CAN: Enhancing Sentence Similarity Modeling with Collaborative and Adversarial Network

Qin Chen[1], Qinmin Hu, Jimmy Xiangji Huang[3] and Liang He[1,4]
[1]Department of Computer Science & Technology, East China Normal University, Shanghai, China
[2]Department of Computer Science, Ryerson University, Toronto, Canada
[3]Information Retrieval & Knowledge Management Research Lab, York University, Toronto, Canada
[4]Shanghai Engineering Research Center of Intelligent Service Robot, Shanghai, China
qchen@ica.stc.sh.cn,{qmhu,lhe}@cs.ecnu.edu.cn,jhuang@yorku.ca

ABSTRACT

The neural networks have attracted great attention for sentence similarity modeling in recent years. Most neural networks focus on the representation of each sentence, while the common features of a sentence pair are not well studied. In this paper, we propose a Collaborative and Adversarial Network (CAN), which explicitly models the common features between two sentences for enhancing sentence similarity modeling. To be specific, a common feature extractor is presented and embedded into our CAN model, which includes a generator and a discriminator playing a collaborative and adversarial game for common feature extraction. Experiments on three benchmark datasets, namely TREC-QA and WikiQA for answer selection and MSRP for paraphrase identification, show that our proposed model is effective to boost the performance of sentence similarity modeling. In particular, our proposed model outperforms the state-of-the-art approaches on TREC-QA without using any external resources or pre-training. For the other two datasets, our model is also comparable to if not better than the recent neural network approaches.

KEYWORDS

Answer Selection; Paraphrase Identification; Neural Networks; Collaborative and Adversarial Learning

ACM Reference Format:
Qin Chen, Qinmin Hu, Jimmy Xiangji Huang and Liang He. 2018. CAN: Enhancing Sentence Similarity Modeling with Collaborative and Adversarial Network. In *SIGIR'18: The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval, July 8-12, 2018, Ann Arbor, MI, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3209978.3210019

1 INTRODUCTION

Sentence similarity modeling is a fundamental problem in the information retrieval (IR) community. For example, in the answer selection task, all the candidate answers are ranked by the sentence similarity with the given question [40, 44]. In addition, sentence similarity has been widely used to determine whether two sentences are paraphrases or not for paraphrase identification [18, 46].

Most traditional methods measure the sentence similarity based on the feature engineering and linguistic tools, which are labour consuming and prone to the errors of external tools such as dependency parsing [36, 45]. Recently, the neural network based approaches have achieved good performance without any human interventions. One of the well-known architectures is the siamese recurrent neural network (RNN) [28], which used the long short-term memory (LSTM) model [19] to obtain a sequential hidden states for each sentence. Then, the hidden states were aggregated for sentence representations, and the similarity score was calculated with a predefined similarity metric. To capture the salient information for better sentence representations, the attention based neural networks have been extensively studied [10]. Tan et al. [34] proposed an attention-based LSTM model for question answering, where the representation of the question sentence was utilized for the attentive weight generation for the answer. Yin et al. [47] presented three attention schemes, which integrated the sentence mutual influence into the convolutional neural network (CNN) for sentence similarity modeling.

To the best of our knowledge, most neural networks focus on better representation of each sentence, while the common or similar features in a sentence pair are not well studied [39, 49]. With the intuition that the similar and dissimilar features were effective to boost sentence similarity modeling, Wang et al. [42] proposed an approach to decompose each word vector into a similar and dissimilar component. Then, the two components were composed as a feature vector with a two-channel CNN model for sentence similarity assessment. Whereas, the word-by-word semantic matching vector was needed for the lexical decomposition, which was time-consuming. Moreover, their approach relied on various decomposition functions to obtain the similar and dissimilar components.

