# Sentence Semantic Matching Based on 3D CNN for Human–Robot Language Interaction



WENPENG LU and RUI YU, School of Computer Science and Technology, Qilu University of Technology (Shandong Academy of Sciences)
SHOUJIN WANG, Department of Computing, Macquarie University, Australia
CAN WANG, School of Information and Communication Technology, Griffith University, Australia
PING JIAN and HEYAN HUANG, School of Computer Science and Technology, Beijing Institute of Technology, China

The development of cognitive robotics brings an attractive scenario where humans and robots cooperate to accomplish specific tasks. To facilitate this scenario, cognitive robots are expected to have the ability to interact with humans with natural language, which depends on **natural language understanding** (**NLU**) technologies. As one core task in NLU, **sentence semantic matching** (**SSM**) has widely existed in various interaction scenarios. Recently, deep learning–based methods for SSM have become predominant due to their outstanding performance. However, each sentence consists of a sequence of words, and it is usually viewed as **one-dimensional** (**1D**) text, leading to the existing available neural models being restricted into 1D sequential networks. A few researches attempt to explore the potential of 2D or 3D neural models in text representation. However, it is hard for their works to capture the complex features in texts, and thus the achieved performance improvement is quite limited. To tackle this challenge, we devise a novel **3D CNN-based SSM** (**3DSSM**) method for human–robot language interaction. Specifically, first, a specific architecture called feature cube network is designed to transform a 1D sentence into a multi-dimensional representation named as semantic feature cube. Then, a 3D CNN module is employed to learn a semantic representation for the semantic feature cube by capturing both the local features embedded in word representations and the sequential information among successive words in a sentence. Given a pair of sentences, their representations are concatenated together to feed into another 3D CNN to capture the interactive features between them to generate the final matching representation. Finally, the semantic matching degree is judged with the sigmoid function by taking the learned matching representation as the input. Extensive experiments on two real-world datasets demonstrate that 3DSSM is able to achieve comparable or even better performance over the state-of-the-art competing methods.

The research work is partly supported by National Key R&D Program of China under Grant No. 2018YFC0831700, National Natural Science Foundation of China under Grants No. 61502259 and No.61751201, and Key Program of Science and Technology of Shandong Province under Grants No. 2020CXGC010901 and No. 2019JZZY020124.

Authors' addresses: W. Lu (corresponding author) and R. Yu, School of Computer Science and Technology, Qilu University of Technology (Shandong Academy of Sciences), Daxue Road, Jinan, China, 250353; emails: lwp@qlu.edu.cn, rui.yu1996@foxmail.com; S. Wang, Department of Computing, Macquarie University, Balaclava Road, Sydney, Australia, NSW 2109; email: shoujin.wang@mq.edu.au; C. Wang, School of Information and Communication Technology, Griffith University, Parklands Dr, Southport, Gold Coast, Australia, QLD 4222; email: can.wang@griffith.edu.au; P. Jian and H. Huang, School of Computer Science and Technology, Beijing Institute of Technology, South Street, Zhongguancun, Beijing, China, 100081; emails: {pjian, hhy63}@bit.edu.cn.



1533-5399/2021/06-ART98 $15.00
https://doi.org/10.1145/3450520

CCS Concepts: • **Computing methodologies** → **Discourse, dialog and pragmatics**; *Neural networks*; • **Human-centered computing** → *Interaction techniques;*

Additional Key Words and Phrases: Sentence semantic matching, 3D CNN, semantic feature cube, human–robot interaction, representation learning

**ACM Reference format:**
Wenpeng Lu, Rui Yu, Shoujin Wang, Can Wang, Ping Jian, and Heyan Huang. 2021. Sentence Semantic Matching Based on 3D CNN for Human–Robot Language Interaction. *ACM Trans. Internet Technol.* 21, 4, Article 98 (June 2021), 24 pages.
https://doi.org/10.1145/3450520

## 1 INTRODUCTION

In recent years, with the development of cognitive robotics, more and more robots have been widely applied into human daily activities, which cooperate together with humans to accomplish some specific tasks [17, 28]. Nowadays, most robots are controlled with graphical user interfaces, keyboards, mouses, languages and gestures, and so on. Language interaction is the most attractive human–robot interface because it is intuitive and convenient for humans in real-world cases [18, 27]. With language interaction, the instructions from humans could be given to robots directly and the responses from robots could be sent to humans in a more natural and simpler mode [34]. The implementation of human–robot language interaction heavily depends on the support of natural language understanding technologies. For an ideal cognitive robot, it should be able to understand the meaning of human language, perform the correct operations, and response to humans with reasonable sentences for further interactions. Therefore, how to accurately understand and identify semantic information embedded in interaction sentences is critical yet challenging in cognitive robotics [17].

As one of the core technologies in natural language understanding, **sentence semantic matching (SSM)** plays an important role in human–robot language interaction. SSM aims to model two sentences and identify the semantic relations between them, which directly affects a series of downstream tasks related to human–robot interactions [30, 57]. To be specific, for the question answering system in an intelligent customer service robot, SSM is utilized to model user questions, candidate answers and standard questions, and judge their matching degrees to select proper answers [2, 59]. For the news recommendation in a personalized news website engine, SSM is employed to encode the titles of the news clicked by users and those of candidate news, and evaluate their correlations to recommend the suitable news for users [47, 48]. For the natural language inference in an automated reasoning robot, SSM is used to model two sentences and judge whether the hypothesis sentence could be inferred from the premise one [32]. It is obvious that SSM is critical and significant for human–robot language interactions because of its indispensable role in semantic modeling and analysis.

Due to the complexity and diversity of natural language, SSM is still a challenging task. As shown in Figure 1, even though two sentences are composed of identical words, they have different or even opposite meanings because of the different sequential orders of words. Speaker A means that he doesn't have money to spend. However, Speaker B means that he has money, but doesn't want to spend it. Apparently, the sequential order of words in a sentence has great influence on the semantic representation of a sentence. Hence, both the inherent meanings carried by words and the semantic information embedded in sequential order should be considered. This brings huge challenges to SSM task.

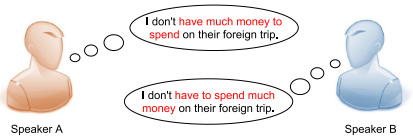



Fig. 1. This is an example of sentences consisting of identical words that have different meanings due to the difference of the sequential order of words.

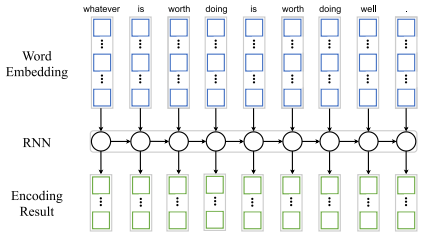



Fig. 2. This is an example of a sentence that is encoded by RNN.

Recent years have seen increasing research on learning the representation of texts with neural networks, which shows the fundamentals for modeling sentence semantics appropriately [13, 31, 41, 49]. **Recurrent neural network** (**RNN**) and **convolutional neural network** (**CNN**) play dominant roles in sentence representation, which are widely applied in various SSM models [35, 47, 57]. Due to the powerful ability in capturing long-distance dependency and sequential information, RNN is popular in text representation learning. A series of variants of RNN have been proposed, such as **long- and short-term memory** (**LSTM**) [12], **bidirectional LSTM** (**BiLSTM**) [40], and **gated recurrent unit** (**GRU**) [6]. As shown in Figure 2, when encoding a sentence, RNN calculates its current state according to the input at the current timestep and the state from the previous timestep. With the chain-like structure, RNN can naturally remember most of the information on all timesteps through the whole sentence. Though RNN is good at capturing the sequential information and long-distance dependency, it is not perfect. The superiority of RNN originates from its mathematical foundation similar to Markov chain. However, it doesn't fully mine and utilize the past state and current input information, which limits the performance. In addition, RNN is not adept at capturing the local features embedded in texts. It is necessary to explore more sophisticated neural structures to capture the important sequential information and local features in texts.

Except for RNN-based models, CNN-based ones present increasingly promising power in representing texts [35]. As a text consists of a sequence of words, it is **one-dimensional** (**1D**). Thus, most of the popular CNN models for **natural language processing** (**NLP**) tasks are 1D CNN-based models [7]. As shown in Figure 3, when encoding a sentence, CNN utilizes convolution kernels to learn and extract the most useful information for sentence representation. In addition, CNN is able to stack multiple convolution layers to capture deeper features and generate a more ideal representation. However, 1D CNN works as $N$-gram models on 1D sequences, which fails to capture more interaction patterns across different dimensions of texts [36]. Considering the superiority of 2D CNN on learning sophisticated features, some work first transforms 1D texts into 2D representations similar to images, and then employs 2D CNN to capture the rich interactions between different dimensions and granularities [16, 22, 36]. Though 2D CNN could achieve better performance by modeling more interaction patterns, it is still troubled by the problem of missing sequential information, and thus has not yet significantly outperformed 1D CNN. Although CNN could stack multiple convolution layers, it only captures more abstract features, which neglects the sequential information embedded in texts.

To solve the above issues confusing the existing RNN- and CNN-based models, we seek to take the advantages of 3D CNN to fully represent both the local features and the sequential information between successive words in a sentence. Though 3D CNN has been widely applied on learning the

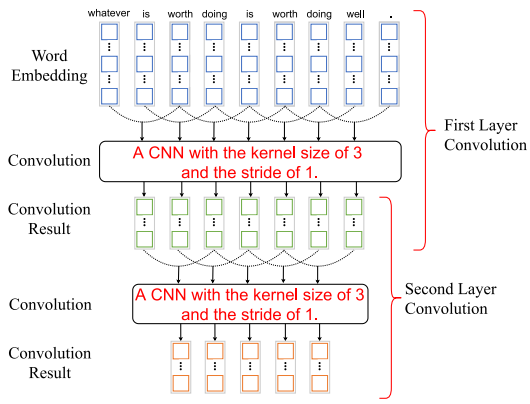



Fig. 3. This is an example of a sentence that is encoded by 1D CNN.

