Natural Language Processing (2026), **32**, pp. 430–450
doi:10.1017/nlp.2026.10025

CAMBRIDGE UNIVERSITY PRESS

ARTICLE

Skeleton-based approach for semantic textual similarity

Hao Wu[1], Degen Huang[1,2] and Xiaohui Lin[1]

[1]School of Computer Science and Technology, Dalian University of Technology, Dalian, China and [2]School of Computing and Artificial Intelligence, Fuyao University of Science and Technology, Fuzhou, China
Corresponding author: Degen Huang; Email: huangdg@dlut.edu.cn

(Received 7 October 2023 UTC; revised 18 March 2024 UTC; accepted 16 March 2026 UTC;
first published online 9 June 2026 UTC)

Abstract

Semantic textual similarity (STS) is to measure semantic equivalence between sentences; it plays an important role in natural language processing (NLP) tasks. The major core of STS is text representation. This paper studies how to obtain a sentence skeleton for text representation in STS. Unlike most existing syntax models, we propose a skeleton-based reinforcement learning method to identify the skeleton and construct sentence representations. Parallel networks are adopted to extract features of different dimensions in the sentence. In the framework of parallel networks, two sentence representation models are designed: context constrained LSTM (CC-LSTM) and Adorned skeleton LSTM (AS-LSTM). CC-LSTM builds the sentence representation by constraining the word context. AS-LSTM constructs the sentence representation through using the identified skeleton and its qualifiers. Our approach achieves good results without using external resources. Especially AS-LSTM, which outperforms the state-of-the-art without using external resources in the SICK dataset.

Keywords: Reinforcement learning; semantic textual similarity; sentence skeleton

1. Introduction

Semantic textual similarity (STS) measures the semantic equivalence between two sentences. This makes STS a crucial preliminary step in many NLP tasks, such as intelligent question answering, machine translation (MT), and automatic summarisation. For intelligent question-answering tasks, all candidate answers are ranked according to sentence similarity to a given question (Wang *et al.* (2007); Yang *et al.* (2015)). In MT, sentence similarity is used to judge whether two sentences have the same meaning (Yin and Schütze (2015); He *et al.* (2016); Santos *et al.* (2020)).

Existing work on STS can be broadly divided into two categories: lexicon-based and semantic-based. Lexicon-based approaches (Richardson and Smeaton (1995); Niwattanakul *et al.* (2013); Opitz *et al.* (2021); Wang and Yu (2023)) calculate the correlation between the character streams of two sentences being compared, which can be applied at the level of characters or words. The major core of semantic-based approaches is text representation. Current text representation approaches are mostly based on two models: sequence models and structured models. Regarding the studies on sequence models, Pagliardini *et al.* (2018) and Le and Mikolov (2014) proposed Sent2vec and Doc2Vec to calculate sentence similarity, which use pre-trained word/sentence embeddings directly for the similarity task without training a neural network model on them. He *et al.* (2015) proposed an elaborate convolutional network (ConvNet) variant which infers sentence similarity by extracting features at multiple levels of granularity and using multiple types of pooling. Mueller and Thyagarajan (2016) proposed a siamese recurrent neural networks, which use two identical long short-term memory (LSTM) networks to project zero-padded word

© The Author(s), 2026. Published by Cambridge University Press. This is an Open Access article, distributed under the terms of the Creative Commons Attribution-NonCommercial licence (http://creativecommons.org/licenses/by-nc/4.0/), which permits non-commercial re-use, distribution, and reproduction in any medium, provided the original article is properly cited. The written permission of Cambridge University Press or the rights holder(s) must be obtained prior to any commercial use.

https://doi.org/10.1017/nlp.2026.10025 Published online by Cambridge University Press

embeddings of a sentence to fixed-sized 50-dimensional vectors using Manhattan distance as the similarity function between 2 sub-networks. Tharindu *et al.* (2019) extended the siamese recurrent neural networks and used thesaurus-based augmentation (Miller, 1992) to add 10,022 additional training examples. The siamese recurrent neural networks acquire excellent results. Bidirectional Encoder Representations from Transformers (BERT) (Devlin *et al.* 2019) is a pre-trained transformer network (Vaswani *et al.* 2017), which set new state-of-the-art results for various NLP tasks, including question answering, sentence classification, and sentence-pair regression. BERT and RoBERTa (Liu *et al.* 2019) have set a new state-of-the-art performance on sentence-pair regression tasks like STS. Reimers and Gurevych (2019) proposed Sentence-BERT (SBERT), a modification of the pre-trained BERT network, and it uses siamese and triplet network structures to derive semantically meaningful sentence representations. Based on the BERT model, Gao *et al.* (2021) proposed SimCSE, a simple contrastive sentence embedding framework, which can produce superior sentence embeddings. Various contrastive learning-based approaches (Chuang *et al.* 2022) rely on two main components, data augmentation and an instance-level contrastive loss. However, the discrete nature of the text makes it challenging to establish universal rules for effective text augmentation generation. Zhang *et al.* 2022 tackle the challenge by presenting a neighbourhood-guided virtual augmentation strategy to support contrastive learning. As it is widely acknowledged that structural information is vital for NLP (Nguyen *et al.* (2019); Zhang *et al.* (2020)), there is an increasing interest in improving text representation by using syntactic information.

In order to use sentence structure information, Richard *et al.* (2013) proposed to use a recurrent neural network to construct a text representation. This method applied a pre-specified parse tree to the recurrent neural network (RNN) to construct a structured representation. Tai *et al.* (2015) proposed a generalisation of the standard LSTM architecture to tree-structured and showed its superiority for representing sentence meaning over a sequential. Most of these works enhanced pre-trained language model (LM) by adding syntax-driven attention components to the transformer (Li *et al.* (2021); Xu *et al.* (2021); Bai *et al.* (2021)). They used the added components to produce a syntax-aware representation and inject this additional representation into the origin alone from the vanilla transformer so as to get a final syntax-enhanced representation. There are also some efforts on incorporating syntax-related objectives into the pre-training stage, such as syntax head prediction (Wang *et al.* 2021) and dependency distance prediction (Xu *et al.* 2021). Zhang *et al.* (2022) proposed the syntax-guided contrastive language model (SynCLM). Based on contrastive learning, SynCLM uses syntactic information to create contrastive positive and negative examples and uses them to help the pre-trained LM to learn rich syntactic knowledge. Yet, there are few available resources for treebank annotations, and the results of existing grammatical analysis models are not very satisfactory.

In the absence of corpus resources, reinforcement learning is introduced to solve problems to a certain degree (Yogatama *et al.* (2017); Feng *et al.* (2017)). Zhang *et al.* (2018) proposed a reinforcement learning (RL) method to build structured sentence representations by identifying task-relevant structures without explicit structure annotations. The frame structure of the sentence can promote the understanding of the overall structure of the sentence, but the sentence components can better reflect the semantic information of the sentence. However, the above models use reinforcement learning to learn sentence structures and build sentence representations with structures. They only refer to the sentence structure at the phrase level, without indicating the role of words in the sentence.

In this paper, we propose a skeleton-based reinforcement learning approach to increase similarity calculation accuracy. Skeleton grammar is essentially a selected subset of dependency grammar, which not only constructs the sentence structure but also indicates the function of each word in the sentence. The approach consists of three components: policy network (PNet), sentence representation model (SRM), and similarity calculation model (SCM). First, the skeleton words are identified through the policy network. Then the skeleton words are delivered to the SRM. According to the skeleton words, the sentence representation is constructed in the framework

https://doi.org/10.1017/nlp.2026.10025 Published online by Cambridge University Press

of parallel networks by designing two sentence representation models: CC-LSTM and AS-LSTM. CC-LSTM builds the sentence representation by constrained word context, and it is thought the relationship between the front word and the back word of the skeleton word is not close, so the skeleton word will be used as the boundary of the context. AS-LSTM first identifies the skeleton word of the sentence and then constructs the skeleton representation of the current skeleton word through the qualifier of the skeleton word. AS-LSTM not only highlights the primary information of the sentence but also retains the secondary information.

2. Related work

2.1 Application of reinforcement learning in NLP tasks

RL (Gu *et al.* 2022) is a promising approach that automatically and actively explores the environment and achieves the optimised strategy for final tasks (Karthik *et al.* (2015); Wang *et al.* (2018); Wang *et al.* (2020); Wang *et al.* (2019)). RL has been explored in NLP-related interactive tasks such as text-based games (Karthik *et al.* 2015), text summarisation (Tang *et al*. 2023), dialogue systems (Wang *et al.* 2018), machine translation (Yehudai *et al.* 2022), information extraction (Huang *et al.* 2023), and text classification (Lichen *et al.* 2021).

Compressive text summarisation offers a balance between the conciseness issue of extractive summarisation and the factual hallucination issue of abstractive summarisation. Dual-agent reinforcement learning (Tang *et al.* 2023) utilises unsupervised dual-agent reinforcement learning to optimise a summary's semantic coverage and fluency by simulating human judgement on text summarisation quality. For sentence summarisation, Hyun *et al.* (2022) devise an abstractive model based on RL without ground-truth summaries. To further enhance the summary quality, the research developed a multi-summary learning mechanism that generates multiple summaries with varying lengths for a given text while making the summaries mutually enhance each other.

