing, intuitive interpretations. For example, the positional relationships between vectors represent word semantic relations such as synonyms, antonyms, etc., as well as complex relationships like gender-dependent job names. Some syntax-incorporated methods learn the word embeddings by using the dependency parsing tree as the contexts instead of the word sequence [9]. The syntax-aware method uses the vectors of a pre-trained syntax parser as the supplements to pre-trained word embeddings for downstream tasks [20]. But in these models, the meanings associated with words are in an isolated manner and do not present the way on word composition.

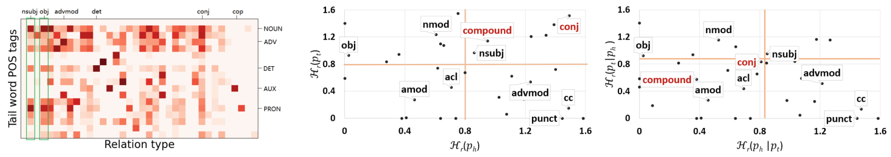
To improve the context-aware semantics, pre-trained language models encode a sentence as the sequence of vectors [4]. These vectors can be used to predict some syntax tasks such as part-of-speech (POS) or dependency relationship [1], which illustrate that the pre-trained text encoders capture some implicit syntax in sentences. But there is not an explicit form of word composition knowledge such that they can not guide the composition of multiple isolated words together.

According to linguistic knowledge, the patterns on words that are systematically linked together are acknowledged as the theory of word syntagmatic and paradigmatic relations [2,10], where words are the basic elements and the syntax is responsible for associations. Based on this theory, we explore the word composition knowledge in language usages by taking into count word semantics and syntax information. This knowledge is learned from a dependency parsing treebank, which is formed as a set of matching-check functions that can distinguish the proper word compositions from the improper ones. This method is economic in computation compared with delicately designed models that are trained with an amount of data [5].

To verify the learned knowledge, we conducted experiments on multiple datasets. The word composition task finds an appropriate alternative for a given phrase, and the paraphrasing task generates different samples with controlled syntax, which are used for data augmentation to enhance the performance of downstream tasks. We also verify it on the questionnaires in psychological testing and find abnormal patterns on the language usages. In the rest of the paper, we will present the details of our method and the experiments.

2 Exploring Word Composition Patterns

To understand the word composition patterns, we analyze the word usages in sentences. Considering the patterns on words are too sparse, we first analyze the part-of-speech (POS) of words and try to reveal the common traits. We adopt the Universal Dependency Treebank as the syntax parsing annotation since it is recognized as high quality [14]. Each dependency relation is in the form of (h, r, t), i.e. the head (dominating) word h and the tail word t are combined by the relation r. As the statistics shown in Fig. 1, there are obvious differences in the word combinations for different relations, such as *nsubj* and *obj* prefer NOUN and PRON as tail words than others, while *amod* prefers ADJ and *advmod* prefers ADV words.



(a) POS counts for tail words against relations (b) Entropy on POS on different relations (c) Conditional entropy on POS tags

Fig. 1. The statistics on word composition.

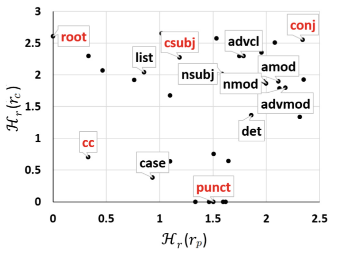
We further analyze the patterns for each relation. Given an instance (h, r, t), the POS tag of the head or tail word is denoted by p_h and p_t, respectively. We adopt the entropy[1] $\mathcal{H}_r(p_h)$ to quantify the stability of head word for r, as well as $\mathcal{H}_r(p_t)$ on the tail. The higher the entropy, the more flexible choices of POS and words. As shown in Fig. 1b, the relations in the top-right quadrant have high entropies on both head and tail words, such as the *nominal subject* relation **nsubj** connecting different POS of word in a sentence, i.e. a NOUN, a VERB, an ADJ or a PRON word. For the relations in the bottom-left quadrant, the POSs of its connected words are stable, such as the *adjectival modifier* **amod** often connecting NOUN with ADJ words and **acl** connecting NOUN with VERB words. For the other two quadrants, POS for head and tail are different.