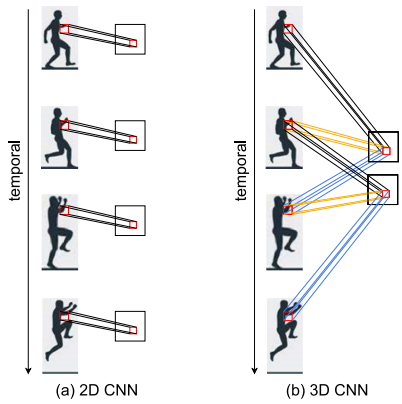



Fig. 4. 2D CNN and 3D CNN for encoding temporal features in action recognition.

temporal representation in video classification and action recognition [1, 9, 15, 52, 55], rare work exists on text representation. In order to illustrate the advantages of 3D CNN more clearly, we first explain it with an example of video-based action recognition. As shown in Figure 4(a), when 2D CNN is adopted, the still image on each frame in the video is separately encoded by 2D convolution kernel. 2D CNN can only capture the local features in the image and fails to model the temporal evolution features between consecutive frames, which inevitably hurts the performance of action recognition. However, as shown in Figure 4(b), once 3D CNN is employed, it can fully capture both the local features in each frame and the temporal information between consecutive frames. The two complementary features are beneficial to video action representation. Inspired by the work of 3D CNN on video processing, we introduce 3D CNN into text representation and propose a novel **3D CNN-based sentence semantic matching** (**3DSSM**) method for modeling language interaction in cognitive robotics. 3DSSM generates multi-dimensional representations for 1D sentences and applies 3D CNN to capture the sophisticated information including local and sequential order features embedded in sentences. Specifically, 3DSSM first implements a specific neural architecture called feature cube network, which is employed to transform the inputted 1D sentence into multi-dimensional representation named as semantic feature cube. Then, a 3D CNN module is utilized to handle the semantic feature cube to capture the local features embedded in different dimensions of words and sequential order information between successive words in the sentence, and generate a semantic representation. Next, both representations from two sentences are concatenated together, which are further encoded by another 3D CNN and max-pooling operations to capture the more complicated interactive features to generate the final matching representation. Finally, the semantic matching degree of the pair of sentences is judged with the sigmoid function by taking the learned matching embedding as the input. The experiments carried out on two real-world public datasets confirm the effectiveness of the proposed model, whose performance is superior or comparable to the representative and state-of-the-art methods.

The main contributions of this work can be summarized as follows

- To the best of our knowledge, this work is the first to utilize 3D CNN to model sentence pairs for semantic matching task in human–robot language interactions. We propose a novel neural architecture called 3DSSM, which consists of a multi-granularity embedding module, feature cube network, 3D CNN module, and label prediction module. With 3D CNN, 3DSSM collectively captures both the local features embedded in words and the sequential order information between successive words, and learns the interactive features between sentences.

- We devise a new neural structure to construct semantic feature cube, which transforms 1D sentences into multi-dimensional representations. Semantic feature cube establishes the foundation for applying 3D CNN to encode 1D sentences, which is critical for simultaneously capturing the local features and the sequential information in sentences. As far as we know, there is rare research work similar to this.
- Extensive experiments are carried out on two real-world public datasets. The experimental results verify the effectiveness of 3DSSM, which shows comparable or better performance over the state-of-the-art methods. The source code of our work is publicly available on GitHub.[1]

The reminder of the article is structured as follows. Some related work is briefly reviewed in Section 2. Section 3 demonstrates the problem statement, and explains the architecture of 3DSSM together with its core modules in detail. Section 4 reports the experiments and results. Finally, the article is concluded in Section 5.

## 2 RELATED WORK

In recent years, with the development of 5G/6G networks and cognitive robotics, cognitive robots have been widely applied in human activities. How to implement satisfied human–robot interactions is crucial for cognitive robotics, which is related with a series of technologies, such as computer vision [10, 19, 54], image–image matching [8, 26, 38], image–text matching [29, 51, 53], text–text matching [30, 35], and so on. Although robots have improved the ways in which humans work and live, the human–robot interactions are traditionally restricted into several common methods, such as graphical user interfaces, keyboards and languages, and so on [17]. Among the existing methods, language interaction is the most appealing way, as it is the most natural mode of human communications in daily activities [18, 34]. The application of human–robot language interaction requires that the robots have the ability to correctly capture the meanings of human languages, accurately understand the instructions from humans, and rapidly make the right responses. Some studies have been done on implementing more intelligent and convenient language interface for robots.

To realize a scheme for seamless human-device interaction, Eugenio et al. proposed an architecture involving two distinct entities, i.e., the controller and the set of devices [39], where the controller utilized NLU technologies to interpret the user intention from natural language expressions, and arranged the devices to make the desired actions. To reason the motions and goals implied in human instructions, Paxton et al. presented an architecture for converting a natural language command to a series of intermediate motions for robots to execute [37]. For improving language understanding for a robot agent, Thomason et al. proposed to utilize human–robot dialogs to enhance the ability of understanding human language, where the agent parsed sentences to analyze the potential semantic meanings [45]. These mentioned works have effectively promoted the development of human–robot interaction; they generally strive for understanding the intention of human language, which is critical for cognitive robots to interact with humans [18, 34].

SSM task is to judge whether the meanings or intentions of two sentences are consistent. In this article, we focus on SSM for human–robot language interactions, which is the foundation for a series of downstream applications, such as intelligent customer service robot [2], automated reasoning robot [32], and so on. With the prevalence of deep learning, recent years have seen increasing research on SSM task with RNN-based and CNN-based neural models.

[1] https://github.com/yurui12138/3DSSM.

### 2.1 RNN-Based Models

RNN is popular in the NLP community due to its powerful ability for capturing the sequential information. Some classical variants are proposed to further strengthen the performance of RNN-based models, such as LSTM, BiLSTM, and GRU.

Xie and Ma proposed a **dual-view variational autoencoder** for text matching called **DV-VAE**, which utilized a BiLSTM as the encoder, a dilated CNN as the decoder, followed by the interaction matcher based on another BiLSTM [50]. The experimental results on three datasets indicated that DV-VAE consistently outperformed the competitors. Liu et al. put forward an **original semantics-oriented attention and deep fusion network** for sentence matching called **OSOA-DFN**, where a BiLSTM was used to encode sentences in input layer, and another BiLSTM was applied to gather interactive features in deep fusion layer [23]. Lu et al. proposed a deep hierarchical encoding model for SSM task, which utilized two BiLSTMs to process the sentences twice to construct enhanced representations [30]. Thorne et al. presented an approach to generate token-level explanations for natural language inference, which adopted the same LSTM to encode the premise and hypothesis [46].

Though RNN-based models have achieved good performance, they are unable to effectively capture the local features embedded in sentences. In addition, though RNN is famous for its sequential encoding ability, it is not perfect because it doesn't fully mine the past state and current input information.

### 2.2 CNN-Based Models

Another group of representative work on SSM task belongs to CNN-based models. According to the difference of convolution kernels, CNN-based models are categorized into 1D CNN, 2D CNN, and 3D CNN.

*2.2.1 1D CNN-Based Models.* As a sentence consists of a sequence of words, it is viewed as a one-dimensional text. Therefore, the most typical model for SSM task is 1D CNN.

Li et al. proposed a hybrid model combining local and global features with a siamese network for SSM, which first utilized 1D convolution kernels to extract the most important multi-scale semantic features, followed by bidirectional GRU to capture global features [20]. Chen et al. presented the architecture of multi-channel information crossing for text matching, which employed a multi-channel 1D CNN to capture text features at varied granularities, and imposed an attention mechanism on feature interactions to achieve better matching performance [3]. Zhang et al. proposed a multi-granularity neural network for answer sentence selection, which utilized 1D CNN to generate the representations of sentences on $N$-gram channels, and implemented a weighting scheme to combine the channels at different granularities [56].

In the above work, the superiority of 1D CNN to capture the local features between words or $N$-grams has been explored. However, it fails to capture more interaction patterns embedded in different dimensions of texts [36].

*2.2.2 2D CNN-Based Models.* Aiming at bridging the gaps of 1D CNN-based models, some researchers attempt to employ 2D CNN to learn deeper features.

Inspired by the success of 2D CNN on image processing, Pang et al. presented an approach to model text matching task as the problem of image recognition, which constructed a matching matrix similar to an image and utilized a 2D CNN to capture rich patterns embedded in the matrix [36]. Song et al. designed a positional convolutional neural network for enhancing text matching, which first constructed a position-similarity matrix, utilized a 2D CNN to capture the positional information at the phrase and sentence levels, then aggregated positional information

from multiple perspectives to generate the final matching representation [42]. Liu et al. proposed to combine attention-based bidirectional GRU and 2D CNN to encode a document, which first utilized bidirectional GRU to learn the 2D matrix representation of the document, and then applied 2D CNN to capture more interaction features between sentences [22].

Though 2D CNN can effectively capture the deep interaction patterns between different dimensions or granularities, it is restricted by the problem of missing sequential information, failing to capture the deep features embedded in sequential orders of texts.

*2.2.3 3D CNN-Based Models.* In order to tackle the problems confusing the above RNN- and CNN-based models, it is necessary to devise a novel neural architecture to simultaneously capture the local features and sequential information between successive words in sentences.

There exists some work based on 3D CNN on video representation or action recognition [1, 15]. Alyrac et al. proposed to leverage a 3D CNN architecture to capture both the features in the still image on each frame and the motion features at multiple temporal scales, which achieved satisfied performance on video representation [1]. Jiang et al. implemented a dual 3D CNN model for real-time action recognition, which utilized two 3D CNN modules to learn multi-resolution spatio-temporal information from a video [15].

Inspired by the above work of 3D CNN on video processing, we propose a novel **3D CNN-based sentence semantic matching** (**3DSSM**) method for SSM task in this article. We first design a special neural architecture to transform the original 1D sentences into multi-dimensional representations, i.e., semantic feature cubes. Then, two 3D CNN components are employed to encode the semantic feature cubes and capture their interactive features to generate the final matching representation, followed by the sigmoid function to judge the matching degree of the pair of sentences.

## 3 PROPOSED 3DSSM MODEL

### 3.1 Problem Statement

Before introducing the detailed implementation of our model, we first formulate the problem. Given two sentences $S^a = \{g_1^a, g_2^a, \ldots, g_n^a\}$ and $S^b = \{g_1^b, g_2^b, \ldots, g_m^b\}$, where $g_i^a$ and $g_j^b$ denote the $i^{\text{th}}$ and $j$th words in the sentences, and $n$ and $m$ indicate the total number of words in the sentences, respectively, the goal of SSM is to learn a classifier $\xi$, which is able to accurately predict whether the meanings expressed by the sentences $S^a$ and $S^b$ are consistent, i.e., $y = \xi(S^a, S^b)$. Here, $y$ is the label indicating the matching relation between $S^a$ and $S^b$.