Applying RL following maximum likelihood estimation (MLE) pre-training is a versatile method for enhancing neural machine translation (NMT) performance. Yehudai *et al.* (2022) explore the impact of the size of the vocabulary and the dimension of the action space on MT performance. The research finds that effectively reducing the dimension of the action space without changing the vocabulary also yields notable improvement as evaluated by BLEU, semantic similarity, and human evaluation.

Information extraction (IE) has been studied extensively. The experiments observe that different extraction orders can significantly affect the extraction results for a great portion of instances, and the ratio of sentences that are sensitive to extraction orders increases dramatically with the complexity of the IE task. Huang *et al.* (2023) propose an RL-based framework to generate an optimal extraction order for each instance dynamically.

Unlike all the above studies, text classification is to assign one or multiple category labels to a sequence of text tokens. The main task of RL in text classification tasks is to assist text feature extraction. Lichen *et al.* (2021) propose SentRL, a reinforcement learning-based framework for aspect-based sentiment classification. In this framework, an agent is deployed to walk on the dependency graphs of sentiment text, explore paths from target aspect nodes to their potential sentimental regions, and differentiate the effectiveness of different paths. In this paper, we propose a RL method to learn sentence representation by discovering the skeleton of the sentence.

2.2 Fusion syntax model

Syntax is a crucial prior for NLP-oriented neural network models. Along this direction, a range of interesting approaches have been proposed, like Tree-LSTM (Tai *et al.* 2015), PECNN (Yang *et al.* 2016), SDP-LSTM (Xu *et al.* 2015), Supervised Treebank Conversion (Jiang *et al.* 2018), PRPN (Shen *et al.* 2018), and ON-LSTM (Shen, 2019).

Recent works also integrate syntactic knowledge into the transformer and BERT. Sachan *et al.* (2021) investigate popular strategies for incorporating dependency structures into pre-trained

https://doi.org/10.1017/nlp.2026.10025 Published online by Cambridge University Press

language models, revealing essential design decisions are necessary for strong performances. Also using dependency parsing, Li *et al.* (2021) propose a syntax-aware local attention (SLA) which applies dependency parsing to the input text and integrates it with BERT. In addition, for constituency parsing, Syntax-BERT (Bai *et al.* 2021) has a complete self-attention typology and decomposes the self-attention network into multiple sub-networks according to the tree structure.

The above research indicates that using syntactic structures in general fields can improve the performance of BERT and Transformer. Some researchers combine domain-specific information to construct sentence frameworks and add them to the corresponding models.

Bao *et al.* (2023) propose a novel opinion tree parser, which aims to model and parse the sentiment elements from the opinion tree structure. The tree structure is a novel context-free opinion grammar, which is generalised and well-designed; it is used to normalise the sentiment elements into a comprehensive and complete opinion tree. The detailed evaluation shows that the model significantly advances the state-of-the-art performance on several benchmark datasets. Tang *et al.* (2022) propose three syntax-enhanced models based on the domain-specific BioBERT model: Chunking-Enhanced-BioBERT and Constituency-Tree-BioBERT, in which constituency information is integrated; and a Multi-Task-Learning framework, Multi-Task-Syntactic (MTS)-BioBERT, in which syntactic information is injected implicitly by adding syntax-related tasks as training objectives.

Multiple research studies show that golden syntax trees can dramatically improve the performance of semantic representation, but there are few available resources for treebank annotations, and the results of existing grammatical analysis models are not very satisfactory. In order to overcome the limitations of a lack of manually annotated syntax trees, Gu *et al.* (2022) revisit the LM-based unsupervised parsing models by providing a phrase-centred perspective. The research proposes to regularise the parser with phrases extracted by an unsupervised phrase tagger to help the LM model quickly manage low-level structures. Similarly, our approach effectively overcomes this limitation. We utilise reinforcement learning to generate sentence skeletons and construct sentence representations based on the skeleton structure. In the experiment, no additional syntax frameworks were used.

3. Methodology

The paper proposes a skeleton-based reinforcement learning method to increase similarity accuracy. The overall process is shown in Figure 1. This approach consists of three components: PNet, SRM, and SCM. PNet adopts a stochastic policy and samples an action at each state. It keeps sampling until the end of a sentence and produces an action sequence for the sentence. Then, PNet transfers the action sequence into a SRM based on the framework of parallel networks. In the parallel network, two sentence representation models are designed: CC-LSTM and AS-LSTM. SCM calculates similarity based on the sentence representation and offers reward computation to PNet.

3.1 Policy network (PNet)

The policy network (Sutton *et al.* 2000) adopts a stochastic policy $\pi(a_t|s_t;\theta)$ and uses a delayed reward to guide the policy learning. It samples an action with the probability at each state. The delayed reward comes from the SCM and the rule of manual summary, so it is necessary to complete the action sampling of the entire sentence to get the corresponding reward. Following, we specifically introduce the state, action, reward function, and objective function.

State. The state contains the context information of the current word. In the model, LSTM is adopted to encode the context of the current word to generate the current state.

https://doi.org/10.1017/nlp.2026.10025 Published online by Cambridge University Press

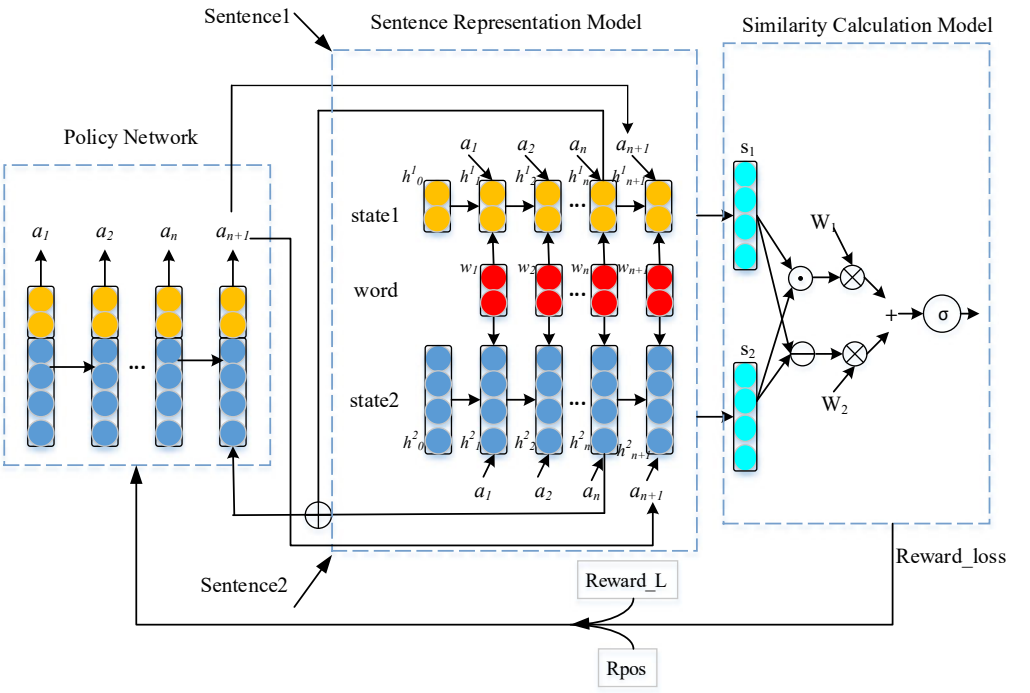


Figure 1. Illustration of the overall process. The policy network (PNet) samples an action at each state. The sentence representation model (SRM) offers state representation to PNet and outputs the final sentence representation to the SCM when all actions are sampled. SCM performs similarity calculation and provides reward_loss to PNet. Reward by Reward_L, Rpos and Reward_Loss consists of three parts. The SRM includes parallel networks. Among them, $h^1{}_n$ represents the low-dimensional hidden layer of LSTM step n-th, and $h^2{}_n$ represents the high-dimensional hidden layer of LSTM step n-th. Taking the n-th word in a sentence as an example to illustrate the interaction between PNet and SRM. $\oplus$ represents the connection of two vectors. $\odot$ represents a dot product. $\otimes$ represents matrix multiplication. $\ominus$ represents vector subtraction. $+$ indicates vector addition.

Action. Actions are generated based on the generated state. In the model, there are two actions to form a binary action space {*qualifier*, *skeleton*}. The action spaces indicating that the word is a qualifier word or a skeleton word of a sentence. *Skeleton* refers to the remaining parts after the layers of attributive, adverbial, and complement are removed one after another and mainly includes the subject, the predicate, and the object. The *qualifier* words refer to the attributive, adverbial, complement, and so on, which modifies the skeleton words.