In this paper, we propose a Collaborative and Adversarial Network (CAN), which integrates a common feature extractor into the RNN framework for enhancing sentence similarity modeling. To ensure the extractor captures the common features between two sentences, we incorporate collaborative learning into the framework of generative adversarial nets (GAN), where a generator and a discriminator play a collaborative and adversarial game for common feature extraction. More concretely, the generator attempts to generate features based on one of the two sentences, and the discriminator tries to distinguish which sentence the features are

SIGIR '18, July 8–12, 2018, Ann Arbor, MI, USA

ACM ISBN 978-1-4503-5657-2/18/07…$15.00
https://doi.org/10.1145/3209978.3210019

generated from (i.e., *generator source*). Different from the previous work which only utilized the adversarial strategy to learn the invariant features in multiple domains and criteria [6, 14, 15, 35], we leverage collaborative learning in addition to the traditional adversarial learning for common feature extraction. To be specific, given a similar sentence pair, it intuitively contains more common features than the dissimilar pair. Thus, the generator aims to extract more common features, making it hard for the discriminator to tell the generator source, namely adversarial learning. For a dissimilar sentence pair, the generator is expected to extract less common features to make them easier to be distinguished by the discriminator, which is our presented collaborative learning strategy. Experiments on three datasets, namely TREC-QA [40] and WikiQA [44] for answer selection and MSRP [8] for paraphrase identification, show that our proposed CAN model is effective to boost the performance of sentence similarity modeling with the extracted common features. Particularly, we outperform the state-of-the-art approaches on TREC-QA. Regarding to the other two datasets, our proposed model is also comparable to if not better than the recent neural network approaches.

The main contributions of our work are as follows: (1) we propose a collaborative and adversarial network, where a common feature extractor is presented and nicely embedded into the RNN framework for common feature extraction; (2) to the best of our knowledge, it is the first attempt to explicitly model the common features by incorporating collaborative learning into the GAN framework; (3) we conduct elaborate analyses of the experimental results on two sentence similarity tasks, which provides a better understanding of the effectiveness of our proposed model.

The rest of this paper is organized as follows. Section 2 presents an overview of the related work. Section 3 gives a brief introduction of using the traditional RNN models for sentence similarity modeling. Section 4 describes the details of our proposed model. The experimental setup details are demonstrated in Section 5, followed by the experimental results and analyses in Section 6. Finally, we conclude our work and present some ideas for future research in Section 7.

2 RELATED WORK

There are a large number of studies on the topic of sentence similarity modeling. Here we mainly review the work which is most related to our research. In addition, since our CAN model incorporates collaborative learning into the framework of generative adversarial nets (GAN) for common feature extraction, we also give a brief introduction of the relevant work about GAN.

2.1 Sentence Similarity Modeling

Sentence similarity modeling plays an important role in many information retrieval tasks, such as question answering [2, 45] and paraphrase identification [22, 36]. Most previous work relied on the feature engineering to model sentence similarities. Yih et al. [45] utilized the external WordNet based semantic features to enhance lexical features for question answering. Wan et al. [36] investigated the dependency-based features, which were particularly useful for classifying cases of false paraphrase in the Microsoft Research Paraphrase Corpus [8]. In [1, 25], the features based on machine translation evaluation metrics and latent semantic analysis were explored respectively for sentence similarity calculation. However, it is usually difficult to decide which features to be used for a specific dataset. Moreover, these methods are labor consuming due to the excessive dependence on the handcraft features.

Recently, the neural network based approaches have been widely used for its effectiveness without any human interventions. Zhao et al. [52] and Fang et al. [13] applied the long short-term memory (LSTM) [19] model for sentence representations of the question and answer. Then, the sentence similarity was calculated for the community-based question answering. To capture the salient information for better sentence representations, the attention mechanism was introduced into the neural networks [5, 10, 37]. Zhang et al. [50] proposed an attentive interactive neural network, which focused on the interactions between text segments for answer selection. In [34], the attentive weights for an answer sentence relied on the interactions with the question sentence. In addition, the word-by-word interactions were utilized for the attentive sentence representations as demonstrated in [10]. To model the internal interactions between two sentences, Wang et al. [37] proposed an IARNN-GATE model, where the question information was added to the active gates to influence the hidden state generation for answers. Moreover, chen et al. [4] proposed a context-aligned RNN model, which absorbed the aligned words' context in the other sentence for the current hidden state update. However, most of the neural network approaches focused on the representation of each sentence, while the similar or common features between a sentence pair were not well studied [39, 49].

The most relevant work to ours is the approach proposed by Wang et al [42]. Specifically, they explicitly decomposed each word vector into a similar and dissimilar component based on the word-by-word semantic matching vector. Then, the two components were composed with a two-channel CNN model for sentence similarity calculation. Although this approach was shown to be effective for sentence similarity modeling, the decomposition was based on the similarity of the vectors corresponding to each word pair, which was time-consuming. Moreover, various functions were needed for the lexical decomposition. In contrast, our proposed CAN model directly extracts the common features by incorporating collaborative learning into the framework of generative adversarial nets, which does not need the lexical decomposition and composition.