Table 1. Relation types based on the entropy analysis

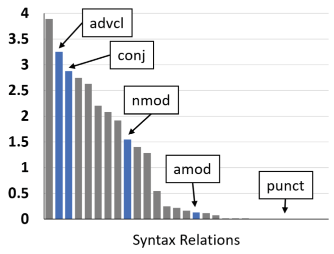
Rel. Type	Relations
Strict	*acl*, *advcl*, *amod*, *aux*, *compound*, *conj*, *det*, *mark*, *xcomp*, *dep*, *dislocated*, *expl*, *flat*, *iobj*
Free	*nsubj*, *appos*, *parataxis*
HeadFree	*advmod*, *case*, *cc*, *cop*, *nummod*, *punct*, *csubj*, *discourse*, *fixed,goeswith*, *orphan*, *vocative*
TailFree	*nmod*, *obj*, *obl*, *root*, *ccomp*, *list*, *reparandum*

Then we analyzed the composition patterns in case one word fixed by the conditional entropies, i.e. $\mathcal{H}_r(p_h|p_t)$ and $\mathcal{H}_r(p_t|p_h)$ for each relation r, show in Fig. 1c. We can see that the uncertainties are reduced for most relations, i.e. fewer relations in the top-right quadrant than before. For example, the *coordinating conjunction* relation **conj** and the *fixed phrase* **compound** change from the first quadrant to the third quadrant indicating a tighter correlation between the head and tail. By these observations, the word composition patterns are classified into four meta types: **Strict** refers to the low entropy case on both ends, **Free** for high entropy, **HeadFree** and **TailFree** are for other two cases. The concrete relations for each type are listed in Table 1.

[1] https://en.wikipedia.org/wiki/Entropy.



(a) Complexity of syntax relation usages by $H_r(r_p)$ and $\mathcal{H}_r(r_c)$



(b) Size of the dependency subtree for each syntax relation

Fig. 2. Syntax relation combination analysis

Then we analyze the complex patterns on multi-word composition that are connected by two consecutive syntax triplets (h, r_p, t) and (t, r_c, tt), where r_p is the parent relation for r_c on the dependency tree of the sentence. For each r, we estimate the diversity on how r forming a bridge between parent and child relations by the entropy $\mathcal{H}_r(r_p)$ and $\mathcal{H}_r(r_c)$. Higher entropy means more diverse of the corresponding pattern. The results in Fig. 2a show the different patterns on relations. For example, the *clause leading* relation *csubj* has diverse subsequent relations and stable pattern on the precedent. *conj* connects many types of relation in usages.

We also analyzed the size of the dependency subtree for each syntax relation, shown in Fig. 2b. For a sentence x, a subtree is a part of a dependency parsing tree including a node and its following syntax relationships. For the relations with large sizes of subtree, there are rich forms on the functional components, such as *nmod*, *conj*, and the *clause leading* relations, such as *advcl*, *csubj*. While the lower ones are mostly leading leaf nodes in the parsing tree such as *punct*.

These patterns on word and relation combinations provide the important guidance for us to effectively model the word composition knowledge in language usages.

3 The Word Composition Knowledge Model

We model word composition knowledge with the consideration of the semantic and syntactical soundness, as shown in Fig. 3. For economic computation purposes, we adopt the pre-trained word embeddings for semantics. To combine word semantics with syntax, we analyze the dependency trees and introduce the mapping function for each syntax relation r. A function $f_r(\boldsymbol{u})$ maps the embedding $\boldsymbol{u}$ of word u to the vector $\boldsymbol{u_r}$ in the syntax space. Considering the asymmetry property of syntax relation, i.e. a word is either dominating or dominated by another word, for each r, two functions are defined for head $f_r^h(\boldsymbol{u})$ and tail $f_r^t(\boldsymbol{u})$, respectively. The functions can be defined as any form of neural network, such as a non-linear transformer $\boldsymbol{u_r^\cdot} = f_r^\cdot(\boldsymbol{u}) = \boldsymbol{W_r^\cdot}\sigma(\boldsymbol{U_r^\cdot}\boldsymbol{u})$, where

$\cdot \in \{h, t\}$ represents head or tail, $\boldsymbol{W}_r^{\cdot}, \boldsymbol{U}_r^{\cdot}$ are the trainable weights and σ is an activation function.