We adopt a stochastic policy. Let the action at state t be denoted; the policy is defined as follows:

$$\pi(a_t|s_t;\theta) = \sigma(W \times s_t + b) \tag{1}$$

where $\pi(a_t|s_t;\theta)$ denotes the probability of choosing, σ denotes the sigmoid function, and $\theta = \{W, b\}$ denotes the parameters set of PNet.

Reward. A reward is a typical delayed reward because no reward can be obtained until the action sequence is fully generated. In this model, the reward is divided into three parts:

- The part-of-speech tagging weight rewards (*Rpos*): The part-of-speech (POS) tagging is very helpful in the identification of the skeleton of the sentence. Words with some parts of speech are probably the skeleton words of the sentence, such as verbs, nouns, pronouns, etc., but words of some parts of speech are of very low probability, such as adjectives,

https://doi.org/10.1017/nlp.2026.10025 Published online by Cambridge University Press

adverbs, etc. Therefore, according to the characteristics of each POS, the corresponding weight can be assigned, which can greatly improve the efficiency of the experiment and avoid many detours. The weight of each part of speech is set according to the ratio of the POS to the skeleton word in the corpus.

$$Rpos = \frac{\delta \sum_{i=1}^{lenth} |a_i{}^{\nu} - \lambda|}{lenth} \tag{2}$$

where *lenth* denotes the sentence length, and λ is the weight of POS. a_i represents an action. If a_i is the *skeleton*, then the value of $a_i{}^{\nu}$ is 1; otherwise, it is 0. δ is the average of loss during pre-training.

- The relative length of the skeleton words rewards ($Reward_L$): In sentences of similar length, the number of skeleton words is similar, and the number of skeleton words also increases as the length of the sentence increases. In the experiment, the $Reward_L$ function is summarised based on the corpus annotated in the validation set, and we get the following reward function.

$$Reward_L = \begin{cases} 0.3^*(len_{skeleton} - lg(\alpha^* lenth^2) - \beta^* lenth) & 0.06^* lenth + 1.8 \leq len_{skeleton} \leq 0.6^* lenth + 1.6 \\ len_{skeleton} - lg(\alpha^* lenth^2) - \beta^* lenth & \text{otherwise.} \end{cases} \tag{3}$$

where $len_{skeleton}$ denotes the number of skeleton words in a sentence, *lenth* is sentence length, and α and β are parameters and set manually. According to the experiment, we set the value of parameter α as 0.4 and the value of parameter β as 0.08.

- After the action sequence is generated, the instructive SRM generates a sentence representation, and the final SCM gets corresponding rewards based on the sentence representation. The reward function includes the loss function of the SCM.

Therefore, the reward function of this model follows Eq. 4:

$$Reward = Rpos + Reward_L + J(\theta) \tag{4}$$

3.2 Sentence representation model

In the SRM, our proposed parallel networks model is outlined in Figure 3. There are two networks, $PN^{(a)}$ and $PN^{(b)}$; the hidden layer dimensions of the two networks are different. Based on the parallel networks framework, we design two sentence representation models: context constrained LSTM (CC-LSTM) and Adorned skeleton LSTM (AS-LSTM). Both of these models are based on the BiLSTM and attention mechanisms.

3.2.1 BiLSTM

LSTM (Hochreiter and Schmidhuber 1997) is a variant of a RNN. It is mainly used to solve the problem of vanishing gradient. The LSTM model can selectively retain context information through control mechanisms such as input gates, forget gates, and output gates, and it can capture long-distance dependencies in the text. Many variations of LSTM have been applied to natural language processing tasks, such as MT, question-answering systems, and sentence classification. Bidirectional LSTM is one variant, which can effectively obtain the features before and after the current word. BiLSTM mainly consists of three representation layers: an embedding layer, a BiLSTM layer, and an output layer. Figure 2 shows the basic structure of BiLSTM.

https://doi.org/10.1017/nlp.2026.10025 Published online by Cambridge University Press

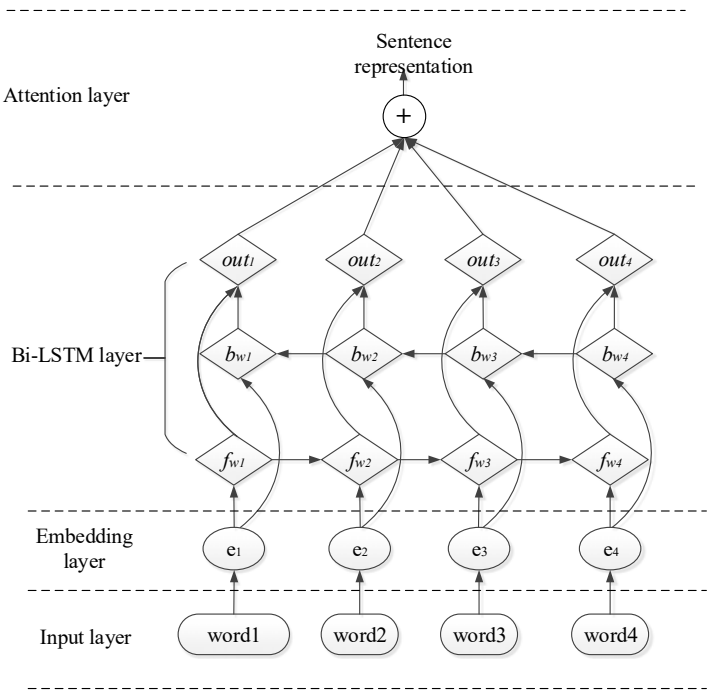


Figure 2. BiLSTM model and Attention mechanism.

3.2.2 Attention mechanism

Transformers and BERT have contributed new state-of-the-art results to various NLP tasks. The attention layer is depicted in Figure 2. Let O be a matrix consisting of the hidden layer vectors $[O_1 O_2 \ldots O_n]$ in the hidden layer, and O is the input of the attention layer. The attention-based method represents the sentence, including sentence skeleton information, as a weighted sum of these hidden layer vectors.

$$A = \tan h(O) \tag{5}$$

$$\alpha = softmax(\omega_T * A) \tag{6}$$

$$S_1 = \tan h(O * \alpha^T) \tag{7}$$

Here, α is the normalised weight vector, and ω is a parameter vector. S_1 is the initial state, $S_1 = Att(X; \theta_1)$, and θ_1 represents all the parameters in this method.

3.2.3 Parallel networks

The proposed parallel networks model is outlined in Figure 3. There are two networks, $PN^{(a)}$ and $PN^{(b)}$, which process the same sentence but have different hidden layer dimensions for obtaining features from multiple levels. Comparing the two networks, the input word embedding for $PN^{(a)}$ and $PN^{(b)}$ are the same at each step. When generating hidden layers, $PN^{(a)}$ uses high-dimensional weight vectors, with hidden layers having higher dimensions. On the contrary, $PN^{(b)}$ uses low latitude weight vectors, and the hidden layer dimensions are relatively low.

The final sentence representation is obtained by concatenating the output results of $PN^{(a)}$ and $PN^{(b)}$. In the experiment, the parallel networks model used two dimensions of hidden layers,

https://doi.org/10.1017/nlp.2026.10025 Published online by Cambridge University Press

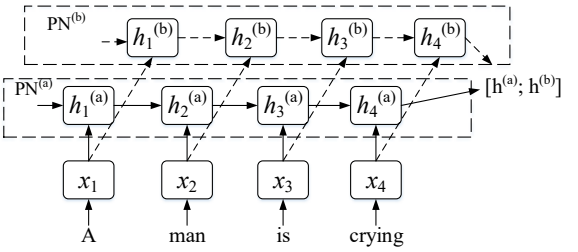


Figure 3. The overall process of parallel networks, $PN^{(a)}$ and $PN^{(b)}$, have the same input, but the hidden layer dimensions are different: sentence representation from an output concatenation of $PN^{(a)}$ and $PN^{(b)}$.

900 and 300, respectively. The experiment proved that the 300-dimension hidden layer performed better on short sentences, while the 900-dimension hidden layer had better results on long sentences.

3.2.4 Context constrained LSTM (CC-LSTM)

The main idea of context constrained LSTM (CC-LSTM) is to build a sentence representation by constrained word context. In this way, it is expected to get more relevant context for the current word. In CC-LSTM, there is a two-level structure:

1. Contextual representation layer which builds contextual representation using skeleton words as the boundary.
2. Sentence representation layer which gets sentence representation with contextual representation as input.

- Contextual representation layer: The skeleton word has a closer relationship with the modifier phrase in the context and the nearby skeleton word. As a qualifier word, its main function is to modify the skeleton word of the context, so its context mainly includes the words between the two skeleton words before and after. Take the sentence in Figure 4 as an example: "*dog*" is the skeleton word, which is more closely related to "*a*," "*brown*," and "*is*." "*In*" is a qualifier word; "*is*" and "*grass*" are the same distance from it, but "*in*" and "*grass*" are more closely related because they are in the same constrained space.