### 3.2 Architecture of 3DSSM Model

Our sentence semantic matching model 3DSSM learns to predict the matching scores of sentence pairs by training an end-to-end neural network with 3D convolutional kernels. As shown in Figure 5, 3DSSM consists of four modules: multi-granularity embedding module, feature cube network, 3D CNN module, and label prediction module, which are marked with ①, ②, ③, and ④, respectively. The multi-granularity embedding module is responsible for converting sentences at character and word granularity into their embeddings, respectively. The feature cube network is designed to transform the embeddings of sentences into multi-dimensional representations called semantic feature cubes by stacking multiple BiLSTM layers and concatenating their outputs to provide the foundation for the application of 3D CNN. The 3D CNN module first encodes the semantic feature cubes to learn the local features embedded in different dimensions and granularities of words and the sequential features between successive words in sentences, and then generates sentence representations. Then, both representations of two sentences are concatenated together

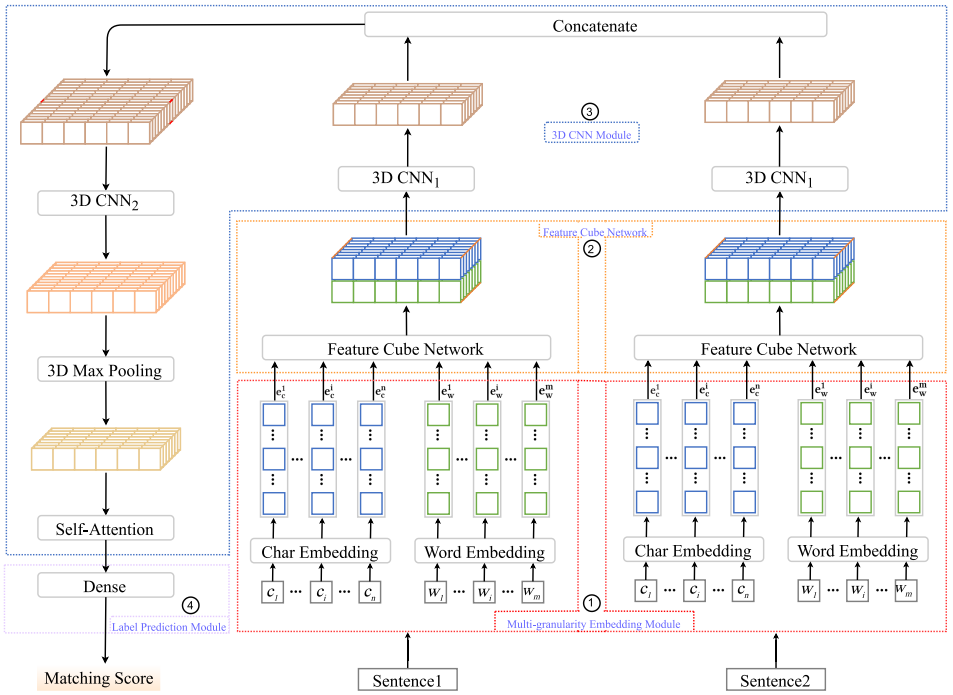



Fig. 5. Model architecture of 3DSSM.

and further encoded with another 3D CNN, max-pooling and self-attention mechanism [21] to generate a final matching representation for the pair of sentences. According to the matching tensor outputted by the 3D CNN module, the label prediction module predicts the matching score of the sentence pair and assigns the corresponding label for it.

### 3.3 Multi-Granularity Embedding Module

As texts at different granularities contain different information, 3DSSM accepts the input sentences segmented at character and word level, respectively, so as to capture semantic features as much as possible. As shown in Figure 5, the multi-granularity embedding module includes two sub-modules, which aim to convert the sentence into the representation at character and word granularity, respectively.

The first sub-module is a character embedding sub-module, which is responsible for transforming a sentence into an embedding at character granularity. The original sentence at character level is denoted as $S_c = \{c_1, c_2, \ldots, c_n\}$, where $n$ is the total number of characters in the sentence. This sub-module converts $S_c$ to an embedding $\mathbf{E}_\mathbf{c}^\mathbf{S} = [\mathbf{e}_\mathbf{c}^\mathbf{1}, \mathbf{e}_\mathbf{c}^\mathbf{2}, \ldots, \mathbf{e}_\mathbf{c}^\mathbf{n}]$ according to a pre-trained character embedding matrix $\mathbf{M}_c \in \mathbb{R}^{N_c \times D_c}$, where $N_c$ is the size of character vocabulary and $D_c$ refers to the dimension of character embedding.

The second one is a word embedding sub-module, which is dedicated to converting a sentence into an embedding at word granularity. Similar to the former sub-module, the sentence at word level is marked as $S_w = \{w_1, w_2, \ldots, w_m\}$, where $m$ is the total number of words in the sentence. This sub-module transforms $S_w$ into an embedding $\mathbf{E}_\mathbf{w}^\mathbf{S} = [\mathbf{e}_\mathbf{w}^\mathbf{1}, \mathbf{e}_\mathbf{w}^\mathbf{2}, \ldots, \mathbf{e}_\mathbf{w}^\mathbf{m}]$ according to a pre-trained word embedding matrix $\mathbf{M}_w \in \mathbb{R}^{N_w \times D_w}$, where $N_w$ is the size of word vocabulary and $D_w$ refers to the dimension of word embedding.

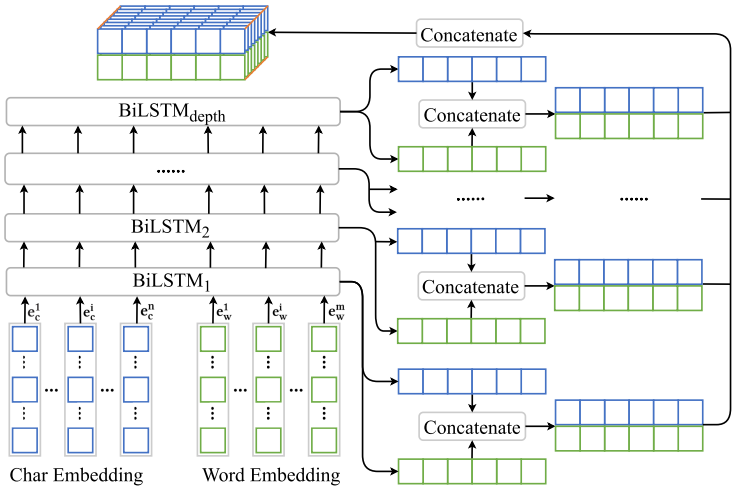



Fig. 6. Feature cube network.

**ALGORITHM 1:** Constructing semantic feature cube.

**Input:**
The character embedding of the sentence, $E_c^S$;
The word embedding of the sentence, $E_w^S$;
The depth of feature cube network, *depth*;

**Output:**
The representation of semantic feature cube for the sentence, *cube*;

**for** $i = 1$ to *depth* **do**
  $E_c^{next} = \text{BiLSTM}(E_c^S)$;
  $E_w^{next} = \text{BiLSTM}(E_w^S)$;
  $cube^i = \text{Concatenate(axis = 2)}(\text{Reshape}(E_c^{next}), \text{Reshape}(E_w^{next}))$;
  $E_c^S = E_c^{next}$;
  $E_w^S = E_w^{next}$;
**end for**
$cube = \text{Concatenate(axis = 4)}(cube^1, cube^2, \ldots, cube^{depth})$;
**return** *cube*;

### 3.4 Feature Cube Network

In order to utilize 3D CNN to simultaneously capture the sophisticated local and sequential features embedded in sentences, it is critical to transform the sentences into the representations which are suitable for 3D CNN to handle. Inspired by the work in video processing [1, 15], our 3DSSM model designs a novel feature cube network to convert the sentences into multi-dimensional representations called semantic feature cubes. As shown in Figure 6, with BiLSTM as the basic component, the semantic cube network first receives the sentence representations at character and word granularity, then stacks multiple BiLSTM components to encode the sentence layer by layer. For the intermediate representations on each layer, the semantic cube network reshapes their dimensions and concatenates them together to generate the semantic feature cube on the corresponding layer. Finally, all cubes are concatenated together to obtain the final semantic feature cube. The entire procedure of the semantic cube network is described as Algorithm 1.

As shown in Figure 6, in the first layer of the feature cube network, the character and word embeddings of a sentence are first encoded by the component $\text{BiLSTM}_1$, which captures the dependency features between characters or words to generate the corresponding embeddings. The embeddings are further reshaped and concatenated together to obtain the first semantic feature cube. Taking the sentence $S$ as an example, the above operations are described with Equation (1):

$$\begin{aligned} \mathbf{r}_c^{1^*} &= BiLSTM_1(\mathbf{E}_c^S, i_c), \quad \mathbf{r}_c^1 = Reshape(\mathbf{r}_c^{1^*}), \\ \mathbf{r}_w^{1^*} &= BiLSTM_1(\mathbf{E}_w^S, i_w), \quad \mathbf{r}_w^1 = Reshape(\mathbf{r}_w^{1^*}), \\ \mathbf{cube}^1 &= Concatenate(axis = 2)(\mathbf{r}_c^1, \mathbf{r}_w^1), \end{aligned} \tag{1}$$

where $i_c$ and $i_w$ are the positions of the $i$th character and the $i$th word, and $\mathbf{E}_c^S$ and $\mathbf{E}_w^S$ are the character and word embeddings of the sentence $S$, respectively. To facilitate the construction of the semantic feature cube, for the BiLSTM component, its states on all timesteps are reserved. Thus, the outputs of $\text{BiLSTM}_1$, i.e., $\mathbf{r}_c^{1^*}$ and $\mathbf{r}_w^{1^*}$, are 3D tensors with the shape of ($batch\_size$, $time\_steps$, $output\_dim$). In order to simulate the 3D CNN operations in video processing, we reshape the two 3D tensors to 5D tensors, i.e., $\mathbf{r}_c^1$ and $\mathbf{r}_w^1$, whose shape is ($batch\_size$, $time\_steps$, $granularity\_dim$, $output\_dim$, $depth\_dim$). Here, both $granularity\_dim$ and $depth\_dim$ are set to 1, i.e., the shape of 5D tensors is ($batch\_size$, $time\_steps$, 1, $output\_dim$, 1). Further, the two 5D tensors $\mathbf{r}_c^1$ and $\mathbf{r}_w^1$ are concatenated together on the $granularity\_dim$ dimension to construct the first semantic feature cube, i.e., $\mathbf{cube}^1$, whose shape is ($batch\_size$, $time\_steps$, 2, $output\_dim$, 1).