To check whether two words u and v are compatible on r, we define the matching score ($\mathcal{S}$ for short) as the measure on (u, r, v). Similar to existing works, $\mathcal{S}$ is defined as the energy function that satisfies the correlation property, i.e. a larger score indicates the higher probability to be a common usage. To reflect the distinct patterns of four meta types on word composition in last section, different parameters are formulated for them. There are many choices for $\mathcal{S}(x)$, such as the Euclidean distance or the inner product with a bilinear function $\mathcal{S}(x) = \boldsymbol{u}_r^h \boldsymbol{M}_{r_t} \boldsymbol{v}_r^t$, where $\boldsymbol{u}_r^h = f_r^h(\boldsymbol{u})$ and $\boldsymbol{u}_r^t = f_r^t(\boldsymbol{u})$ are the vectors for u in the r space. We would verify different forms of $\mathcal{S}(x)$ in the experiments.

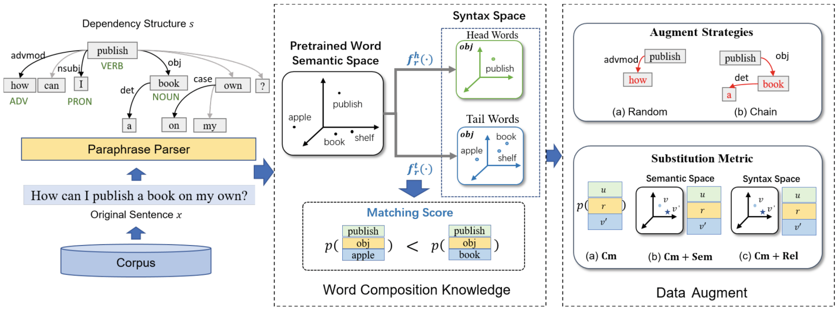


Fig. 3. The word composition knowledge model and data augment process

To be mentioned here, the function $\mathcal{S}$ is independent of specific syntax relations such that it can systemically check the soundness of word compositions. Comparatively, the mapping functions model the different traits of dependency relations. The composition knowledge is formalized as the tuple $\mathbf{CK} =< \{f_r^{\cdot}\}, \mathcal{S} >$, where $\{f_r^{\cdot}\}$ denotes the set of mapping functions.

We design two objectives for learning the word composition knowledge. One is to maximize the probabilities of existing word usages. For a word composition $x = (u, r, v)$, the probability is $p(u, r, v) \propto \exp(\mathcal{S}(x))$. This loss is the log-posterior probability over the observed samples, namely the tuples (u, r, v) in the corpus. Since the computation of the denominator of the loss is intractable, we adopt the widely used noise contrastive estimation technique (NCE)[2] and transfer the loss function to the following form ℓ_1. C is the training set, $C'(x)$ is the set of negative samples for sample x. The number of negative samples against one positive sample is denoted by K_s. Given a normal case (u, r, v), the negative samples refer to the unusual ones, which are constructed as follows: 1) reverse the head and tail (v, r, u); 2) change the syntax relation $(u, r', v), r' \in \{R - r\}$ and 3) change the head or tail $(u, r, v'), (u', r, v), u', v' \in V$.

$$\ell_1 = -\sum_{x \in C} \{\log \sigma(\mathcal{S}(x)) + \sum_{x' \in C'(x)} \log \sigma(-\mathcal{S}(x'))\} \qquad (1)$$

[2] https://scholar.sun.ac.za/handle/10019.1/98758.