CC-LSTM translates the actions obtained from PNet to a SRM. Formally, given a sentence $W = w_1 w_2 \cdots w_n$, there is a corresponding action sequence $A = a_1 a_2 \cdots a_n$ obtained from PNet. In this setting, each action a_i at word position x_i is chosen from {*qualifier*, *skeleton*}, where *qualifier* indicates that the word has a modification function in a sentence, and a *skeleton* means that the word is skeleton word and acts as a boundary of context.

To limit the word context through the skeleton words, each word will get a variable-length context window. In order for context representation to work in the BiLSTM model, we need to map variable-length contexts to fixed-length spaces. We use the LSTM to map the variable-length context. Taking w_6 in Figure 4 as an example, the upper boundary of w_6 is w_5, and the lower boundary is w_9. The context representation is as follows Eq. 8–10:

$$c_6^1 = LSTM(w_5, w_6) \tag{8}$$

$$c_6^2 = LSTM(w_6, w_7, w_8, w_9) \tag{9}$$

$$c_6 = c_6^1 \oplus c_6^2 \tag{10}$$

https://doi.org/10.1017/nlp.2026.10025 Published online by Cambridge University Press

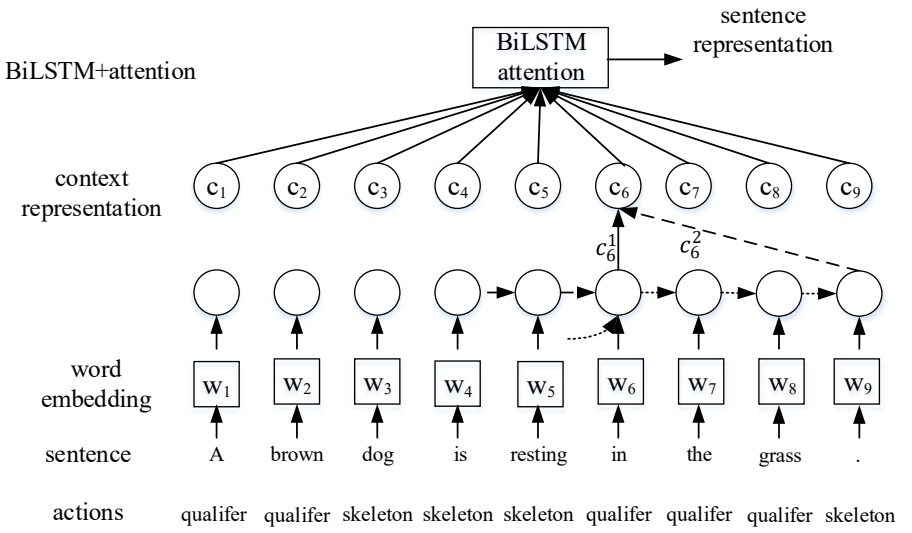


Figure 4. Examples for context constrained LSTM (CC-LSTM).

where LSTM denotes the LSTM model, w_6 is the embedding of "*in*," c_6 is the context representation of "*in*," and $\oplus$ indicates vector concatenation. The contextual representation of other words is similar to w_6. We need to generate a contextual representation of each word one by one.

- The sentence representation layer. The sentence representation layer contains two parts: BiLSTM and an attention mechanism. The input of BiLSTM is the context representation, and its hidden layers are used as the input of the attention mechanism.

$$o = BiLSTM(c_1, c_2, \ldots, c_n) \tag{11}$$

$$S = att(o) \tag{12}$$

where o denotes hidden layers of BiLSTM, it contains multiple outputs; S is sentence representation.

3.2.5 Adorned skeleton LSTM (AS-LSTM)

Adorned skeleton LSTM (AS-LSTM) first identifies the skeleton word of the sentence and then constructs the skeleton representation of the current skeleton word through the qualifiers of the skeleton word. We define the qualifiers for the current skeleton word to be the word between the two skeletons before and after it. This process is implemented by sampling an action in {*qualifier*, *skeleton*} at each word position, where *skeleton* indicates that a word is the skeleton of a sentence and *qualifier* means the word has a modification function.

As shown in Figure 5, the AS-LSTM adopts a two-level structure: a skeleton representation layer and a sentence representation layer.

- Skeleton representation layer: In the skeleton representation layer, the skeleton words divide the sentence into several independent fragments; each fragment includes a skeleton word and several qualifier words of the skeleton word. The skeleton word needs to be combined with qualifier words to reflect its true meaning. Therefore, we need to construct the skeleton representation based on the qualifier. The qualifiers of a skeleton are divided

https://doi.org/10.1017/nlp.2026.10025 Published online by Cambridge University Press

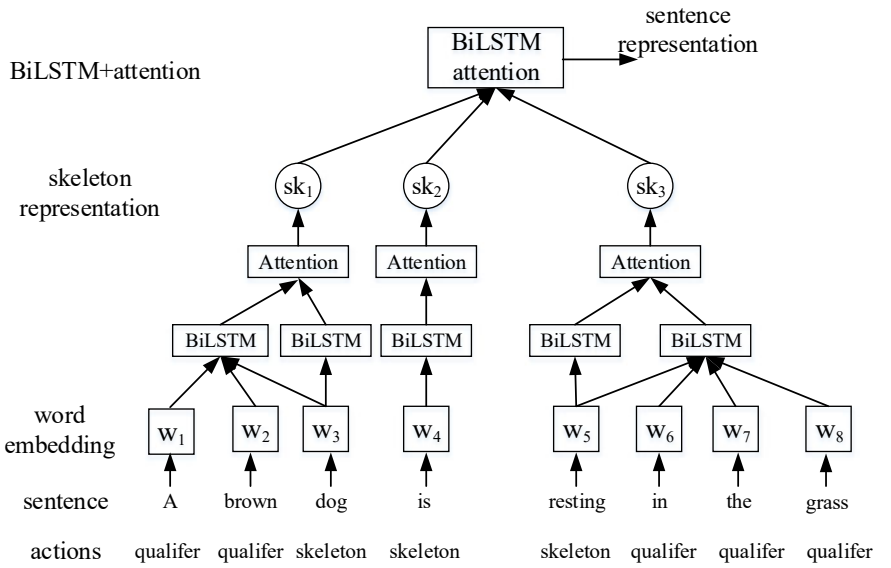


Figure 5. Examples for Adorned skeleton LSTM (AS-LSTM).

into pre-modifiers and post-modifiers according to their positions. A pre-modifier and a post-modifier are allowed to be null. The pre-modifier and skeleton word generate the top skeleton representation. The post-modifier and skeleton word generate the tail skeleton representation. Take the skeleton word "*dog*" as an example. "*dog*" is a skeleton word; its pre-modifier is "*A*" and "*brown*," and its post-modifier is none.

$$f_1 = BiLSTM(w_1, w_2, w_3) \tag{13}$$

$$b_1 = BiLSTM(w_3) \tag{14}$$

$$sk_1 = att(f_1, b_1) \tag{15}$$

where *BiLSTM* denotes the BiLSTM model, f_1 is the skeleton representation of the pre-modifier, b_1 is the skeleton representation of the post-modifier, and sk_1 is the skeleton representation of "*dog*." The attention mechanism is applied to give different weights to f_1 and b_1 that have different effects on the skeleton words.

- Sentence Representation Layer: The sentence representation layer contains two parts: a BiLSTM and an attention mechanism. The input of BiLSTM is the skeleton representation, and its hidden layers are used as the input of the attention mechanism.

$$o = BiLSTM(sk_1, sk_2, sk_3) \tag{16}$$

$$S = att(o) \tag{17}$$

where o denotes hidden layers of BiLSTM, it contains multiple outputs; S is sentence representation.

3.3 Similarity calculation model

Given a sentence pair, we predict a real-valued similarity score in some range $[1, K]$, where $K > 1$ and is an integer. The sequence $\{1, 2, \cdots, K\}$ is some ordinal scale of similarity, where

https://doi.org/10.1017/nlp.2026.10025 Published online by Cambridge University Press

higher scores indicate greater degrees of similarity, and we allow real-valued scores to account for ground-truth ratings that are the average over the evaluations of several human annotators.

Firstly, SRM generates a representation of sentence pairs. Given these sentence representations, we predict the similarity score $\hat{y}$ using a neural network that considers both the distance and angle between the pair (h_L, h_R):

$$h_\times = h_L * h_R \tag{18}$$

$$h_ = |h_L - h_R| \tag{19}$$

$$h_s = \sigma\left(W^{(\times)}h_\times + W^{(+)h_ + b^{(h)}}\right) \tag{20}$$

$$\hat{p}\theta = softmax\left(W^{(p)}h_s + b^{(p)}\right) \tag{21}$$

$$\hat{y} = r^T\hat{p}\theta \tag{22}$$

where $r^T = [12...K]$ and the absolute value function is applied elementwise. The use of both distance measures $h_\times$ and $h_$ is empirically motivated: it is found that the combination outperforms the use of either measure alone. The multiplicative measure $h_\times$ can be interpreted as an element-wise comparison of the signs of the input representation. The subtractive measure $h_$ can measure the difference between the input representation $\hat{p}\theta$ and the vector representation of our predicted outcome. $\hat{y}$ is predicted outcome.