In the second layer of the feature cube network, the second semantic feature cube is constructed with a method similar to the first one. To be specific, the outputs of the component $\text{BiLSTM}_1$ at different granularities are further encoded by another component $\text{BiLSTM}_2$, whose outputs are reshaped into 5D tensors ($batch\_size$, $time\_steps$, 1, $output\_dim$, 1). The tensors are concatenated on the $granularity\_dim$ dimension to construct the second semantic feature cube, i.e., $\mathbf{cube}^2$, whose shape is ($batch\_size$, $time\_steps$, 2, $output\_dim$, 1). The detailed operations are described with Equation (2):

$$\begin{aligned} \mathbf{r}_c^{2^*} &= BiLSTM_2(\mathbf{r}_c^{1^*}, i_c), \quad \mathbf{r}_c^2 = Reshape(\mathbf{r}_c^{2^*}), \\ \mathbf{r}_w^{2^*} &= BiLSTM_2(\mathbf{r}_w^{1^*}, i_w), \quad \mathbf{r}_w^2 = Reshape(\mathbf{r}_w^{2^*}), \\ \mathbf{cube}^2 &= Concatenate(axis = 2)(\mathbf{r}_c^2, \mathbf{r}_w^2), \end{aligned} \tag{2}$$

where the meanings of the symbols are similar to those in Equation (1).

In the following layers of the feature cube network, the operations are similar to those in the second layer. For the $depth_{\text{th}}$ layer, it generates the $depth_{\text{th}}$ semantic feature cube, as described in Equation (3), where the meanings of the symbols are similar to those in Equation (2).

$$\begin{aligned} \mathbf{r}_c^{depth^*} &= BiLSTM_d(\mathbf{r}_c^{depth-1^*}, i_c), \\ \mathbf{r}_c^{depth} &= Reshape(\mathbf{r}_c^{depth^*}), \\ \mathbf{r}_w^{depth^*} &= BiLSTM_d(\mathbf{r}_w^{depth-1^*}, i_w), \\ \mathbf{r}_w^{depth} &= Reshape(\mathbf{r}_w^{depth^*}), \\ \mathbf{cube}^{depth} &= Concatenate(axis = 2)(\mathbf{r}_c^{depth}, \mathbf{r}_w^{depth}). \end{aligned} \tag{3}$$

Finally, all semantic feature cubes are concatenated together on the $depth\_dim$ dimension to generate the final semantic feature cube, i.e., $\mathbf{cube}$, whose shape is ($batch\_size$, $time\_steps$, 2, $output\_dim$, $depth$). The operations are described with Equation (4):

$$\mathbf{cube} = Concatenate(axis = 4)(\mathbf{cube}^1, \mathbf{cube}^2, \ldots, \mathbf{cube}^{depth}). \tag{4}$$

### 3.5 3D CNN Module

Once the semantic feature cube of each sentence is generated, our 3DSSM model utilizes the 3D CNN module to learn the sophisticated features and generate the final matching representation for the pair of sentences. As shown in Figure 5, the 3D CNN module mainly consists of two 3D CNN sub-modules. The first one encodes the semantic feature cube to generate the representation for each sentence. Both representations from two sentences are concatenated together to obtain the initial representation of the sentence pair, which is further encoded by the second 3D CNN sub-module, max-pooling and self-attention mechanism to generate the final matching representation of the sentence pair.

The first 3D CNN sub-module learns the semantic feature cube to capture the local features embedded in different dimensions and granularities of words and the sequential features between successive words to generate an embedding for the sentence. The detailed operations are described with Equation (5):

$$
\begin{aligned}
c^f_{i,j,k} &= ReLU\left(\sum_{i=0}^{m_l-1}\sum_{j=0}^{m_h-1}\sum_{k=0}^{m_d-1}\mathbf{g}^f_{(x_1,y_1,z_1)}\mathbf{cube}_{i:i+x_1-1,j:j+y_1-1,k:k+z_1-1}+b^f_1\right),\\
\mathbf{c}^f_k &= \begin{bmatrix} c^f_{1,1} & \cdots & c^f_{1,((m_l-x_1)/s_{x_1})+1}\\ \vdots & \ddots & \vdots \\ c^f_{((m_h-y_1)/s_{y_1})+1,1} & \cdots & c^f_{((m_h-y_1)/s_{y_1})+1,((m_l-x_1)/s_{x_1})+1}\end{bmatrix},\\
\mathbf{c}^f &= [\mathbf{c}^f_1,\mathbf{c}^f_2,\ldots,\mathbf{c}^f_{(m_d-z_1)/s_{z_1}}],\\
\mathbf{c} &= [\mathbf{c}^1,\mathbf{c}^2,\ldots,\mathbf{c}^f,\ldots,\mathbf{c}^{n-1},\mathbf{c}^n],
\end{aligned}
\tag{5}
$$

where $f$ denotes the $f_{\text{th}}$ convolution kernel; **cube** refers to the semantic feature cube; $m_l$, $m_h$, and $m_d$ are the length, width, and depth of **cube**; $\mathbf{g}^f_{(x_1,y_1,z_1)}$ and $b^f_1$ denote the weight matrix and bias of the $f_{\text{th}}$ convolution kernel; and $c^f_{i,j,k}$ is the convolution result of the $f_{\text{th}}$ kernel at the region starting at position $(i,j,k)$. $s_{x_1}$, $s_{y_1}$, and $s_{z_1}$ denote convolution strides on the three dimensions, $\mathbf{c}^f_k$ denotes the convolution result of the $f_{\text{th}}$ kernel at the $k_{\text{th}}$ depth,. $\mathbf{c}^f$ denotes the convolution result of the $f_{\text{th}}$ kernel at all depths, and $\mathbf{c}$ denotes the convolution result of all kernels on all depths for **cube**.

For a pair of sentences ($S^a$, $S^b$), each sentence is encoded with the first 3D CNN sub-module to generate the corresponding representation, marked as $\mathbf{c}_a$ and $\mathbf{c}_b$, respectively. We concatenate them together on the *depth_dim* dimension to generate the initial representation for the pair of sentences, as shown in Equation (6):

$$
\mathbf{t} = Concatenate(axis = 4)(\mathbf{c}_a, \mathbf{c}_b). \tag{6}
$$

Based on the initial representation $\mathbf{t}$, the second 3D CNN sub-module further captures the deeper interactive features between two sentences to generate the matching representation. The detailed operations are described in Equation. (7):

$$
\begin{aligned}
q^f_{i,j,k} &= ReLU\left(\sum_{i=0}^{r_l-1}\sum_{j=0}^{r_h-1}\sum_{k=0}^{r_d-1}\mathbf{g}^f_{(x_2,y_2,z_2)}\mathbf{t}_{i:i+x_2-1,j:j+y_2-1,k:k+z_2-1}+b^f_2\right),\\
\mathbf{q}^f_k &= \begin{bmatrix} q^f_{1,1} & \cdots & q^f_{1,((r_l-x_2)/s_{x_2})+1}\\ \vdots & \ddots & \vdots \\ q^f_{((r_h-y_2)/s_{y_2})+1,1} & \cdots & q^f_{((r_h-y_2)/s_{y_2})+1,((r_l-x_2)/s_{x_2})+1}\end{bmatrix},
\end{aligned}
\tag{7}
$$

$$\mathbf{q}^f = [\mathbf{q}_1^f, \mathbf{q}_2^f, \ldots, \mathbf{q}_{(r_d - z_2)/s_{z_2}}^f],$$
$$\mathbf{q} = [\mathbf{q}^1, \mathbf{q}^2, \ldots, \mathbf{q}^f, \ldots, \mathbf{q}^{n-1}, \mathbf{q}^n],$$

where $f$ denotes the $f_{\text{th}}$ convolution kernel; $r_l$, $r_h$, and $r_d$ are the length, width, and depth of $\mathbf{t}$, $\mathbf{g}^f_{(x_2, y_2, z_2)}$ and $b_2^f$ refer to the weight matrix and bias of the $f_{\text{th}}$ convolution kernel; and $q^f_{i,j,k}$ denotes the convolution result of the $f_{\text{th}}$ kernel at the region starting at position $(i, j, k)$. $s_{x_2}$, $s_{y_2}$, and $s_{z_2}$ denote convolution strides on the three dimensions; $\mathbf{q}_k^f$ denotes the convolution result of the $f_{\text{th}}$ kernel at the $k_{\text{th}}$ depth; $\mathbf{q}^f$ denotes the convolution result of the $f_{\text{th}}$ kernel at all depths; and $\mathbf{q}$ denotes the convolution result of all kernels at all depths for $\mathbf{t}$.

The matching representation $\mathbf{q}$ of the pair of sentences is further processed by the 3D max-pooling operation to capture the key information, as described in Equation (8):

$$\begin{aligned} p_{i,j,k} &= \max_{0 \le i \le w} \max_{0 \le j \le h} \max_{0 \le k \le d} \mathbf{q}_{i:i+x_3-1, j:j+y_3-1, k:k+z_3-1}, \\ \mathbf{p}_k &= \begin{bmatrix} p_{1,1} & \cdots & p_{1,((w-x_3)/s_{x_3})+1} \\ \vdots & \ddots & \vdots \\ p_{((h-y_3)/s_{y_3})+1,1} & \cdots & p_{((h-y_3)/s_{y_3})+1,((w-x_3)/s_{x_3})+1} \end{bmatrix}, \\ \mathbf{p} &= [\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_{(d-z_3)/s_{z_3}}], \end{aligned} \tag{8}$$

where $x_3$, $y_3$, and $z_3$ denote the length, width, and depth of the pooling window; $s_{x_3}$, $s_{y_3}$ and $s_{z_3}$ are the strides on the three dimensions; $p_{i,j,k}$ denotes the maximum element in the specific pooling region of $\mathbf{q}$; $p_k$ denotes the maximum element at the $k_{\text{th}}$ depth; and $\mathbf{p}$ denotes all maximum elements at all depths for $\mathbf{q}$.