2.2 Generative Adversarial Nets

The generative adversarial networks (GAN) [16] have attracted much attention for its powerful framework in learning generative models of arbitrarily complex data distributions. In general, the GAN framework contains a generator that attempts to generate samples which are similar to the existing data, and a discriminator that aims at distinguishing between the real and generated samples. In other words, the generator and discriminator play an adversarial game to make the generated samples as close to the real data as possible. Many GAN variants have been studied for image generation [9, 11]. Denton et al. [7] proposed an LAPGAN model by integrating a conditional form of GAN into a Laplacian pyramid framework, which produced higher quality images than the GAN baseline model. In [31], a deep convolutional generative adversarial

network (DCGAN) was proposed, and it was effective to learn a hierarchy of representations from object parts to scenes in both the generator and discriminator.

In addition, the GAN framework has been explored to learn the shared features among different domains or criteria. Ganin et al. [14] and Tzeng et al. [35] proposed a domain adversarial adaptation method, which incorporated the adversarial learning into the deep networks to learn the transferable features between the source and target domains. Li et al. [24] introduced an end-to-end Adversarial Memory Network (AMN) for cross-domain sentiment classification, where two parameter-shared memory networks were embedded for sentiment classification and domain classification respectively. To resolve the complex multi-mode structures underlying the data distributions, Durugkar [12] and Pei et al. [29] presented a multi-adversarial domain adaptation approach, which applied multiple domain discriminators to discriminate different data distributions. Moreover, the GAN framework was also utilized for multi-criteria learning in Chinese word segmentation (CWS) [6]. Particularly, the underlying shared knowledge across multiple heterogeneous criteria was learned and proved to be more effective for CWS than the single-criterion learning based approach.

To the best of our knowledge, there are few studies about using the GAN framework for sentence similarity modeling. Wang et al. [39] presented an IRGAN framework, which unified the generative and discriminative models in information retrieval. Whereas, it is essentially different from ours. First, its generative retrieval process is stochastic sampling over the discrete data, i.e., the candidate documents. Instead, our model aims at generating the continuous data, i.e., the common features between two sentences. Second, their framework still focuses on the representation of each document or sentence, while our model explicitly models the common features between two sentences for enhancing sentence similarity modeling. Furthermore, our experimental results confirm that incorporating both collaborative learning and adversarial learning into the GAN framework can be more effective for common feature extraction than using the adversarial learning only.

3 TRADITIONAL RNN

Both the convolutional neural network (CNN) and the recurrent neural network (RNN) have been widely used for sentence similarity modeling in previous studies [5, 37, 44, 47]. Since our proposed model is built within the RNN framework, we first give a brief introduction about how to use the traditional RNN for sentence similarity modeling in the following.

In particular, we focus on the long short-term memory (LSTM) based RNN model, which can effectively mitigate the gradient vanish problem [34]. More concretely, given a sentence pair as $X = x_1, x_2, ..., x_m$ and $Y = y_1, y_2, ..., y_n$, the goal is to calculate the similarity between the two sentences. First, each word in X and Y is mapped to a distributed embedding vector, which is denoted by $\mathbf{x}_i \in \mathbb{R}^d$ and $\mathbf{y}_j \in \mathbb{R}^d$ respectively. Then, the LSTM model is utilized to obtain the hidden state corresponding to each word in the sentence. Take the sentence Y as an example, a sequential hidden states as $\mathbf{h}_1^y, \mathbf{h}_2^y, ..., \mathbf{h}_n^y$ are obtained with LSTM, and each $\mathbf{h}_j^y \in \mathbb{R}^k$ can be formulated as:

$$\mathbf{h}_j^y = f(\mathbf{y}_j, \mathbf{h}_{j-1}^y) \tag{1}$$

where f is defined based on the gate mechanism and memory cell in LSTM to maintain the long dependency information, and $\mathbf{h}_j^y$ contains the context information from the first word to the current one [23]. After that, each sentence is represented by the pooling [38] or attention [34] method over the hidden states. Finally, the similarity score is calculated according to the two sentence representations ($\mathbf{H}_X$ and $\mathbf{H}_Y$) by the Manhattan distance similarity function with l_1 norm, which performs slightly better than the other alternatives such as cosine similarity as indicated in [28]:

$$S(X, Y) = exp(-||\mathbf{H}_X - \mathbf{H}_Y||_1) \tag{2}$$

4 OUR PROPOSED MODEL

In this section, we introduce the details of our proposed CAN model. In particular, we first present the framework of our model in Section 4.1, which integrates a common feature extractor into the traditional RNN model for enhancing sentence similarity modeling. Then, we describe how our common feature extractor extracts the common features with a collaborative and adversarial game in Section 4.2. After that, the collaborative and adversarial loss used in the common feature extractor is presented in Section 4.3. Finally, the joint learning approach that combines various losses for parameter learning is introduced in Section 4.4.

4.1 The Framework of CAN

It is notable that the traditional neural networks mainly rely on the representation of each sentence to calculate the similarity, while the common features between the two sentences are neglected. Motivated by the effectiveness of using similar and dissimilar components for sentence similarity learning [42], we propose a collaborative and adversarial network, which embeds a common feature extractor to explicitly extract the common features with both collaborative and adversarial learning for sentence similarity calculation.

The framework of our proposed CAN model is presented in Figure 1. Compared with the traditional neural networks that focus on better representation of each sentence, our model additionally extracts the common features between two sentences with an integrated common feature extractor, which will be introduced in Section 4.2. To investigate whether our extracted common feature can help boost the performance of sentence similarity modeling, we concatenate it with the similarity score calculated by Formula (2). Then, the concatenated feature is fed into a softmax layer to obtain the similarity distributions:

$$p(c|X, Y) = softmax(\mathbf{W}_s(S(X, Y) \oplus \mathbf{F}_G) + \mathbf{b}_s) \tag{3}$$

where $c \in \{0, ..., C-1\}$ is a similarity label with C classes ($C = 2$ in this paper), $\mathbf{F}_G \in \mathbb{R}^k$ is the generated common feature, $\oplus$ denotes the concatenation operation, $\mathbf{W}_s \in \mathbb{R}^{C\times(k+1)}$ and $\mathbf{b}_s \in \mathbb{R}^C$ are the parameters in the softmax layer.

The loss of the sentence similarity task is usually defined by the cross-entropy of the predicted and true similarity label distributions:

$$\mathcal{J}_s(\Theta^s) = -\sum_{(X,Y,c)\in\mathcal{D}} \sum_{j=0}^{C-1} [c = j] \log p(c = j|X, Y) \tag{4}$$

where Θ^s represents all the parameters in the sentence similarity task, including the parameters in LSTM and the final softmax layer,

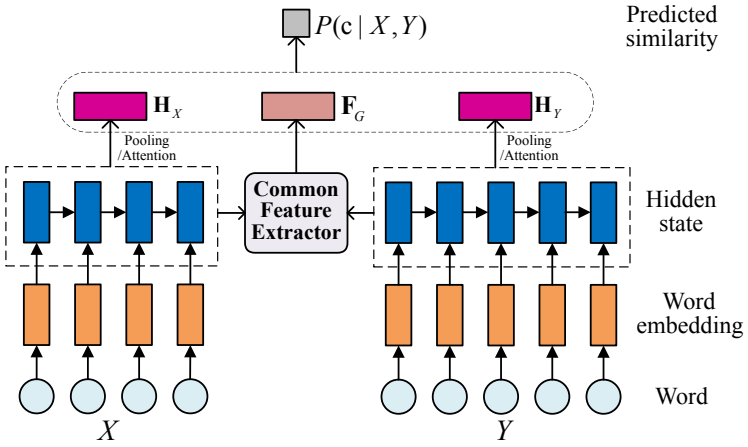


Figure 1: Framework of CAN for sentence similarity modeling.

(X, Y, c) is a sample in the dataset $\mathcal{D}$, and $[c = j]$ is an indicator function of the gold similarity label, which equals to 1 with ground truth and otherwise is 0.

4.2 Common Feature Extractor

Our common feature extractor focuses on extracting the common features between two sentences, which can be utilized for enhancing sentence similarity modeling. In particular, a generator G and a discriminator D are incorporated for common feature extraction. Figure 2 illustrates the details of our common feature extractor, which mainly includes the following steps.