The cost function is the regularised KL-divergence (Solomon and Leibler 1951) between p and $\hat{p}\theta$:

$$J(\theta) = \frac{1}{m}\sum_{k=1}^{m} KL(p^k||\hat{p}\theta^k) + \frac{\lambda}{2}||\theta||_2^2 \tag{23}$$

where m is the number of training pairs and the superscript k indicates the kth sentence pair. Note that we use the same KL-loss function and sparse target distribution technique as (Tai *et al.* 2015).

4. Experiments

4.1 Tasks and datasets

The datasets SICK and SemEval are used in the experiment.

4.1.1 Sentences involving compositional knowledge (SICK) dataset

The dataset is from the 2014 SemEval competition (Marelli *et al.* 2014) and consists of 9927 sentence pairs, 4500 pairs of training set, 500 pairs of validation set, and 4927 pairs of test set. Each sentence pair is scored manually, and the relatedness score range is [1, 5]. With higher scores indicating the two sentences are more closely related.

4.1.2 STS benchmark dataset

The dataset comprises 8628 sentence pairs. The STS Benchmark dataset was used in the STS tasks and organised in the context of SemEval between 2012 and 2016. The selection of datasets includes text from image captions, news headlines, and user forums. 8,628 different sentence pairs of varying

The Microsoft Video Paraphrase Dataset (MSRVID) (Agirre *et al.* 2012) was collected for the 2012 SemEval competition and consists of 1,500 pairs of short video descriptions which were then annotated. Among them, 1000 sentence pairs are used as a training corpus, 250 sentence pairs as a test corpus, and 250 sentence pairs as a validation corpus.

https://doi.org/10.1017/nlp.2026.10025 Published online by Cambridge University Press

4.1.3 Hyperparameters and training detail

Hyperparameters. Dropout (Srivastava, Salakhutdinov, and Hinton 2013) is applied to embedding hidden states with a rate of 0.5. All models are optimised using the Adam optimiser (Kingma and Jimmy 2015), with an initial learning rate of 0.0014 and a decay rate of 0.95. A batch size of 32 is adopted. Sentences with similar lengths are batched together. The L2 regularisation parameter is set to 0.00001.

Training detail. The initialisation of word vectors consists of three types of vectors. The three types of vectors mainly include GloVe (Pennington, Socher, and Manning 2014) embeddings, trained word vector from the word2vec (Mikolov *et al.* 2013) model, and trained POS vector from the word2vec model. In the experiment, these three vectors are spliced. Among them, the word vector training corpus based on word2vec is the SICK dataset and STS benchmark dataset. Similarly, the POS tagging sequence required by the POS vector also comes from the POS of the SICK dataset and STS benchmark dataset. The POS tagging tool we use is NLTK. The initial hidden layer required for training is randomly generated. In order to ensure the stability of the initial hidden layer during testing, we uniformly use the mean of the hidden layer vectors generated during the training process.

- Rpos setting: The Rpos function is set based on the weight of the corresponding POS for each word in the sentence. In the corpus, the POS of each word corresponds to a weight, which is calculated based on the corpus annotated in the SICK corpus validation set. That is, the proportion of each POS as the backbone in the corpus is the weight of each POS. Due to the Rpos function assisting reinforcement learning to learn the optimal solution faster, the value of Rpos being too high can interfere with the reward of the result and affect the PNet. To prevent the reward values calculated based on POS weights from being too large, multiply them by a balance parameter. The parameter is the average value of loss during pre-training.
- Reward_L: The reward_L function is summarised based on the corpus annotated in the validation set. In statistics, the sentence length in the corpus is used as the horizontal axis, and the number of skeletons in the sentence is used as the vertical axis. The points formed are denoted as (x, y). When labelling points in coordinates, there is a high coincidence rate because the number of skeleton y and sentence length x are both integers. In order to highlight the density of the points, a number between [–0.5, 0.5] is randomly added to the number of skeleton y and sentence length x, which is equivalent to randomly marking points in a circle with (x, y) as the centre and 0.5 as the radius.

The points composed of the length of each sentence and the number of skeletons in the sentence are marked in Figure 6. And three lines were set, y_1, y_2, and y_3. Among them, y_1 and y_2 are boundary functions, and y_3 is the objective function. The points in the figure are almost all concentrated between the lines y_1 and y_2; therefore, we choose a piecewise function for Reward_L. At the coordinate points between lines y_1 and y_2, we believe that the number of skeletons corresponding to the sentence length is relatively reasonable. Therefore, when calculating the error with the objective function in this area, we multiply it by a coefficient; the coefficient was 0.3 in the experiment. At points outside of lines y_1 and y_2, we consider them unreasonable and use the original values when calculating the error with the objective function.

4.2 Results on SICK dataset

The results are summarised in Table 1. The model uses Pearson's, Spearman's, and MSE as evaluation metrics. In the experiment, we did not choose to expand the external training corpus, so we chose the experimental results under the same training corpus for comparison. In addition, due to some researchers choosing to incorporate validation set data into the training set to train

https://doi.org/10.1017/nlp.2026.10025 Published online by Cambridge University Press

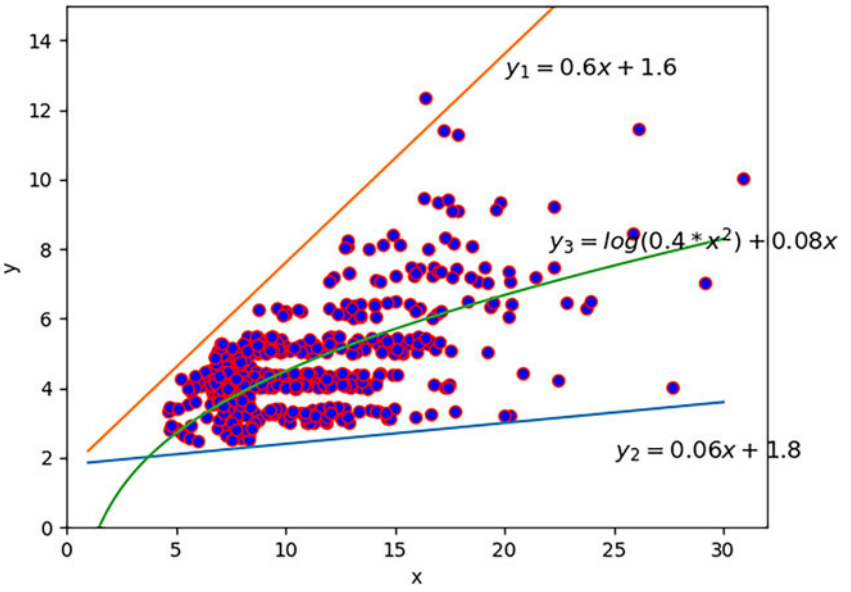


Figure 6. The relationship between the number of skeletons and sentence length in the annotated corpus.

the training model during experiments, we also trained and compared the results under the same training data. We used the suffix 'dev' as a marker for these results. In each column, the best result is shown in bold.

The experiment involves two setups: not using syntactic structure vs using syntactic structure in the process of constructing the sentence representation. In the model of not-using syntactic structure, it is observed that the MALSTM leads to a huge improvement. SimCSE and BERT also used a large amount of corpus for pre-training, but Bert's results on the SICK dataset were not satisfactory. SimCSE added a comparative model on top of Bert's pre-training model, significantly improving the test results. In models that use syntactic information, the best results are not achieved for Tree-LSTM, but the Tree-LSTM improves 1 point compared with the LSTM baseline model.

Skeleton-based approaches do not rely on parse trees but POS to assist skeleton word identification. From the experimental result, AS-LSTM exceeds all the above models under the same training corpus. Even with a single hidden layer, AS-LSTM (900) achieves excellent results. In addition, compared with models using only a single hidden layer, the results are significantly improved in the parallel networks.

In order to find out the reason why the PN can improve the calculation results of sentence similarity, they designed experiments to select a 300-dimensional hidden layer and a 900-dimensional hidden layer for the LSTM model. In the experiment, choose two different sets of data, including length of sentence pairs greater than 13 (393 pairs) and less than 6 (195 pairs). Experimental results in Table 2 show that the 300-dimensional hidden layer has better results in short sentences under the same model. To the contrary, higher dimensions have excellent performance on long sentence pairs.

In the CC-LSTM and AS-LSTM models, the skeleton word decomposes the sentence into multiple parts, reducing the connection of words between different parts. Both of these models perform relatively well in longer sentence pairs.

https://doi.org/10.1017/nlp.2026.10025 Published online by Cambridge University Press

Table 1. Test set results on the SICK, MALSTM-dev model, CC-LSTM-dev model, and AS-LSTM-dev model use 5000 sentence pairs as a training corpus, including the SICK training set and SICK validation set. Others only use the training set as the training corpus. (300) and (900) represent that the models select 300-dimensional and 900-dimensional hidden layers in the process of training the corpus, respectively.