The second objective ℓ_2 is to differentiate the syntax relation spaces. Namely, for two distinct relations r and $r' \neq r$, their syntax spaces are drawn away from each other so as to make the model relation-specific and avoid collapse to the trivial solution. For each positive sample, we choose K_r negative relations instead of all types to reduce the computation cost.

$$\ell_2 = -\sum_{u,v \in V, r, r' \in R} \log \mathcal{S}(x) \tag{2}$$

Let Θ denote the parameters of model, the final loss function for learning **CK** is given below, where $||\Theta||_2^2$ is a regularization, α and λ are the hyper-parameters.

$$\Theta^* = \arg\min_{\Theta}\{\ell_1 + \alpha\ell_2 + \lambda||\Theta||_2^2\} \tag{3}$$

4 Model Validation Experiments

4.1 Word Composition Predication

Given two words u, v and the syntax relation r, we predicate the relationship between u, v: namely (u, r, v), (v, r, u), or neither of them $p(\overline{u, r, v})$. With **CK**, the probabilities for $p(u, r, v)$ or $p(v, r, u)$ is computed as $p(u, r, v) = \frac{\exp S(u,v)}{\sum_{(u',r',v') \in C} \exp S(u',v')}$ $p(\overline{u, r, v}) = min[1 - p(u, r, v), 1 - p(v, r, u)]$ [13].

The pre-trained GloVe word embeddings [16] are adopted as the semantic knowledge. We use the English Web Treebank (EWT) from Universal Dependencies to extract syntax relationships as the training and testing samples. There are 200k syntax triplets in the training set and 24k triplets for testing. Our model is implemented with PyTorch 1.8.0. All experiments are conducted on an NVIDIA GeForce RTX 2070 with 8G RAM. The dimension size of word embeddings is set 300 and the dimension size of syntax space is 25. λ is set 1e-7. Other hyperparameters are set $K_s \in \{3, 6, 9, 12\}, K_r \in \{0, 1, 3, 7\}$ and $\alpha \in \{0, 1, 3, 7\}$.

The representative method on dependency based word embedding (**DepVec** for short) [8] is adopted for comparison, which uses the dependency tree context for the word semantics. It divided (u, r, v) into the center word u and the dependency context (r, v). For given u, it predicts whether (r, v) is suitable. In case of unknown cases (r', v') in the testing phase, a random vector is used. Our model is denoted by **CKP** in the experimental results. The variants of our model are also adopted for comparison. The subscript indicates the energy function, while the superscript R indicates the independent parameters for the four meta types, e.g. $\mathbf{CKP}^R_{\text{bilin}}$ for the bilinear energy function. $\mathbf{CKP}^R_{\text{bilin}}$ is for $\mathcal{S}(x) = \boldsymbol{u}_r^h \boldsymbol{M}_{r_t} \boldsymbol{v}_r^t + \boldsymbol{M}_{r_t}^h \boldsymbol{u}_r^h + \boldsymbol{M}_{r_t}^t \boldsymbol{v}_r^t + b$. We adopt the accuracy and macro precision (denoted by Mac.Pre.) as the metrics. The results are the average of 4 training and testing runs for each method.

As the results in Table 2, our model and the variants achieve better results than the baselines, which shows that the word composition knowledge learned by our model can infer the alternatives effectively. As for the form of the matching score function, the *bilinear* function shows better performances than other

Table 2. Syntax Relationship Predication Results on EWT

Methods	Accuracy	Mac. Pre.	Acc@Relation Types				High Freq. r		Low Freq. r	
			Strict	Free	Head Free	Tail Free	Stric	Head Free	Strict	Head Free
$\mathbf{CKP}^{R}_{\text{bilin}}$	**0.873**	0.834	**0.877**	**0.833**	**0.903**	**0.837**	**0.862**	**0.888**	0.967	**0.975**
$\mathbf{CKP}^{R}_{\text{biaff}}$	0.866	0.823	0.871	0.822	0.899	0.829	0.854	0.883	0.969	0.973
$\mathbf{CKP}_{\text{bilin}}$	0.87	**0.836**	0.872	0.827	**0.903**	0.837	0.855	0.888	0.968	**0.975**
$\mathbf{CKP}_{\text{biaff}}$	0.867	0.826	0.871	0.826	0.898	0.833	0.854	0.882	**0.97**	0.974
DepVec	0.564	0.487	0.567	0.489	0.651	0.463	0.541	0.631	0.718	0.745