Elite hidden state collection. Intuitively, if a word appears in both sentences, this word is probably more important than other words for similarity measurement, and the corresponding hidden states in the two sentences will contain more common information. Therefore, we first collect the hidden states corresponding to the overlapped words in the two sentences, which will be further utilized for common feature extraction. For ease of description, we call those hidden states as the **elite hidden states**.

Individual feature representation. Since a hidden state in RNN contains the contextual information from the first word to the current one [23], even the hidden states corresponding to the same word in two sentences may indicate different meaning due to the different surrounding contexts of the word. To obtain the individual feature representation of the elite hidden states as stated above, the max pooling method is applied, which has been widely used to obtain the sentence representation for its simplicity and effectiveness [38]. The individual features for sentence X and Y are denoted by $\mathbf{F}_X \in \mathbb{R}^k$ and $\mathbf{F}_Y \in \mathbb{R}^k$ as shown in Figure 2.

Common feature generation. To obtain the common features between two sentences, we introduce a generator G into the extractor. Specifically, the generator generates a new representation based on the individual feature of one sentence, and attempts to retain the information that is common with the other one by playing a collaborative and adversarial game with the discriminator. More formally, given an original individual feature as $\mathbf{F}_Y \in \mathbb{R}^k$, the new generated feature is formulated as:

$$\mathbf{F}_G = G(\mathbf{F}_Y) = \tanh(\mathbf{W}_g \mathbf{F}_Y + \mathbf{b}_g) \tag{5}$$

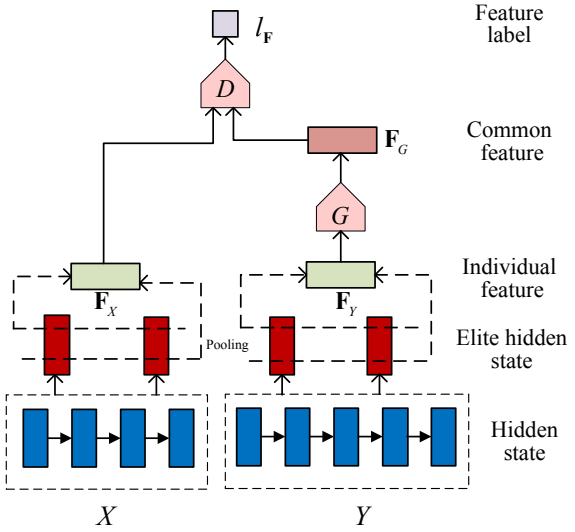


Figure 2: Common feature extractor.

where $\mathbf{W}_g \in \mathbb{R}^{k \times k}$ and $\mathbf{b}_g \in \mathbb{R}^k$ are the parameters in the generator G, and tanh is the hyperbolic tangent function.

Feature discrimination. Ideally, if the new generated feature $\mathbf{F}_G$ can not be distinguished from the individual feature of the other sentence (i.e., $\mathbf{F}_X$), the feature generated based on sentence Y will also contain the information of sentence X. In other words, the generated feature is common between the two sentences. To judge whether the generator generates the common feature, a discriminator D is used for the feature discrimination. Specifically, given a feature representation as $\mathbf{F}$ ($\mathbf{F}_X$ or $\mathbf{F}_G$), the discriminator outputs the probability that $\mathbf{F}$ comes from sentence X or Y. For ease of description, a feature label $l_{\mathbf{F}}$ is introduced and defined as follows:

$$l_{\mathbf{F}} = \begin{cases} 1, & \text{if } \mathbf{F} \text{ comes from sentence } X \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

The probability of the feature label is calculated by the softmax function in the discriminator:

$$D(l_{\mathbf{F}}|\mathbf{F}) = p_d(l_{\mathbf{F}}|\mathbf{F}) = softmax(\mathbf{W}_d \mathbf{F} + \mathbf{b}_d) \tag{7}$$

where $\mathbf{W}_d \in \mathbb{R}^{2\times k}$ and $\mathbf{b}_d \in \mathbb{R}^2$ are the parameters in the discriminator.

4.3 Collaborative and Adversarial Loss

The discriminator aims at discriminating the feature label, namely the feature comes from sentence X or Y. More formally, it minimizes the cross-entropy of the predicted feature label distribution $p_d(l_{\mathbf{F}}|\mathbf{F})$ and the true label:

$$\min_{\Theta^d} \mathcal{J}_d(\Theta^d) = -\sum_{(X,Y,c)\in\mathcal{D}} \log p_d(l_{\mathbf{F}_X}=1|\mathbf{F}_X) + \log p_d(l_{\mathbf{F}_G}=0|\mathbf{F}_G) \quad (8)$$

where Θ^d denotes all the parameters in the discriminator D.