Methods	Pearson	Spearman	MSE
Not-use syntactic structure			
Mean vectors	0.7624	0.7031	0.4321
LSTM	0.8432	0.7827	0.2866
BiLSTM	0.8568	0.7945	0.2756
Bert-Sent2vec	0.8143	0.7735	0.2286
CNN (He *et al.* (2015))	0.8686	0.8047	0.2606
MALSTM-dev (Mueller and Thyagarajan (2016))	0.8822	0.8346	0.2286
BERT (Devlin *et al.* (2019))	\	0.4263	\
SRoBERTa (Reimers and Gurevych (2019))	\	0.7446	\
SBERT (Reimers and Gurevych (2019))	\	0.7291	\
SimCSE-RoBERTa base (Gao *et al.* (2021))	\	0.8050	\
SimCSE-RoBERTa large (Gao *et al.* (2021))	\	0.8195	\
AoE (Xianming and Jing (2024))	\	0.8132	\
PromptEOL + CSEOPT (Ting *et al.* (2024))	\	0.8206	\
Use syntactic structure			
DT-RNN (Socher *et al.* (2014))	0.7923	0.7319	0.3822
SDT-RNN (Socher *et al.* (2014))	0.7900	0.7304	0.3848
Const.LSTM (Tai *et al.* (2015))	0.8582	0.7966	0.2734
Dep. LSTM (Tai *et al.* (2015))	0.8676	0.8083	0.2532
CC-LSTM(300)	0.8663	0.8075	0.2563
AS-LSTM(300)	0.8688	0.8111	0.2502
CC-LSTM(900)	0.8685	0.8094	0.2425
AS-LSTM(900)	0.8714	0.8203	0.2396
CC-LSTM	0.8704	0.8139	0.2425
AS-LSTM	0.8762	0.8273	0.2333
CC-LSTM-dev	0.8793	0.8324	0.2361
AS-LSTM-dev	**0.8835**	**0.8367**	**0.2249**

4.3 Results on STS benchmark dataset

Our results on the STS benchmark dataset are summarised in Table 3. Compared with the results obtained by SimCSE-RoBERTa in large-scale corpus training, the results obtained only in the training set are relatively poor. After analysis, the main reasons are twofold. Firstly, the STS corpus contains too many types of sentences, and there are significant differences between different types of sentences. Secondly, the number of sentences in each type is too small, and SimCSE-RoBERTa is trained on a large-scale corpus to obtain more information. Although our results are worse than SimCSE-RoBERTa, SBERT and SRoBERTa, they are much better than other models trained only in the training corpus. In each column, the best result is shown in bold.

https://doi.org/10.1017/nlp.2026.10025 Published online by Cambridge University Press

Table 2. Compare the results of sentences of different lengths, (300) and (900), representative models. Select 300-dimensional and 900-dimensional hidden layers in the process of training the corpus, respectively

	lenth >13	lenth <6
Method	Pearson	Pearson
Mean vectors	0.6275	0.8237
LSTM	0.7574	0.8782
BiLSTM	0.7738	0.8796
CC-LSTM(300)	0.7825	**0.9053**
AS-LSTM(300)	0.7595	0.9020
CC-LSTM(900)	0.7928	0.8872
AS-LSTM(900)	**0.8019**	0.8863

Table 3. Results on the STS benchmark dataset: "$*$" indicates that the result comes from the internet page

Models	Pearson
Not-use syntactic structure	
Mean vectors	0.6231
LSTM	0.7236
BiLSTM	0.7382
HCTI (Shao 2017))	0.7840
Reddit tuned (Yang *et al.* (2018))	0.7810
BERT (Devlin *et al.* (2019))	0.8430
SRoBERTa (Reimers *et al.* (2019))	0.8492
SBERT (Reimers et al. (2019))	0.8467
SimCSE-RoBERTa base (Gao *et al.* (2021))	0.8583
SimCSE-RoBERTa large (Gao *et al.* (2021))	0.8670
AoE (Xianming and Jing (2024)	**0.8969**
PromptEOL + CSEOPT (Ting *et al.* (2024))	0.8856
Use syntactic structure	
Dependency Tree-LSTM*	0.7120
Constituency Tree-LSTM*	0.7190
CC-LSTM	0.7853
AS-LSTM	0.7946

https://doi.org/10.1017/nlp.2026.10025 Published online by Cambridge University Press

Table 4. Results on MSRVID dataset

Method	Pearson
LSTM	0.8672
BiLSTM	0.8746
PSL (Beltagy *et al.* (2014))	0.8300
CC-LSTM	**0.8933**
AS-LSTM	0.8870

Our results on the MSRVID data are presented in Table 4. The results show the CC-LSTM model has a better performance in short sentence pairs.

4.4 Result analysis

To investigate how the discovered skeleton sentence influences sentence similarity calculation performance, we present result analysis from quantitative perspectives. Statistics show that there is a certain difference between the skeleton information obtained through reinforcement learning and the manually annotated skeleton information. However, from the experimental results, skeleton-based approaches are higher than other models that use syntactic information, so we believe that the skeleton information obtained through reinforcement learning is more suitable for the STS task.

We present further analysis on what POS is identified as the skeleton word. We investigate the impact of part-of-speech tagging on skeleton recognition.

Next, further analyse which POS is easily identified as a skeleton word. And the impact of part of speech tagging on skeleton recognition was studied.

For words of different POS, the probability of being identified as the skeleton of the sentence is also different. We separately count the SICK dataset test results and the sentences marked with the skeleton information in the SICK dataset. Among them, the sentences marked with the skeleton information in the SICK dataset are 495 sentences, and skeleton information is manually labelled. Table 5 shows the total number of important POS in the dataset and the proportion of skeletons. We find the following POS are more likely to be identified as the skeleton word: NN (Noun, singular), NNS (Noun, plural), VBG (Verb, gerund or present participle), VBZ (Verb, 3rd person singular present), VBD (Verb, past tense), and VBN (Verb, participle). In addition, some POS with a low probability of being identified as the skeleton include CD (Cardinal number), DT (Determiner), IN (Preposition or subordinating conjunction), JJ (Adjective), and RB (Adverb).

In this skeleton-based approach, the reinforcement learning can obtain the skeleton of the sentence in the model training and then use the skeleton information to construct the sentence representation. In order to verify the quality of the sentence skeleton recognition results, we use sentences annotated with skeleton information as the standard skeleton tagging dataset for experiments. Table 6 shows the accuracy of the skeleton annotation.

According to the specific analysis of the results based on the annotated corpus, the accuracy of JJ, VBG, VBD, CD, IN, and other parts of speech recognition achieves a satisfactory high level, but the accuracy of noun and NNS recognition is comparatively low.

4.5 Ablation studies

Figure 7 presents ablation studies for the reward function. The reward function is divided into three parts: the loss function, Reward_L, and Rpos. The loss function is the foundation of the

https://doi.org/10.1017/nlp.2026.10025 Published online by Cambridge University Press

Table 5. The frequency of various POS in the dataset and the ratio of the skeleton

	SICK-test		Manually annotated	
	Total	Skeleton ratio	Total	Skeleton ratio
NN	23940	0.7662	902	0.6940
NNS	3900	0.7982	156	0.7308
VBZ	11041	0.9891	479	1.0
VBG	11014	0.9774	536	0.9627
JJ	6024	0.1874	275	0.0655
VBN	960	0.9271	68	0.8529
VBD	63	0.7519	4	0.5
DT	21643	0.0465	1077	0
IN	25581	0.0346	527	0

Table 6. The accuracy of skeleton recognition and the accuracy of each POS skeleton recognition

POS	Accuracy ($\times$ 100%)
All	91.60
NN	77.83
NNS	73.08
VBZ	100.00
VBG	98.88
JJ	93.44
VBN	77.94
VBD	50.00
DT	96.20
IN	97.34

reward function, Reward_L, and Rpos plays a certain auxiliary role. We conduct ablation studies on the AS-LSTM to further analyse the effectiveness of each reward. Compared to using only the loss function as a reward, AS-LSTM_L and AS-LSTM_P have improved the results by 0.06 and 0.22, respectively. They not only have some improvement in results but also greatly reduce the epoch when achieving the best results. Compared to AS-LSTM_L, AS-LSTM_P not only achieves higher and more robust results but also has a lower epoch when obtaining the best results. Adding Reward_L to AS-LSTM_P resulted in an increase of 0.11 in AS-LSTM_A results and a decrease of 3 epochs for optimal results. In addition, in Figure 6, we can also observe that although the BiLSTM model achieves relatively high results quickly, it is particularly prone to overfitting. The AS-LSTM model does not exhibit this performance and exhibits more robust results.

https://doi.org/10.1017/nlp.2026.10025 Published online by Cambridge University Press