choices. But for the relations of **TailFree**, $\mathbf{CKP}^{R}_{\text{lin}}$ with the *linear* function achieves the highest, which indicates that the energy function is the key point to model different meta types of relations. Overall, the variants with meta type dependent functions (R) are better than those with the same function.

As the results in the middle part of Table 2, we found that it is easier to predicate the word composition with the **Strict** dependency relation than **Free** or **TailFree**. Many **TailFree** relations connect the words with nominal POS such as NOUN and PRON, which associate a large vocabulary and make the prediction difficult. With no surprise, the accuracy on **Free** is the lowest. These results are consistent with the previous observations on word composition patterns. An interesting result is the higher accuracy for the **HeadFree** type. We check this type of relations and find that most relations are with low occurrences in the dataset, such as *csubj*, *discourse*, *fixed*, *goeswith*, *orphan*, *vocative*, etc. These relations have comparatively stable patterns on word compositions and thus they are easy to predict.

Table 3. The influence of K_s,K_r and α on EWT

	K_s				K_r				α			
	3	6	9	12	0	1	3	7	0	1	3	7
Acc	.869	**.873**	.867	.861	.859	.869	**.873**	.869	.860	.869	**.870**	.868
Mac.Pre	.833	.847	.855	**.861**	.813	.846	.847	**.853**	.841	**.847**	.847	.846

We then analyze how the hyper-parameters influence the performance. As shown in Table 3, the model achieves the best when K_s is 6. Since accuracy is the mainly considered metric, we do not check the cases on K_s higher than 12, although the macro precision shows an increasing tendency. As shown in column 0 for K_r and α, the performances drop significantly without the ℓ_2, which confirms the importance of differentiating syntax relation spaces. Meanwhile, increasing K_r and α shows a smaller impact on the performances than K_s.

4.2 Paraphrasing for Data Augmentation

In this section, we use the learned word composition knowledge **CK** for text paraphrasing, which can be used for data augmentation in the downstream tasks [11].

Word level replacement is a basic way for paraphrasing [17], which is lightweight and robust. For a sentence, a pre-trained dependency parser is used to generate the parsing tree. There are two steps for paraphrasing, shown in Fig. 3, which are performed multiple times.

Augmentation Strategy: 1) *Random substitutions* (**Random**) chooses multiple triplets $< u, r, v >$ for paraphrasing. 2) *Syntax chain substitutions* (**Chain**) changes consecutive words on the parsing tree to provide more fluent paraphrases. For example, a path $(h, r_p, t), (t, r, tt)$ is chosen, where t and tt are replaced.

Word Replacement: For each selected triplet (u, r, v), the new phrase (u, r, v') should be semantically consistent with the given sentence and satisfy the syntax soundness. We adopt several metrics to choose the substitution word v'. One uses the matching score $p(u, r, v')$ (**Cm** for short) as the reference. Another considers both semantics and syntax (**Cm+Sem**), i.e. $sim(v, v') \cdot p(u, r, v')$. The third considers the semantics in the syntax r spaces and soundness(**Cm+Rel**), i.e.$sim(\boldsymbol{v}_r^t, \boldsymbol{v}_r'^t) \cdot p(u, r, v')$.

Table 4. Effectiveness of Data Augmentation. Supervised for 100% data.