Regarding to our generator, it is quite different from the traditional generator in multi-domain or multi-criteria tasks [6, 29]. Specifically, the traditional generator merely plays against the discriminator, and tries to generate the invariant representations that the discriminator can not distinguish. In this paper, we have an assumption that the similar sentence pairs share more in common, and the generated feature based on one sentence should be closer to the individual feature of the other one. For the dissimilar sentence pairs, it is the opposite. Therefore, we devise a new generator, which plays a collaborative and adversarial game with the discriminator according to the sentence similarity label. In particular, if two sentences are **similar** (i.e., $c = 1$), the generator aims to make the generated feature and the individual feature of the other sentence as close as possible. In other words, it plays an **adversarial** game with the discriminator, making it difficult to discriminate the feature label. While for the **dissimilar** sentence pair (i.e., $c = 0$), the generator is expected to generate the feature which is far from the other sentence since the two sentences share less in common. In another word, it plays a **collaborative** game with the discriminator to make the feature discrimination easier. Thus, our generator tries to maximize the feature label prediction of the generated feature according to the sentence similarity as follows:

$$\max_{\Theta^g} \mathcal{J}_g(\Theta^g) = \sum_{(X,Y,c)\in\mathcal{D}} \sum_{j=0}^{C-1} [c=j] \log p_d(l_{\mathbf{F}_G}=j|\mathbf{F}_G) \quad (9)$$

where Θ^g denotes all the parameters in the generator G, $[c = j]$ is an indicator function of the gold similarity label as described in Formula (4).

4.4 Joint Learning

With the loss of the sentence similarity task defined by Formula (4), and the losses of the discriminator and generator presented in Section 4.3, we combine them together as the final objective function for joint learning:

$$\mathcal{J}(\Theta) = \mathcal{J}_s(\Theta^s) + \mathcal{J}_d(\Theta^d) - \lambda\mathcal{J}_g(\Theta^g) \quad (10)$$

where λ is a trade-off parameter that controls the loss of the generator as demonstrated in [6].

The AdaDelta [48] algorithm with minibatches is used to obtain the optimal parameters $(\Theta^s, \Theta^d, \Theta^g)$ that minimize the objective function $\mathcal{J}(\Theta)$. A brief overview of the joint learning procedure is shown in Algorithm 1, where $\mathcal{B}$ denotes all the b samples in a batch, *n_epoch* and *n_batch* are the number of iterations and batches.

Algorithm 1 Joint learning for the CAN model

```
for i = 1; i <= n_epoch; i++ do
    # Train sentence similarity predictor and feature generator
    for j = 1; j <= n_batch; j++ do
        B = (X, Y, c)^b ⊂ D
        Θ^s = Θ^s − α∇_{Θ^s} J(Θ; B)
        Θ^g = Θ^g − α∇_{Θ^g} J(Θ; B)
    end for
    # Train feature label discriminator
    for j = 1; j <= n_batch; j++ do
        B = (X, Y, c)^b ⊂ D
        Θ^d = Θ^d − α∇_{Θ^d} J(Θ; B)
    end for
end for
```

5 EXPERIMENTAL SETUP

In this section, we first introduce the datasets and evaluation metrics used in our experiments in Section 5.1. Then, the details of the training procedure and parameter settings are presented in Section 5.2 and Section 5.3 respectively.

5.1 Datasets and Evaluation Metrics

To evaluate the effectiveness of our proposed model, we conduct experiments on two well-known sentence similarity tasks, i.e., answer selection and paraphrase identification as demonstrated in [42]. For ease of comparison with the state-of-the-art approaches, we adopt three representative datasets, namely TREC-QA, WikiQA and MSRP, which have been widely used in recent studies [5, 42, 49]. The details of the datasets are presented as follows.