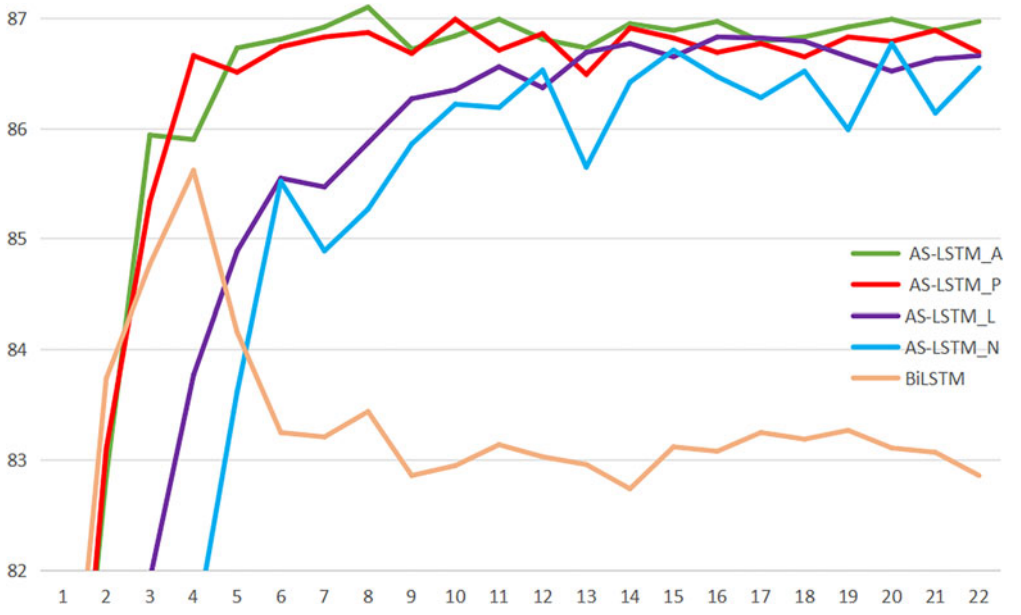


Figure 7. AS-LSTM_N only uses the loss function as the reward function. AS-LSTM_L represents the addition of Reward_L to the reward function in AS-LSTM_N. AS-LSTM_P represents the addition of Rpos to the reward function in AS-LSTM_N. AS-LSTM_A indicates that all rewards have been added. BiLSTM represents using only the BiLSTM model to obtain results. The horizontal axis represents the epoch, and the vertical axis represents the corresponding result of the epoch.

5. Conclusion

This paper presents a skeleton-based reinforcement learning method which learns sentence representation by discovering sentence skeletons. In the framework of the parallel networks, there are two representation models: CC-LSTM and AS-LSTM. CC-LSTM builds the sentence representation by constraining word context and uses the skeleton as the boundary of the word context, restricting the acquisition of the word context. In the AS-LSTM model, words in the sentence are first classified into skeleton words or modifiers, and then the skeleton representation is constructed. AS-LSTM constructs a sentence vector through skeleton representation. Extensive experiments show that our method has outstanding performance, which not only finds the skeleton information of the sentence but also exploits the skeleton information to improve the accuracy of similarity calculation.

6. Limitations

The limitations of our work can be stated from three perspectives. First, the current generated sentence skeleton contains relatively little structural information. It can be future work to explore how to automatically generate the more complex grammar. Secondly, the current model only has two action spaces. In future work, we will try to design multiple action spaces to complete the experiment. Thirdly, the AS-LSTM model is not very friendly to sentences containing negative words. In future work, we will attempt to identify negative words as the skeleton as much as possible for experimentation.

Acknowledgements. This work was supported by the National Natural Science Foundation of China (No.U1936109).

https://doi.org/10.1017/nlp.2026.10025 Published online by Cambridge University Press

References

Agirre E., **Diab M.**, **Cer D.** and **Gonzalez-Agirre A.** (2012). SemEval-2012 task 6: A Pilot on semantic textual similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics*, pp. 385–393.

Bai J., **Wang Y.**, **Chen Y.**, **Yang Y.**, **Bai J.**, **Yu J.** and **Tong Y.** (2021). Syntax-BERT: Improving pre-trained transformers with syntax trees. In *EACL*.

Beltagy I., **Erk K.** and **Mooney R.** (2014). Probabilistic soft logic for semantic textual similarity. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics*, pp. 1210–1219.

Chuang Y.S., **Dangovski R.**, **Luo H.**, **Zhang Y.**, **Chang S.**, **Soljacic M.**, **Li S.W.**, **Yih W.-T.**, **Kim Y.** and **Glass J.** (2022). Diffcse: Difference-based contrastive learning for sentence embeddings. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4207–4218.

Devlin J., **Chang M.-W.**, **Lee K.** and **Toutanova K.** (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT 2019*, pp. 4171–4186.

Feng Y., **Zhang H.**, **Hao W.** and **Chen G.** (2017). Joint Extraction of Entities and Relations Using Reinforcement Learning and Deep Learning. *Computational Intelligence & Neuroscience* **2017**(1), 7643065.

Gao T., **Yao X.** and **Chen D.** (2021). Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Punta Cana, The Dominican Republic, pp. 6894–6910.

Gu X., **Shen Y.**, **Shen J.**, **Shang J.** and **Han J.** (2022). Phrase-aware unsupervised constituency parsing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pp. 6406–6415.

He H., **Gimpel K.** and **Lin J.** (2015). Multi-perspective sentence similarity modeling with convolutional neural network. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1576–1586.

He H., **Wieting J.**, **Gimpel K.**, **Rao J.** and **Lin J.** (2016). UMD-TTIC-UW at SemEval-2016 Task 1: Attention-based multi-perspective convolutional neural networks for textual similarity measurement. In *Proceedings of SemEval-2016*. San Diego, California, pp. 662–667.

Hochreiter S. and **Schmidhuber J.** (1997). Long short-term memory. *Neural Computation* **9**(8), 1735–1780.

Huang W., **Liang J.**, **Zhixu L.**, **Xiao Y.** and **Ji C.** (2023). Adaptive ordered information extraction with deep reinforcement learning. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13664–13678.

Hyun D., **Wang X.**, **Park C.**, **Xie X.** and **Yu H.** (2022). Generating multiple-length summaries via reinforcement learning for unsupervised sentence summarization. In *Findings of the Association for Computational Linguistics: EMNLP, 2022*, pp. 2939–2951.

Jiang X., **Li Z.**, **Zhang B.**, **Zhang M.**, **Li S.** and **Si L.** (2018). Supervised treebank conversion: data and approaches. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long Papers)*, pp. 2706–2716.

Karthik N., **Kulkarni T.** and **Barzilay R.** (2015). Language understanding for text-based games using deep reinforcement learning. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal, pp. 1–11.

Kingma D.P. and **Jimmy B.** (2015). Adam: A method for stochastic optimization. In *Proceedings of International Conference on Learning Representations*.

Le Q. and **Mikolov T.** (2014). Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on International Conference on Machine Learning (ICML'14). JMLR.org*. Vol. 32, pp. 1188–1196.

Li Z., **Zhou Q.**, **Li C.**, **Xu K.** and **Cao Y.** (2021). Improving BERT with syntax-aware local attention. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 645–653.

Lichen W., **Zong B.**, **Liu Y.**, **Qin C.**, **Cheng W.**, **Yu W.**, **Zhang X.**, **Chen H.** and **Fu Y.** (2021). Aspect-based sentiment classification via reinforcement learning. In *2021 IEEE International Conference on Data Mining (ICDM)*, pp. 1391–1396.

Liu Y., **Ott M.**, **Goyal N.**, **Du J.**, **Joshi M.**, **Chen D.**, **Levy O.**, **Lewis M.**, **Zettlemoyer L.** and **Stoyanov V.** (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv preprint arXiv:1907.11692.

Marelli M., **Bentivogli L.**, **Baroni M.**, **Bernardi R.**, **Menini S.** and **Zamparelli R.** (2014). SemEval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. In *International Workshop on Semantic Evaluation*.

Mikolov T., **Chen K.**, **Corrado G.** and **Dean J.** (2013). Efficient Estimation of Word Representations in Vector Space. arXiv preprint arXiv:1301.3781.

Miller G. A. (1992). Wordnet: A lexical database for English. *Communications of the ACM* **38**, 22–32. Prague.

Mueller J. and **Thyagarajan A.** (2016). Siamese recurrent architectures for learning sentence similarity. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI'16)*. AAAI Press, pp. 2786–2792.

Nguyen X.-P., **Joty S.**, **Hoi S.** and **Socher R.** (2019). Tree-structured attention with hierarchical accumulation. In *International Conference on Learning Representations*.

Niwattanakul S., **Singthongchai J.**, **Naenudorn E.** and **Wanapu S.** (2013). Using of Jaccard coefficient for keywords similarity. In *Proceedings of the International Multiconference of Engineers and Computer Scientists*. Vol. 1, pp. 380–384.