Methods	IMDB		SST2		SST5	
	40%	10%	40%	10%	40%	10%
Sem	92.7	91.0	91.1	88.1	51.3	48.0
BERT	92.9	91.4	91.2	88.6	52.5	49.6
SemAug	92.6	90.9	90.2	87.4	52.1	50.6
Ours	**93.1**	**91.7**	**91.8**	**89.5**	**53.5**	**51.7**
Supervised	93.2		91.9		53.0	

We adopt the text classification task and use the widely adopted datasets IMDB [12], SST-2 and SST-5 [18] for experiments. For the comparison methods, **Sem** uses the similarity of the pre-trained embeddings as the reference to choose the substitution word. **BERT** [4] uses the pre-trained masked language model to predicate the substitution. **SemAug** uses a sememe based word substitution [19]. Since the validation focuses on the effects of data augmentation, we choose the representative pre-trained language model BERT as the text classifier. We select a portion of training data for paraphrasing, i.e. $10\%, 40\%$. Then both the original data and generated paraphrases are used together for training the classifier. The accuracy is adopted as the evaluation metric.

Results. The results in Table 4 show that after data augmentation, our method achieves the best results in most settings. Compared to the methods that use semantic information for choosing substitution words, our method benefits from the word composition knowledge. Although the original performance of BERT is high, it still gets improved by our data augmentation. We also see that with only 40% training set and the augmented samples, the model achieved the same performance with the whole training set.

Then we evaluated the strategy combinations on paraphrasing and data augmentation with 10% training set on IMDB. The results listed in Table 5 show that the performances are improved with the generated samples. **Cm+Rel** on word substitution shows the advantages over other metrics with the same data augmentation strategy, illustrating the effects of syntactical similarity in paraphrasing. For data augmentation strategies, **Chain** is better than **Random**, which shows the importance of syntax correctness and the effectiveness of usages based relation choosing.

Table 5. Accuracy improvement with 10% IMDB data against baseline(80.9%). Columns are the paraphrasing metrics and rows are for data augmentation.

	Cm	**Cm+Sem**	**Cm+Rel**
Random	+2.6%	+3.1%	+3.8%
Chain	+2.2%	+3.2%	**+4.6%**

Table 6. Semantics and diversity(BS for BertScore, SB for Sel-BLEU)

	IMDB(BS↑)	SST(BS↑)	IMDB(SB↓)	SST(SB↓)
Sem	0.943	0.942	0.851	0.685
BERT	**0.952**	**0.955**	0.839	0.812
SemAug	0.861	0.860	0.982	0.803
Cm+Rel & Chain	0.949	0.946	**0.802**	**0.552**

Analyzing the Quality of Paraphrases. The effectiveness of data augmentation mainly depends on two aspects: the expression diversity of the generated samples and their semantic consistency with the original samples. We choose BertScore [21] as the metric and a higher value indicates a better quality. As the results shown in Table 6, our method is better than others except the BERT method since they are with the same origin. For the diversity, Self-BLEU [22] is chosen as the metric and lower value means higher diversity of the generated samples. Our method outperforms the compared methods

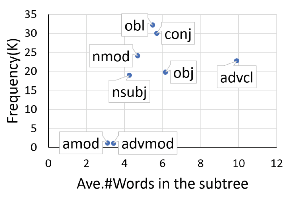
Table 7. Case study on the paraphrasing results

IMDB Sample	This movie does a great job of explaining the problems that we faced and the fears t-hat we had before we put man into space. As a history of space flight, it is still used t-oday in classrooms that can get one of the rare prints of it.
Our Paraphrase	This movie does a great job of explaining the problems that we *encount-ered ered* and fears that we had before we *stated* man into space. *Behind achronicle*chronicle of spa-ce flight, it is stillused today in classrooms that can *have* one of the rare prints of it.
SemAug Paraphrase	this *telefilm* does a *lovely assignment* of *reciting* the problems that we *counter* and t-he fears that we had before we put man into *seat*. As a *chronicle* of space flight it is st-ill *applied* today in classrooms that can *obtain* one of the *precious* prints of it.
BERT Paraphrase	movie does a great job of explaining *every fears* that we faced and the fears that we h-ad before we put man into space. *for* a history of space flight, it is still used today in classrooms that can get one of the rare prints with it.