Finally, a self-attention mechanism is employed on the matching representation $\mathbf{p}$ to further capture the important information to generate the final matching representation of the pair of sentences, as described in Equation (9):

$$\begin{aligned} \mathbf{z}_i &= \mathbf{U} tanh(\mathbf{W}_a \mathbf{p}_i), \\ \mathbf{a}_i &= exp(\mathbf{z}_i) / \sum_{j=1}^{N} exp(\mathbf{z}_j), \\ \mathbf{M} &= \sum_{i=1}^{N} \mathbf{a}_i \mathbf{p}_i, \end{aligned} \tag{9}$$

where $\mathbf{U}$ and $\mathbf{W}_a$ are the parameter matrices, $\mathbf{z}_i$ is the projection of $\mathbf{p}_i$, $\mathbf{a}_i$ is the attention weight of $\mathbf{p}_i$, and $\mathbf{M}$ is the final matching representation of the pair of sentences.

### 3.6 Label Prediction Module

After obtaining the final matching representation by the 3D CNN module, our 3DSSM model utilizes a label prediction module to predict the matching score of the pair of sentences. According to the matching degree, the target label is judged. The detailed operations are described in Equation (10):

$$y^{pred} = sigmoid(\mathbf{W}_d \mathbf{M}), \tag{10}$$

where $\mathbf{W}_d$ is the weight matrix of the dense layer, $\mathbf{M}$ is the final matching representation outputted by the 3D CNN module, and $y^{pred}$ is the predicted matching score.

Binary cross-entropy loss is adopted to optimize our model. The loss function is described in Equation (11):

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}[(y^{true})log(y^{pred}) + (1 - y^{true})log(1 - y^{pred})], \tag{11}$$

where $y^{true}$ is the true tag, $y^{pred}$ is the predicted score, and $N$ is the batch size.

## 4 EXPERIMENTS

Extensive experiments on two real-world public datasets are carried out to evaluate the performance of our 3DSSM model by comparing it with those representative and state-of-the-art models. In addition, the importance of each individual component of 3DSSM is investigated by an ablation study. Finally, we make a detailed hyperparameter analysis for 3DSSM.

### 4.1 Datasets and Experimental Settings

The experiments are conducted on two real-world datasets, i.e., BQ [4] and LCQMC [25]. BQ is a Chinese corpus for sentence semantic equivalence identification developed by Chen et al. [4]. BQ consists of 120,000 question sentence pairs, which come from 1-year online logs of bank custom service. It is divided into three groups: a training set containing 100,000 pairs, a validation set containing 10,000 pairs, and a test set containing 10,000 pairs. LCQMC is another large-scale Chinese question matching corpus established by Liu et al. [25], which focuses on semantic intention matching. The corpus consists of 260,068 question pairs collected from Baidu Knows, which includes three parts: 238,766 question pairs for training, 8,802 question pairs for validation, and 12,500 question pairs for test. In order to introduce the datasets intuitively, we randomly select two pairs of sentences from each dataset, as shown in Table 1.

In our experiments, for the multi-granularity embedding module, the dimension $D_c$ of character embedding $\mathbf{M}_c$ and $D_w$ of word embedding matrix $\mathbf{M}_w$ were set to 400. We utilized Word2Vec to train the character and word embeddings on the dataset, respectively, where *window*, *min_count*, *sg*, *workers*, *seed*, and *iter* were set to 5, 1, 1, 4, 1,234, and 25, respectively, and the default settings were adopted for other parameters. In the feature cube network, the output dimension of BiLSTM was set to 300. The best performance was achieved when the depth of feature cube network was set to 3 for BQ, and 4 for LCQMC. The dropout strategy was applied [43], where the dropout rate was set to 0.6 for BQ, and 0.05 for LCQMC. In the 3D CNN module, for the first 3D CNN sub-module for learning semantic feature cube, the number of filters was set to 16 for BQ and LCQMC, the convolution kernel size was set to (5, 2, 4) for BQ and (2, 2, 5) for LCQMC, and the convolution stride was set to (1, 1, 2) for BQ and (1, 1, 1) for LCQMC. For the second 3D CNN sub-module for capturing interactive features, the number of filters was set to 32 for BQ and LCQMC, the convolution kernel size was set to (3, 1, 4) for BQ and (5, 1, 2) for LCQMC, and the convolution stride was set to (2, 1, 2) for BQ and (1, 1, 1) for LCQMC. The window size of 3D max-pooling operation was set to (4, 1, 5) for BQ and (3, 1, 5) for LCQMC, and the stride was set to (1, 1, 3) for BQ and (2, 1, 3) for LCQMC. ReLU was employed as the activation function in the above-mentioned modules [33]. When fitting the model, we employed the binary cross-entropy loss and chose RMSProp as the optimizer with an initial learning rate of 0.0015. All experiments were conducted on a workstation equipped with two Intel Xeon Gold 5118 CPUs, 64 GB memory, and two Nvidia GTX 2080Ti GPU accelerators.

Table 1. Examples of BQ and LCQMC

| Dataset | | Sentence Content | Matching Tag |
|---|---|---|---|
| BQ | Sentence Pair1 | 我需要修改电话号码<br>(En: I need to change my phone number)<br>更改预留手机号码<br>(En: Change the pre-reserved phone number) | 1 |
| | Sentence Pair2 | 啥时候给我打电话<br>(En: When do you call me)<br>我的审核需要多久<br>(En: How long will my audit take) | 0 |
| LCQMC | Sentence Pair3 | 为什么南极比北极冷?<br>(En: Why is the South Pole colder than the North Pole?)<br>南极为什么比北极冷<br>(En: Why is the Antarctic colder than the North Pole) | 1 |
| | Sentence Pair4 | 春天适合种什么花?<br>(En: What kind of flowers are suitable for planting in spring?)<br>春天适合种什么菜?<br>(En: What kind of vegetables are suitable for growing in spring?) | 0 |

### 4.2 Baselines

According to the existing work on the two datasets, we categorize the baselines into three groups, as follows.

- **Traditional unsupervised methods**
  - —**Random:** It randomly judges whether the sentences are semantically matched [4].
  - —**TF-IDF:** It determines the semantic matching degree between sentences according to TF-IDF features [4].
  - —**WMD:** It employs an algorithm based on Wasserstein distance to calculate the matching degree of two sentences [25].
  - —$\mathbf{C}_{wo}$, $\mathbf{C}_{ngram}$, $\mathbf{D}_{edt}$, and $\mathbf{S}_{cos}$**:** All of them are unsupervised matching methods, which make the matching judgement according to word overlap, *N*-gram overlap, edit distance, and cosine similarity, respectively [25].
- **Methods based on basic neural networks**
  - —**Text-CNN and CNN:** It employs CNN to capture sentence features to generate the sentence representations, followed by a softmax classifier to compute the matching degree [4, 25].
  - —**BiLSTM:** It is similar to the former Text-CNN and CNN methods; however, it replaces CNN with the BiLSTM component [4, 25].
- **Methods based on advanced neural networks**
  - —**DIIN:** It extracts semantic features from an interaction space to realize the high-level understanding for sentence pairs, which is a dense interactive inference network [11].

—**DFF:** It combines the original embeddings from different encoders to enhance the feature representation of sentences. Three matching strategies are implemented to generate the final matching representation [58].
—**MSEM:** It utilizes a connected graph to describe the relation between sentences, and realizes a neural architecture of multi-task learning to address both the sentence matching and classification problems [14].
—**GMN:** It implements a neural graph matching network, which is fed with all possible segmentation paths to form word lattice graphs, and learns graph-based representations for sentences [5].

According to the granularity of input sentences, some baselines are further implemented on character and word granularity, respectively. The variants are marked with different subscripts, such as $\text{DFF}_{char}$ and $\text{DFF}_{word}$, $\text{WMD}_{char}$ and $\text{WMD}_{word}$.

### 4.3 Evaluation Metrics

The goal of the SSM task is to predict whether the semantic information of two sentences is matched, whose outputs are expected to be consistent with the annotated tags in datasets. In order to compare with the baselines, precision, recall, $\text{F}_1$-score, and accuracy are selected as the evaluation metrics. According to the consistency between the true tags and predicted tags, the instances can be categorized into true-positive instances (TP), false-positive ones (FP), true-negative ones (TN), and false-negative ones (FN). The four evaluation metrics are defined in Equation (12).

$$\begin{aligned} Precision &= \frac{TP}{TP + FP}, \\ Recall &= \frac{TP}{TP + FN}, \\ F_1\text{-}score &= \frac{2 \times Precision \times Recall}{Precision + Recall}, \\ Accuracy &= \frac{TP + TN}{TP + FN + FP + FN}. \end{aligned} \tag{12}$$

### 4.4 Performance Evaluation

The experimental results on the datasets are summarized in Table 2 and Table 3. According to the tables, we have several observations.

First, the methods based on basic neural networks perform significantly better than the traditional unsupervised methods. For the BQ dataset, all the basic neural models (i.e., Text-CNN and BiLSTM) outperform all traditional models (i.e., Random and TF-IDF) in terms of precision, recall, $\text{F}_1$-score, and accuracy. A similar observation is found on the LCQMC dataset. The basic neural models including $\text{CNN}_{char}$, $\text{CNN}_{word}$, $\text{BiLSTM}_{char}$, and $\text{BiLSTM}_{word}$, outperform the traditional ones including $\text{WMD}_{char}$, $\text{WMD}_{word}$, $\text{C}_{wo}$, $\text{C}_{ngram}$, $\text{D}_{edt}$, and $\text{S}_{cos}$ in terms of precision, $\text{F}_1$-score, and accuracy. This is probably because the traditional unsupervised methods model sentences according to the shallow and surface features of sentences, which fail to utilize the deep and latent semantic information effectively. However, the methods based on basic neural networks employ CNN and LSTM to capture more deep semantic features, which are essential for representing the sentences more accurately.