Opitz J., **Daza A.** and **Frank A.** (2021). Weisfeiler-Leman in the BAMBOO: Novel AMR graph metrics and a benchmark for AMR graph similarity. *Transactions of the Association for Computational Linguistics* **9**, 1425–1441.

https://doi.org/10.1017/nlp.2026.10025 Published online by Cambridge University Press

Pagliardini M., **Gupta P.** and **Jaggi M.** (2018). Unsupervised learning of sentence embeddings using compositional n-gram features. In *Proceedings of NAACL-HLT 2018*. New Orleans, Louisiana, pp. 528–540.

Pennington J., **Socher R.** and **Manning C.D.** (2014). GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 1532–1543.

Reimers N. and **Gurevych I.** (2019). Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*. Hong Kong, China, pp. 3982–3992.

Richard S., **Perelygin A.**, **Wu J.Y.**, **Chuang J.**, **Manning C.D.**, **Ng A.Y.** and **Potts C.** (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Seattle, Washington, USA, pp. 1631–1642.

Richardson R. and **Smeaton A.F.** (1995). Using Wordnet in a Knowledge-based Approach to Information Retrieval. Technical Report CA-0395, Dublin City University, Dublin, Ireland.

Rogers A., **Boyd-Graber J.** and **Okazaki N.** (2023). Opinion tree parsing for aspect-based sentiment analysis. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 7971–7984.

Sachan D.S., **Zhang Y.**, **Qi P.** and **Hamilton W.** (2021). Do syntax trees help pre-trained transformers extract information? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 2647–2661.

Santos J., **Alves A.** and **Oliveira H.G.** (2020). Leveraging on semantic textual similarity for developing a Portuguese dialogue system. In *International Conference on Computational Processing of the Portuguese Language*, pp. 131–142.

Shao Y. (2017). Hcti at semeval-2017 task 1: Use Convolutional neural network to evaluate semantic textual similarity. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pp. 130–133.

Shen Y., **Lin Z.**, **wei Huang C.** and **Courville A.** (2018). Neural language modeling by jointly learning syntax and lexicon. In *International Conference on Learning Representations*.

Shen Y., **Tan S.**, **Sordoni A.** and **Courville A.** (2019). Ordered neurons: Integrating tree structures into recurrent neural networks. In *International Conference on Learning Representations*.

Socher R., **Karpathy A.**, **Le Q.V.**, **Manning C.D.** and **Ng A.Y.** (2014). Grounded compositional semantics for finding and describing images with sentences. *Transactions of the Association for Computational Linguistics* **2**(2014), 207–218,

Solomon K. and **Leibler R.A.** (1951). On information and sufficiency. *The Annals of Mathematical Statistics* **22**, 79–86.

Srivastava N., **Salakhutdinov R.R.** and **Hinton G.E.** (2013). Modeling Documents with Deep Boltzmann Machines. In *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence (UAI)*, pp. 616–624.

Sutton R.S., **McAllester D.**, **Singh S.** and **Mansour Y.** (2000). Policy gradient methods for reinforcement learning with function approximation. In *Neural Information Processing Systems*, pp. 1057–1063.

Sutton R.S., **McAllester D.A**, **Singh S.P.** and **Mansour Y.** (2000). Policy gradient methods for reinforcement learning with function approximation. In *Neural Information Processing Systems*, pp. 1057–1063.

Tai K.S., **Socher R.** and **Manning C.D.** (2015). Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of The 53rd Annual Meeting of the Association for Computational Linguistics*, pp. 1556–1566.

Tang A., **Deléger L.**, **Bossy R.**, **Zweigenbaum P.** and **Nédellec C.** (2022). Do syntactic trees enhance bidirectional encoder representations from transformers (BERT) models for chemical–drug relation extraction? *Database* **2022**, baac070.

Tang P., **Gao J.**, **Zhang L.** and **Wang Z.** (2023). Efficient and interpretable compressive text summarisation with unsupervised dual-agent reinforcement learning. In *Proceedings of The Fourth Workshop on Simple and Efficient Natural Language Processing (SustaiNLP)*, pp. 227–238.

Tharindu R., **Orasan C.** and **Mitkov R.** (2019). Semantic textual similarity with siamese neural networks. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*. Varna, Bulgaria, pp. 1004–1011.

Ting J., **Huang S.**, **Luan Z.**, **Wang D.** and **Zhuang F.** (2024). Scaling sentence embeddings with large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*. Miami, Florida, USA, pp. 3186–3196.

Vaswani A., **Shazeer N.**, **Parmar N.**, **Uszkoreit J.**, **Jones L.**, **Gomez A.N.**, **Kaiser Ł.** and **Polosukhin I.** (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, Vol. 30, pp. 5998–6008.

Wang H. and **Yu D.** (2023). Going beyond sentence embeddings: A token-level matching algorithm for calculating semantic textual similarity. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*. Vol. 2: Short Papers, pp. 563–570.

Wang L., **Ding Z.**, **Han S.**, **Han J.-J.**, **Choi C.** and **Fu Y.** (2019). Generative correlation discovery network for multi-label learning. In *2019 IEEE International Conference on Data Mining (ICDM)*, pp. 588–597.

Wang L., **Liu Y.**, **Qin C.**, **Sun G.** and **Fu Y.** (2020). Dual relation semi-supervised multi-label learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 6227–6234.

Wang M., **Smith N.A.** and **Mitamura T.** (2007). What is the Jeopardy model? A quasi-synchronous grammar for QA. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Prague, pp. 22–32.

https://doi.org/10.1017/nlp.2026.10025 Published online by Cambridge University Press

Wang R., **Tang D.**, **Duan N.**, **Wei Z.**, **Huang X.**, **Cao C.**, **Jiang D.** and **Zhou M.** (2021). K-adapter: infusing knowledge into pre-trained models with adapters. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 1405–1418.

Wang W.Y., **Li J.** and **He X.** (2018). Deep Reinforcement Learning for NLP. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics-Tutorial Abstracts*, pp. 19–21.

Xianming L. and **Jing L.** (2024). AoE: Angle-optimized embeddings for semantic textual similarity. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Bangkok, Thailand, pp. 1825–1839.

Xu Y., **Mou L.**, **Li G.**, **Chen Y.**, **Peng H.** and **Jin Z.** (2015). Classifying relations via long short term memory networks along shortest dependency paths. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1785–1794.

Xu Z., **Guo D.**, **Tang D.**, **Su Q.**, **Shou L.**, **Gong M.**, **Zhong W.**, **Quan X.**, **Duan N.** and **Jiang D.** (2021). Syntax-enhanced pre-trained model. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pp. 5412–5422.

Yang Y., **Tong Y.**, **Ma S.** and **Deng Z.-H.** (2016). A position encoding convolutional neural network based on dependency tree for relation classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas, pp. 65–74.

Yang Y., **Yih W.-T.** and **Meek C.** (2015). WikiQA: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 2013–2018.

Yang Y., **Yuanc S.**, **Cera D.**, **Konga S.-Y.**, **Constanta N.**, **Pilarc P.**, **Gea H.**, **Sunga Y.-H.**, **Stropea B.** and **Kurzweil R.** (2018). Learning semantic textual similarity from conversations. In *Proceedings of the 3rd Workshop on Representation Learning for NLP*. Melbourne, Australia, pp. 164–174.

Yehudai A., **Choshen L.**, **Fox L.** and **Abend O.** (2022). Reinforcement learning with large action spaces for neural machine translation. In *Proceedings of the 29th International Conference on Computational Linguistics*, pp. 4544–4556.

Yin W. and **Schütze H.** (2015). Convolutional neural network for paraphrase identification. In *Human Language Technologies: The 2015 Annual Conference of the North American Chapter of the ACL*. Denver, Colorado, pp. 901–911.

Yogatama D., **Blunsom P.**, **Dyer C.**, **Grefenstette E.** and **Ling W.** (2017). Learning to compose words into sentences with reinforcement learning. In *Proceedings of International Conference on Learning Representations*.

Zhang D., **Xiao W.**, **Zhu H.**, **Ma X.** and **Arnold A.O.** (2022). Virtual augmentation supported contrastive learning of sentence representations. In *Findings of the Association for Computational Linguistics: ACL 2022*. Dublin, Ireland, pp. 2430–2440.

Zhang S., **Wang L.**, **Xiao X.** and **Wu H.** (2022). Syntax-guided contrastive learning for pre-trained language model. In *Findings of the Association for Computational Linguistics: ACL 2022*. Dublin, Ireland, pp. 2430–2440.

Zhang T., **Huang M.** and **Zhao L.** (2018). Learning structured representation for text classification via reinforcement learning. In *Proceedings of Thirty-Second AAAI Conference on Artificial Intelligence*, pp. 6053–6060.

Zhang Z., **Wu Y.**, **Zhou J.**, **Duan S.**, **Zhao H.** and **Wang R.** (2020). Sg-net: Syntax-guided machine reading comprehension. In *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34, pp. 9636–9643.

Cite this article: Wu H, Huang D and Lin X (2026). Skeleton-based approach for semantic textual similarity. *Natural Language Processing* **32**, 430–450. https://doi.org/10.1017/nlp.2026.10025

https://doi.org/10.1017/nlp.2026.10025 Published online by Cambridge University Press