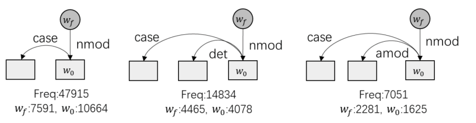
We show some examples of paraphrasing by different methods in Table 7, where the top row is chosen from the IMDB dataset. We can see the replaced word by our method has the correct syntactical roles and similar semantics with the original text. Though some replaced words might not be ideal, the paraphrase has similar sentiment polarity with the original sentence and serves as a useful augmented sample for downstream models. Comparably, the paraphrase by SemAug slightly drifts the semantics from the original one and some words have incorrect syntax roles. Some even repeats a few words, such as "every fears" repeating the non-masked "the fears".

Analyzing the Influence of Subtree on Paraphrasing. Since the key feature of the subtree is the syntactical function it takes in the sentence, we first analyzed the frequency of the parent relation r_p and the average number of words in subtrees, shown in Fig. 4a. We can see the popularity and complexities of subtrees vary greatly, e.g. the *advcl* subtrees are more complex than others since they lead a clause. The results help us choosing the appropriate subtrees in paraphrasing for maintaining the semantics consistency while satisfying the syntax soundness.

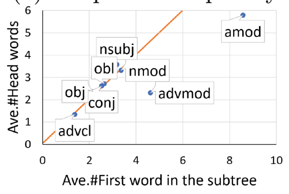
We also analyzed the semantic functionality of a subtree, where the root word w_0 is the core semantics of the structure and its head word w_p shows how the subtree modifies other words. For example, for *amod*, w_p can be "event, place, book", while the w_0 can be "blue, central, most". The average numbers for different w_0 and w_p are shown in Fig. 4c, where the orange line denotes the angle bisector. *advmod* and *amod* show lower numbers on w_0 than w_p compared to *nsubj* subtrees. Besides, we found some subtrees can be reused across different scenarios. For example, the subtrees of *nmod* and *amod* are the popular structures, shown in Fig. 4b and 4d. These subtrees share similar structures and can be considered replaceable to each other, which provide more diverse paraphrases.



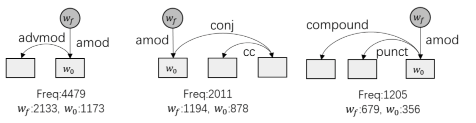
(a) Freq. and complexity



(b) The high frequent pattens on the relation *nmod*



(c) Usage diversity

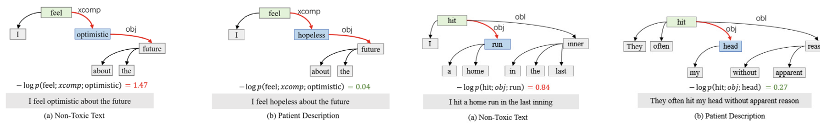


(d) The high frequent pattens on the relation *amod*

Fig. 4. Syntax and semantic functionalities

4.3 Applications on Psychological Analysis

Psychological disorders, like depression, significantly affect individuals' mental health In severe cases, it may even increase the risk of suicide. Psychological surveys play a crucial role in addressing these challenges by aiding in the early identification of potential mental health issues. Early detection supports timely diagnosis, slows the condition's progression, and helps prevent worsening symptoms. In this section, we use the learned word composition knowledge in the psychological analysis. We select some sentences from the questionnaire and analyze the patterns of word composition, shown in Fig. 5. With the learned **CK**, we can compute the probabilities of word composition patterns and find the obvious difference. For example, for the case in Fig. 5a, the probability for *(feel, xcomp, optimistic)* is higher than *(feel, xcomp, hopeless)*. Figure 5a shows an example. We can see the probability of the word composition *(feel, xcomp, optimistic)* is higher, aligning with the distribution in expressions of healthy individuals. Conversely, the probability of the word composition *(feel, xcomp, hopeless)* is lower, deviating from the expression pattern of normal individuals. Between normal individuals and those with psychological disorders. Another example in Fig. 5b shows that the words *hit* and *run* signify a term in baseball, while the combination of *hit* and *head* is an abnormal signal, which may show the patient having an abuse experience at childhood.