Second, the methods based on advanced neural networks significantly outperform the methods based on basic neural models. For the BQ dataset, all advanced neural methods (BiMPM, DIIN, $\text{DFF}_{char}$, $\text{DFF}_{word}$, MSEM, and GMN) beat all basic neural methods (Text-CNN and

Table 2. Experimental Results on BQ Dataset

| Methods | Precision | Recall | $F_1$-score | Accuracy |
|---|---|---|---|---|
| Random | 50.43 | 50.56 | 50.49 | 50.43 |
| TF-IDF | 64.68 | 60.94 | 62.75 | 63.83 |
| Text-CNN | 67.77 | 70.64 | 69.17 | 68.52 |
| BiLSTM | 75.04 | 70.46 | 72.68 | 73.51 |
| BiMPM | 82.28 | 81.18 | 81.73 | 81.85 |
| DIIN | 81.58 | 81.14 | 81.36 | 81.41 |
| $DFF_{char}$ | **85.32** | 76.33 | 80.52 | 81.69 |
| $DFF_{word}$ | 84.43 | 77.48 | 80.70 | 81.59 |
| MSEM | 82.88 | **84.36** | 83.62 | 83.47 |
| GMN | 84.82 | 83.42 | 84.11 | 84.24 |
| 3DSSM | 84.99 | 83.67 | **84.25** | **84.46** |

Table 3. Experimental Results on LCQMC Dataset

| Methods | Precision | Recall | $F_1$-score | Accuracy |
|---|---|---|---|---|
| $WMD_{char}$ | 67.0 | 81.2 | 73.4 | 70.6 |
| $WMD_{word}$ | 64.4 | 78.6 | 70.8 | 60.0 |
| $C_{wo}$ | 61.1 | 83.6 | 70.6 | 70.7 |
| $C_{ngram}$ | 52.3 | 89.3 | 66.0 | 61.2 |
| $D_{edt}$ | 46.5 | 86.4 | 60.5 | 52.3 |
| $S_{cos}$ | 60.1 | 88.7 | 71.6 | 70.3 |
| $CNN_{char}$ | 67.1 | 85.6 | 75.2 | 71.8 |
| $CNN_{word}$ | 68.4 | 84.6 | 75.7 | 72.8 |
| $BiLSTM_{char}$ | 67.4 | 91.0 | 77.5 | 73.5 |
| $BiLSTM_{word}$ | 70.6 | 89.3 | 78.92 | 76.1 |
| $BiMPM_{char}$ | 77.6 | 93.9 | 85.0 | 83.4 |
| $BiMPM_{word}$ | 77.7 | 93.5 | 84.9 | 83.3 |
| $DFF_{char}$ | 78.58 | 93.88 | 85.51 | 84.15 |
| $DFF_{word}$ | 77.69 | 94.08 | 85.06 | 83.53 |
| MSEM | 78.90 | 93.73 | 85.68 | 84.33 |
| GMN | 78.92 | **94.49** | 86.0 | 84.62 |
| 3DSSM | **81.88** | 91.66 | **86.45** | **85.70** |

BiLSTM) in all evaluation metrics. A similar tendency is shown on the LCQMC dataset, on which the advanced neural models ($BiMPM_{char}$, $BiMPM_{word}$, $DFF_{char}$, $DFF_{word}$, MSEM, and GMN) also outperform the basic neural models ($CNN_{char}$, $CNN_{word}$, $BiLSTM_{char}$, and $BiLSTM_{word}$) in all evaluation metrics. This is probably because the basic neural methods usually employ some basic neural components (CNN and LSTM) to encode sentences, which only capture some simple local or sequential features and cannot effectively exploit more sophisticated information embedded in sentences.

Third, our approach can consistently outperform all compared baseline methods on two datasets in terms of $F_1$-score and accuracy. The compared methods based on advanced neural networks have strived to implement some complicated architectures to capture the deep features in sentences. For instance, BiMPM utilizes LSTM networks to implement the bilateral multi-perspective matching

Table 4. Comparison with BERT-Based Models

| Task | Metrics | Model | (M = Million) |
|---|---|---|---|
| LCQMC | Accuracy(#FLOPs) | BERT [24] | **86.68(21785M)** |
| | Accuracy(#FLOPs) | DistilBERT (6 layers) | 84.12(10,918M) |
| | Accuracy(#FLOPs) | DistilBERT (3 layers) | 84.07(5,428M) |
| | Accuracy(#FLOPs) | DistilBERT (1 layers) | 71.34(1,858M) |
| | Accuracy(#FLOPs) | FastBERT (speed=0.1) | 86.59(12,930M) |
| | Accuracy(#FLOPs) | FastBERT (speed=0.5) | 84.05(6,352M) |
| | Accuracy(#FLOPs) | FastBERT (speed=0.8) | 77.45(3,310M) |
| | Accuracy(#FLOPs) | 3DSSM | 85.70(21.02M) |
| BQ | Accuracy(#FLOPs) | BERT [44] | **84.8(-)** |
| | Accuracy(#FLOPs) | 3DSSM | 84.46(10.81M) |

mechanism. DIIN designs an interactive inference network and applies 2D CNN kernels to extract interactive features. DFF assembles multiple LSTM components to build a feature fusion model. MSEM constructs a connected graph to describe the relations between sentences and employs multi-task learning to solve the matching problem. GMN implements a neural graph matching network, which learns graph-based representations for sentences. Although these compared state-of-the-art methods are powerful on sentence representation, similar to the limitation of the traditional RNN and CNN networks, they fail to simultaneously learn the local features in different dimensions of words and the sequential information between consecutive words in sentences. However, our 3DSSM method utilizes 3D CNN to encode sentences, which first transforms sentences into semantic feature cubes, then applies 3D convolution kernels to capture the local and sequential features in sentences, and utilizes another 3D CNN module to capture the interactive features between two sentences. Compared with the baselines, 3DSSM is able to capture and represent the matching features of two sentences more accurately. Thus, our method can outperform the others.

### 4.5 Comparison with BERT-Based Models

In order to investigate the effectiveness of our model and BERT-based ones, we adopt two important metrics to compare them, i.e., accuracy and #FLOPs. #FLOPs represents the number of floating-point operations, which is a metric to evaluate the computational complexity of a neural model. The larger #FLOPs means the longer inference time. As shown in Table 4, in terms of accuracy, although 3DSSM is inferior to BERT and FastBert (speed = 0.1) on the LCQMC dataset, it still beats the other models based on BERT, including DistilBERT and FastBert (speed = 0.5, 0.8). On the BQ dataset, 3DSSM achieves comparable performance to BERT. In terms of #FLOPs, 3DSSM demonstrates a significant superiority, whose computational complexity is much lower than that of BERT-based models. That is, 3DSSM is able to achieve comparable performance in terms of accuracy with BERT-based models while requiring much less #FLOPs than them.

### 4.6 Ablation Study

In this subsection, we investigate the importance of each key component of 3DSSM by comparing the performance of 3DSSM with that of its three variants, which are described as follows:

- **3DSSM**$^{-En3D}$**:** Recalling that the standard 3DSSM encodes semantic feature cube with the first 3D CNN sub-module in Equation (5), the variant removes this sub-module.

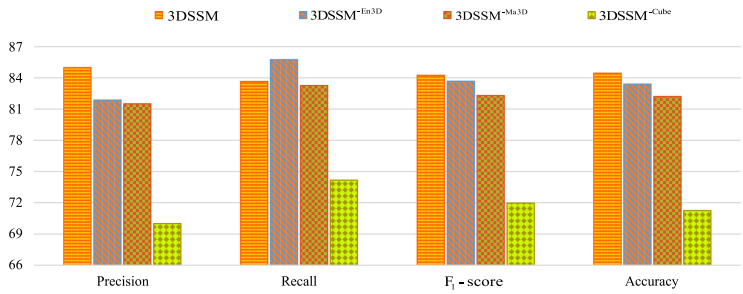



Fig. 7. Ablation experimental results on the BQ dataset.

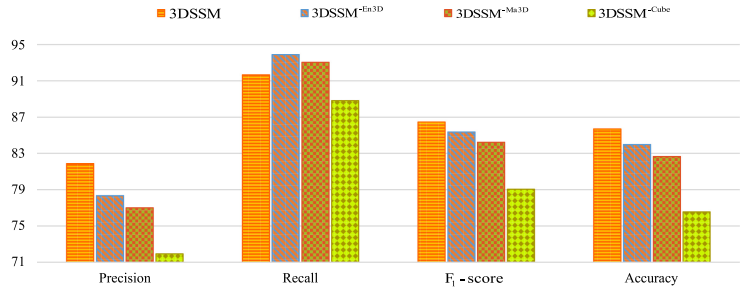



Fig. 8. Ablation experimental results on the LCQMC dataset.

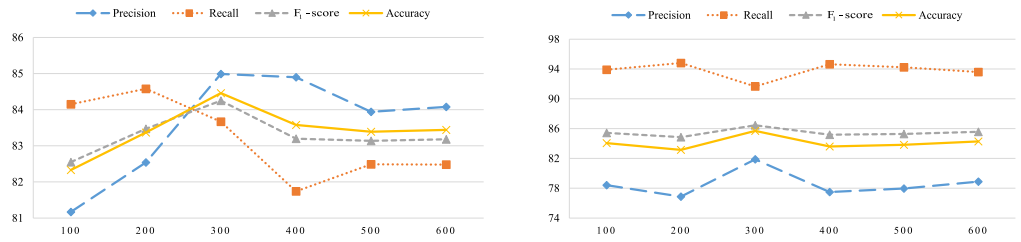
- **3DSSM$^{-Ma3D}$:** Recalling that the standard 3DSSM encodes the initial representation of a sentence pair to capture the interactive features and generate a matching representation with the second 3D CNN sub-module in Equation (7), the variant removes this sub-module.
- **3DSSM$^{-Cube}$:** Recalling that the standard 3DSSM generates semantic feature cubes for sentences with the feature cube network described in Section 3.4, the variant removes this network. Thus, the character and word embeddings are concatenated directly to replace the original semantic feature cube.

The experimental results are shown in Figure 7 and Figure 8. According to the figures, we have several observations, as follows.

First, in terms of precision, $F_1$-score, and accuracy, 3DSSM$^{-En3D}$ obtains worse results than 3DSSM on two datasets. This is probably because 3DSSM$^{-En3D}$ removes the first 3D CNN sub-module for encoding semantic feature cube. This leads to the local features between different dimensions of words and the sequential information in successive words being not fully captured, which inevitably affects the quality of sentence representation and the performance of SSM task.

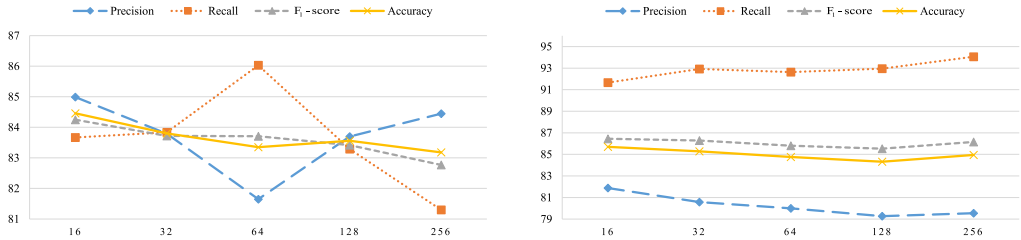
Second, in terms of precision, $F_1$-score, and accuracy, 3DSSM$^{-Ma3D}$ achieves worse results than 3DSSM on two datasets. This is probably because 3DSSM$^{-Ma3D}$ removes the second 3D CNN sub-module for capturing the interactive features between two sentences. The missing of interactive features leads to the decline of SSM performance. In addition, comparing 3DSSM$^{-En3D}$ and 3DSSM$^{-Ma3D}$, we find that the performances of the former are usually superior to those of the latter. This means that the second 3D CNN sub-module described in Equation (7) plays a more important role than the first sub-module described in Equation (5).

Third, in terms of precision, recall, $F_1$-score, and accuracy, 3DSSM$^{-Cube}$ obtains the worst results on two datasets. This is probably because 3DSSM$^{-Cube}$ completely removes the feature cube network, which seriously destroys the entire architecture of the 3DSSM model, greatly weakens




(a) Results on BQ (b) Results on LCQMC

Fig. 9. Comparison on encoding dimension of BiLSTM.




(a) Results on BQ (b) Results on LCQMC

Fig. 10. Comparison on number of convolution kernels of 3D CNN in Equation (5).

its ability for sentence pair modeling, and hurts the performance severely. Feature cube network is the core of 3DSSM method, which provides the foundation for the following 3D CNN module. Without the feature cube network, it is impossible for the variant to beat the other baselines.

In summary, each key component in 3DSSM is helpful for SSM task. All of them are crucial for the outstanding performance of the 3DSSM model.

### 4.7 Hyperparameter Analysis

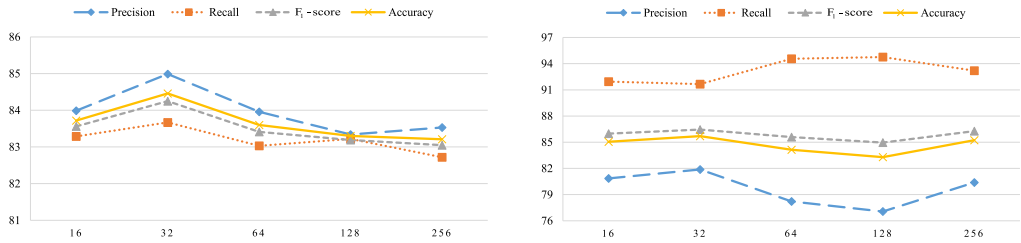
In this subsection, we discuss the influences of different hyperparameters on the performance of the 3DSSM model.

First, for the encoding dimension of BiLSTM in the feature cube network, according to Figure 9(a) and 9(b), in terms of $F_1$-score and accuracy, 3DSSM obtains the best performance on the BQ and LCQMC datasets when the encoding dimension of BiLSTM is set to 300. The larger dimensions mean the better ability to model sentences. However, too many dimensions reduce the generalization ability and result in unsatisfactory performance on the test data. According to the experimental results, 300 is the best parameter for the encoding dimension of BiLSTM.

Second, for the number of convolution kernels in the first 3D CNN sub-module as described in Equation (5), according to Figure 10(a) and 10(b), in terms of $F_1$-score and accuracy, 3DSSM achieves the optimal performance on two datasets when the number of convolution kernels is set to 16. In addition, for the number of convolution kernels in the second 3D CNN sub-module as described in Equation (7), according to Figure 11(a) and 11(b), in terms of $F_1$-score and accuracy, 3DSSM achieves the best performance on two datasets when the number of convolution kernels is set to 32. The more convolution kernels there are, the more powerful the CNN-based model will be. However, the neural network will overfit the training dataset when the number of convolution kernels is too large, which will damage the performance on the test data. According to the

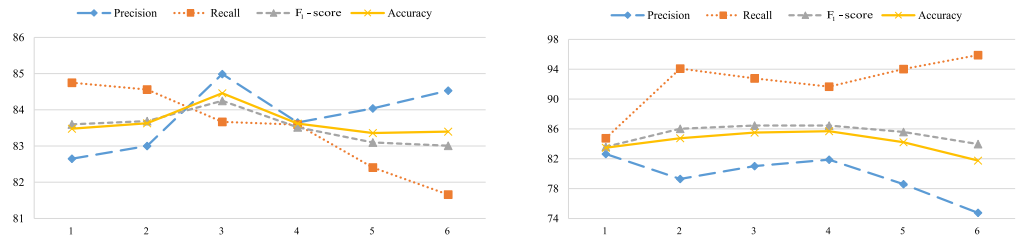
Table 5. Comparison of Time Consumption on Different Depths of Feature Cube Network

| Depths | BQ | LCQMC |
|---|---|---|
| 1 | 87 s | 227 s |
| 2 | 172 s | 434 s |
| 3 | 253 s | 637 s |
| 4 | 338 s | 851 s |
| 5 | 421 s | 1,046 s |
| 6 | 504 s | 1,241 s |




(a) Results on BQ (b) Results on LCQMC

Fig. 11. Comparison on number of convolution kernels of 3D CNN in Equation (7).



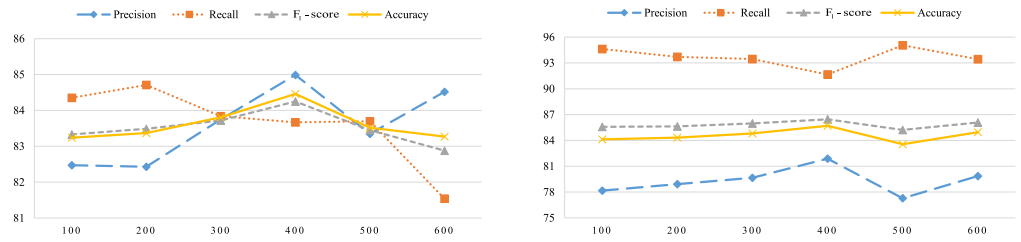

(a) Results on BQ (b) Results on LCQMC

Fig. 12. Comparison on depth of the feature cube network.

experimental results, for the two 3D CNN sub-modules in the 3DSSM model, the numbers of their convolution kernels are set to 16 and 32, respectively.

Third, for the depth of the feature cube network, according to Figure 12(a) and 12(b), in terms of $F_1$-score and accuracy, 3DSSM demonstrates the best performance when the depths are set to 3 and 4 on two datasets, respectively. Considering the differences between the two datasets, it is natural to encode sentences with feature cube networks of different depths. In order to investigate the time consumption of 3DSSM models equipped with different feature cube networks, we report the time consumption on each training epoch, as shown in Table 5. It is obvious that with the increase of the depths of the feature cube network, the time consumption on each epoch grows gradually. When the depths are the same, the time consumption on LCQMC is two times longer than that on BQ. This may be caused by the size of the two datasets, i.e., LCQMC is larger than BQ.

Last, for the dimension of character and word embedding fed into 3DSSM, according to Figure 13(a) and 13(b), in terms of precision, $F_1$-score, and accuracy, 3DSSM obtains the best




(a) Results on BQ

(b) Results on LCQMC

Fig. 13. Comparison on dimension of character and word embedding.

performance on the BQ and LCQMC datasets when the dimension is set to 400. The analysis for this hyperparameter is similar to that for the encoding dimension of BiLSTM.

## 5 CONCLUSIONS

Sentence semantic matching is a critical yet challenging task for human–robot language interaction, which requires an accurate representation for the sophisticated features embedded in and between sentences. Existing work including neural models is troubled by the inability to fully capture the local and sequential features. This article devises a novel 3D CNN-based sentence semantic matching method for human–robot language interaction. Our proposed 3DSSM model receives the character and word embeddings of sentences, and designs a specific feature cube network to transform the embeddings into semantic feature cubes. Then, a 3D CNN module is implemented to encode the semantic feature cube to capture the deep features in sentences, and learn the interactive features between sentences. Exhaustive experiments and empirical analysis validate the outstanding performance of the 3DSSM model. In the future, we will try to further integrate more information into the feature cube network and verify the ability of 3DSSM on the related tasks.

## REFERENCES

[1] JeanBaptiste Alayrac, Joao Carreira, and Andrew Zisserman. 2019. The visual centrifuge: Model-free layered video representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2457–2466.

[2] Cen Chen, Xiaolu Zhang, Sheng Ju, Chilin Fu, Caizhi Tang, Jun Zhou, and Xiaolong Li. 2019. AntProphet: An intention mining system behind Alipay's intelligent customer service bot. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. 6497–6499.

[3] Haolan Chen, Fred X. Han, Di Niu, Dong Liu, Kunfeng Lai, Chenglin Wu, and Yu Xu. 2018. Mix: Multi-channel information crossing for text matching. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 110–119.

[4] Jing Chen, Qingcai Chen, Xin Liu, Haijun Yang, Daohe Lu, and Buzhou Tang. 2018. The BQ corpus: A large-scale domain-specific Chinese corpus for sentence semantic equivalence identification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 4946–4951.

[5] Lu Chen, Yanbin Zhao, Boer Lyu, Lesheng Jin, Zhi Chen, Su Zhu, and Kai Yu. 2020. Neural graph matching networks for Chinese short text matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 6152–6158.

[6] Kyunghyun Cho, van Bart Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. 1724–1734.

[7] Jian Fu, Xipeng Qiu, and Xuanjing Huang. 2016. Convolutional deep neural networks for document-based question answering. In *Proceedings of the Natural Language Understanding and Intelligent Applications*, Lecture Notes in Computer Science, Vol. 10102. 790–797.

[8] Guangwei Gao, Yi Yu, Jin Xie, Jian Yang, Meng Yang, and Jian Zhang. 2020. Constructing multilayer locality-constrained matrix regression framework for noise robust face super-resolution. *Pattern Recognit.* 110 (2020), 107539.