feel
xcomp
obj
I
optimistic
future
about
the
$-\log p(\text{feel}; xcomp; \text{optimistic}) = 1.47$
I feel optimistic about the future
(a) Non-Toxic Text
feel
xcomp
obj
I
hopeless
future
about
the
$-\log p(\text{feel}; xcomp; \text{optimistic}) = 0.04$
I feel hopeless about the future
(b) Patient Description
hit
obl
obj
I
run
inner
a
home
in
the
last
$-\log p(\text{hit}; obj; \text{run}) = 0.84$
I hit a home run in the last inning
(a) Non-Toxic Text
hit
obl
obj
They
often
head
reason
my
without
apparent
$-\log p(\text{hit}; obj; \text{head}) = 0.27$
They often hit my head without apparent reason
(b) Patient Description

(a) Comparison on the relation *xcomp* (b) Comparison on the relation *obj*

Fig. 5. Identifying the patterns for psychological disorder

5 Conclusion

We investigate the word composition knowledge by integrating the syntax information with pre-trained word semantics. We conducted a series of experiments on multiple datasets to validate the learned word composition knowledge and reuse the knowledge in downstream tasks like data augmentation. The results convince the positive effects.

References

1. Blevins, T., et al.: Deep RNNs encode soft hierarchical syntax. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 14–19 (2018)

2. Bowman, S.R., et al.: A fast unified model for parsing and sentence understanding. In: 54th Annual Meeting of the Association for Computational Linguistics, pp. 1466–1477 (2016)
3. Chomsky, N.: Cartesian Linguistics: A Chapter in the History of Rationalist Thought. Cambridge University Press, Cambridge (2009)
4. Devlin, J., et al.: BERT: pre-training of deep bidirectional transformers for language understanding. arXiv:1810.04805 (2018)
5. Hromei, C.D., et al.: End-to-end dependency parsing via auto-regressive large language models (2023)
6. Imai, M., et al.: A cross-linguistic study of early word meaning: Universal ontology and linguistic influence. Cognition **62**(2), 169–200 (1997)
7. Joulin, A., et al.: Bag of tricks for efficient text classification. arXiv preprint arXiv:1607.01759 (2016)
8. Levy, O., et al.: Dependency-based word embeddings. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 302–308 (2014)
9. Li, C., et al.: Training and evaluating improved dependency-based word embeddings. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)
10. Liu, T., Sun, Y.: End-to-end adversarial sample generation for data augmentation. In: Findings of the Association for Computational Linguistics: EMNLP 2023, pp. 11359–11368 (2023)
11. Liu, T., et al.: Unsupervised paraphrasing under syntax knowledge. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, pp. 13273–13281 (2023)
12. Maas, A.L., et al.: Learning word vectors for sentiment analysis. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pp. 142–150 (June 2011)
13. Mamdani, E.H.: Application of fuzzy logic to approximate reasoning using linguistic synthesis. IEEE Trans. Comput. **26**(12), 1182–1191 (1977)
14. Marneffe, D., et al.: Universal dependencies. Comput. Linguist. **47**(2), 255–308 (2021)
15. Mikolov, T., et al.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
16. Pennington, J., et al.: Glove: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543 (2014)
17. Roth, T., et al.: Token-modification adversarial attacks for natural language processing: a survey (2021)
18. Socher, R., et al.: Recursive deep models for semantic compositionality over a sentiment treebank. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 1631–1642 (2013)
19. Zang, Y., et al.: Word-level textual adversarial attacking as combinatorial optimization. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 6066–6080 (2020)
20. Zhang, M., et al.: Dependency-based syntax-aware word representations. Artif. Intell. **292**, 103427 (2021)
21. Zhang, T., et al.: Bertscore: evaluating text generation with BERT. In: 8th International Conference on Learning Representations, ICLR 2020 (2020)
22. Zhu, Y., et al.: Texygen: a benchmarking platform for text generation models. In: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, pp. 1097–1100. ACM (2018)