[9] Guangwei Gao, Yi Yu, Jian Yang, Guo-Jun Qi, and Meng Yang. 2020. Hierarchical deep CNN feature set-based representation learning for robust cross-resolution face recognition. *IEEE Trans. Circ. Syst. Video Technol.* (2020).
[10] Guangwei Gao, Yi Yu, Meng Yang, Heyou Chang, Pu Huang, and Dong Yue. 2020. Cross-resolution face recognition with pose variations via multilayer locality-constrained structural orthogonal procrustes regression. *Inf. Sci.* 506 (2020), 19–36.
[11] Yichen Gong, Heng Luo, and Jian Zhang. 2018. Natural language inference over interaction space. In *Proceedings of the 6th International Conference on Learning Representations*. 1–15.
[12] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.* 9, 8 (1997), 1735–1780.
[13] Xinyu Hua and Lu Wang. 2019. Sentence-level content planning and style specification for neural text generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*. 591–602.
[14] Qiang Huang, Jianhui Bu, Weijian Xie, Shengwen Yang, Weijia Wu, and Liping Liu. 2019. Multi-task sentence encoding model for semantic retrieval in question answering systems. In *Proceedings of the International Joint Conference on Neural Networks*. 1–8.
[15] Shengqin Jiang, Yuankai Qi, Haokui Zhang, Zongwen Bai, Xiaobo Lu, and Peng Wang. 2020. D3D: Dual 3D convolutional network for real-time action recognition. *IEEE Trans. Ind. Inf.* 17, 7 (2020), 4584–4593.
[16] Rushi Lan, Long Sun, Zhenbing Liu, Huimin Lu, Cheng Pang, and Xiaonan Luo. 2020. MADNet: A fast and lightweight network for single-image super resolution. *IEEE Trans. Cybern.* 51, 3 (2020), 1443–1453.
[17] Danny Lange. 2019. Cognitive robotics: Making robots sense, understand, and interact. *Computer* 52, 12 (2019), 39–44.
[18] Seong-Gyun Leem, In-Chul Yoo, and Dongsuk Yook. 2019. Multitask learning of deep neural network-based keyword spotting for IoT devices. *IEEE Trans. Consum. Electron.* 65, 2 (2019), 188–194.
[19] Peixia Li, Dong Wang, Lijun Wang, and Huchuan Lu. 2018. Deep visual tracking: Review and experimental comparison. *Pattern Recogn.* 76 (2018), 323–338.
[20] Yulong Li, Dong Zhou, and Wenyu Zhao. 2020. Combining local and global features into a siamese network for sentence similarity. *IEEE Access* 8 (2020), 75437–75447.
[21] Zhouhan Lin, Minwei Feng, Cícero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. In *Proceedings of the 5th International Conference on Learning Representations*. 1–15.
[22] Fagui Liu, Jingzhong Zheng, Lailei Zheng, and Cheng Chen. 2020. Combining attention-based bidirectional gated recurrent neural network and two-dimensional convolutional neural network for document-level sentiment classification. *Neurocomputing* 371 (2020), 39–50.
[23] Mingtong Liu, Yujie Zhang, Jinan Xu, and Yufeng Chen. 2019. Original semantics-oriented attention and deep fusion network for sentence matching. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*. 2652–2661.
[24] Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Haotang Deng, and Qi Ju. 2020. FastBERT: A self-distilling BERT with adaptive inference time. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 6035–6044.
[25] Xin Liu, Qingcai Chen, Chong Deng, Huajun Zeng, Jing Chen, Dongfang Li, and Buzhou Tang. 2018. LCQMC: A large-scale Chinese question matching corpus. In *Proceedings of the 27th International Conference on Computational Linguistics*. 1952–1962.
[26] Xiaomin Liu, Jun-Bao Li, Jeng-Shyang Pan, Shuo Wang, Xudong Lv, and Shuanglong Cui. 2020. Image-matching framework based on region partitioning for target image location. *Telecommun. Syst.* 74, 3 (2020), 269–286.
[27] Huimin Lu, Yujie Li, Min Chen, Hyoungseop Kim, and Seiichi Serikawa. 2018. Brain intelligence: Go beyond artificial intelligence. *Mob. Netw. Appl.* 23, 2 (2018), 368–375.
[28] Huimin Lu, Yujie Li, Shenglin Mu, Dong Wang, Hyoungseop Kim, and Seiichi Serikawa. 2017. Motor anomaly detection for unmanned aerial vehicles using reinforcement learning. *IEEE Internet Things.* 5, 4 (2017), 2315–2322.
[29] Huimin Lu, Ming Zhang, Xing Xu, Yujie Li, and Heng Tao Shen. 2021. Deep fuzzy hashing network for efficient image retrieval. *IEEE Trans. Fuzzy Syst.* 29, 1 (2021), 166–176.
[30] Wenpeng Lu, Xu Zhang, Huimin Lu, and Fangfang Li. 2020. Deep hierarchical encoding model for sentence semantic matching. *J. Vis. Commun. Image Represent.* 71 (2020), 102794.
[31] Wenpeng Lu, Yuteng Zhang, Shoujin Wang, Heyan Huang, Qian Liu, and Sheng Luo. 2021. Concept representation by learning explicit and implicit concept couplings. *IEEE Intell. Syst.* 36, 1 (2021), 6–15.
[32] Arindam Mitra, Ishan Shrivastava, and Chitta Baral. 2020. Enhancing natural language inference using new and expanded training data sets and new learning models. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*. 8504–8511.
[33] Vinod Nair and Geoffrey E. Hinton. 2010. Rectified linear units improve restricted Boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning*. 807–814.

[34] Pin Ni, Yuming Li, Gangmin Li, and Victor Chang. 2020. Natural language understanding approaches based on joint task of intent detection and slot filling for IoT voice interaction. *Neural. Comput. Appl.* 32 (2020), 16149–16166. https://link.springer.com/article/10.1007/s00521-020-04805-x.

[35] Guocheng Niu, Hengru Xu, Bolei He, Xinyan Xiao, Hua Wu, and Sheng Gao. 2019. Enhancing local feature extraction with global representation for neural text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*. 496–506.

[36] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text matching as image recognition. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*. 2793–2799.

[37] Chris Paxton, Yonatan Bisk, Jesse Thomason, Arunkumar Byravan, and Dieter Foxl. 2019. Prospection: Interpretable plans from language by predicting the future. In *Proceedings of the 2019 International Conference on Robotics and Automation*. 6942–6948.

[38] Juncai Peng, Yuanjie Shao, Nong Sang, and Changxin Gao. 2020. Joint image deblurring and matching with feature-based sparse representation prior. *Pattern Recognit.* 103 (2020), 107300.

[39] Eugenio Rubio-Drosdov, Daniel Díaz-Sánchez, Florina Almenárez, Patricia Arias-Cabarcos, and Andrés Marín. 2017. Seamless human-device interaction in the internet of things. *IEEE Trans. Consum. Electron.* 63, 4 (2017), 490–498.

[40] Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Trans. Signal Process.* 45, 11 (1997), 2673–2681.

[41] Hamidreza Shahidi, Ming Li, and Jimmy Lin. 2020. Two birds, one stone: A simple, unified model for text generation from structured and unstructured data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 3864–3870.

[42] Yang Song, Qinmin Vivian Hu, and Liang He. 2019. P-CNN: Enhancing text matching with positional convolutional neural network. *Knowl. Based Syst.* 169 (2019), 67–79.

[43] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* 15, 1 (2014), 1929–1958.

[44] Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2020. ERNIE 2.0: A continual pre-training framework for language understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 8968–8975.

[45] Jesse Thomason, Aishwarya Padmakumar, Jivko Sinapov, Nick Walker, Yuqian Jiang, Harel Yedidsion, Justin Hart, Peter Stone, and Raymond Mooney. 2020. Jointly improving parsing and perception for natural language commands through human-robot dialog. *J. Mach. Learn. Res.* 67 (2020), 327–374.

[46] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2019. Generating token-level explanations for natural language inference. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 963–969.

[47] Heyuan Wang, Fangzhao Wu, Zheng Liu, and Xing Xie. 2020. Fine-grained interest matching for neural news recommendation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 836–845.

[48] Shoujin Wang, Liang Hu, Yan Wang, Quan Z. Sheng, Mehmet Orgun, and Longbing Cao. 2020. Intention nets: Psychology-inspired user choice behavior modeling for next-basket prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 6259–6266.

[49] Shoujin Wang, Liang Hu, Yan Wang, Quan Z. Sheng, Mehmet Orgun, and Longbing Cao. 2020. Intention2Basket: A neural intention-driven approach for dynamic next-basket planning. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence*. 2333–2339.

[50] Zhongbin Xie and Shuai Ma. 2019. Dual-view variational auto-encoders for semi-supervised text matching. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. 5306–5312.

[51] Xing Xu, Kaiyi Lin, Lianli Gao, Huimin Lu, Heng Tao Shen, and Xuelong Li. 2020. Learning cross-modal common representations by private-shared subspaces separation. *IEEE Trans. Cybern.* (2020), 1–15.

[52] Xing Xu, Huimin Lu, Jingkuan Song, Yang Yang, Hengtao Shen, and Xuelong Li. 2019. Ternary adversarial networks with self-supervision for zero-shot cross-modal retrieval. *IEEE Trans. Cybern.* 77, 17 (2019), 21847–21860.

[53] Xing Xu, Tan Wang, Yang Yang, Lin Zuo, Fumin Shen, and Heng Tao Shen. 2020. Cross-modal attention with semantic consistence for image-text matching. *IEEE Trans. Neural Networks Learn. Syst.* 31, 12 (2020), 5412–5425.

[54] Bin Yan, Haojie Zhao, Dong Wang, Huchuan Lu, and Xiaoyun Yang. 2019. 'Skimming-Perusal' Tracking: A framework for real-time and robust long-term tracking. In *Proceedings of the IEEE International Conference on Computer Vision*. 2385–2393.

[55] Yi Yu, Suhua Tang, Kiyoharu Aizawa, and Akiko Aizawa. 2018. Category-based deep CCA for fine-grained venue discovery from multimodal data. *IEEE Trans. Neural Netw. Learn.* 30, 4 (2018), 1250–1258.

[56] Chenggong Zhang, Weijuan Zhang, Daren Zha, Pengjie Ren, and Nan Mu. 2019. A multi-granularity neural network for answer sentence selection. In *Proceedings of the 2019 International Joint Conference on Neural Networks*. 1–7.

[57] Kun Zhang, Guangyi Lv, Linyuan Wang, Le Wu, Enhong Chen, Fangzhao Wu, and Xing Xie. 2019. DRr-Net: Dynamic re-read network for sentence semantic matching. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 7442–7449.

[58] Xu Zhang, Wenpeng Lu, Fangfang Li, Xueping Peng, and Ruoyu Zhang. 2019. Deep feature fusion model for sentence semantic matching. *CMC-Comput. Mater. Contin.* 61, 2 (2019), 601–616.

[59] Yuteng Zhang, Wenpeng Lu, Weihua Ou, Guoqiang Zhang, Xu Zhang, Jinyong Cheng, and Weiyu Zhang. 2020. Chinese medical question answer selection via hybrid models based on CNN and GRU. *Multim. Tools Appl.* 79, 21–22 (2020), 14751–14776.

Received September 2020; revised January 2020; accepted February 2021