Answer selection. Given a question and a list of candidate answers, the answer selection task is to rank the candidates according to their similarities with the question. Two widely used datasets, namely TREC-QA and WikiQA, are adopted in our experiments. TREC-QA was created by Wang et al. [40] based on the QA track (8-13) data of Text REtrieval Conference. WikiQA [44] is an open domain QA dataset in which all answers were collected from the Wikipedia. Both TREC-QA and WikiQA have the train, development and test sets, and each sample is labeled as 1 or 0 to indicate whether the candidate answer is right or wrong for a given question. The statistics of the datasets are presented in Table 1. The performance of answer selection is usually measured by the mean average precision (MAP) and mean reciprocal rank (MRR) [10, 37].

Paraphrase identification. The paraphrase identification task can be treated as a binary classification problem, and the goal is to judge whether two sentences are paraphrases or not according to their similarity. We utilize the Microsoft Research Paraphrase corpus (MSRP) [8] for experiment, which is constructed from a large corpus of temporally and topically clustered news articles. The MSRP dataset contains 4,076 sentence pairs in the training set, and 1,725 ones in the test set. Each sentence pair is labeled with 1 or 0 to indicate whether the two sentences are paraphrases or not. Since no development set is provided, we randomly select 100 positive pairs (labeled as 1) and 100 negative pairs (labeled as 0) from the training set as the development set. To evaluate the performance, two widely used metrics, namely accuracy (Acc) and F1 score are adopted [18, 46].

Table 1: Statistics of the datasets for answer selection. We remove all the questions with no right or wrong answers. "Avg QL" and "Avg AL" denote the average length of questions and answers.

Dataset		# of quetions	Avg QL	Avg AL
TREC-QA	Train	1162	7.57	23.21
	Dev	65	8.00	24.90
	Test	68	8.63	25.61
WikiQA	Train	873	7.16	25.29
	Dev	126	7.23	24.59
	Test	243	7.26	24.59

5.2 Training

We use the bidirectional LSTM (BLSTM) [17] model as the function in Formula (1) to obtain the hidden states, which can effectively mitigate the gradient vanish problem [34]. Since the answer sentence intuitively contains more information than the question sentence due to the larger length, the generator is built upon the *answer* sentence, which is denoted as $\mathbf{F_G}$-**A**. As to paraphrase identification, the generator is applied for the *right* sentence for the model consistency, namely $\mathbf{F_G}$-**R**. To obtain the representation of each sentence, we utilize the recently proposed attention mechanism [34], which generates the attentive answer representation based on the interaction with the question. Finally, the extracted common features and the Manhattan distance similarity between the two sentence representations are concatenated as a feature vector, and a softmax layer is utilized to predict the sentence similarity as demonstrated in Formula (3). For the answer selection task, the candidate answers are ranked by the predicted similarity $p(c = 1|X, Y)$. As to the paraphrase identification task, we apply a general threshold of 0.5 as used in [4]. To be specific, the predicted label is 1 when $p(c = 1|X, Y) > 0.5$. Otherwise, the predicted label is 0.

5.3 Parameter Settings

We use different word embeddings for different tasks. Specifically, for the answer selection task, we use the 100-dimensional GloVe word vectors[1], which are trained based on the global word co-occurrence [30]. For paraphrase identification, we concatenate the GloVe vectors with the 25-dimensional PARAGRAM vectors[2] that are developed for paraphrase tasks [43]. The dimension of the hidden state is set to 50. The parameter λ that controls the loss of the generator ranges from 0.1 to 1 with an increment of 0.1. We use the AdaDelta [48] algorithm for parameter update when training. In order to avoid overfitting, we apply a dropout rate of 0.5 for the concatenated feature. In addition, the early stopping is used during training and we keep the optimal parameters based on the best performance on the development set. Then, the optimal model is used for evaluation on the test set.

6 RESULTS AND ANALYSES

In this section, we present our experimental results and conduct extensive analyses. To be specific, we first investigate the effectiveness of our proposed model in Section 6.1. Then, we make more comparisons with the recent progress in answer selection and paraphrase identification in Section 6.2. After that, we investigate the influence of the parameter λ that controls the loss of the generator in Section 6.3. To affirm the effectiveness of our collaborative learning, we remove it from our CAN model and compare with the solely adversarial learning based model in Section 6.4. Finally, the influence of the generator source and the quality of our extracted common feature are studied in Section 6.5 and Section 6.6 respectively.

6.1 Effectiveness of CAN

To investigate the effectiveness of our proposed CAN model, the following two representative baselines are used for comparisons on all the three datasets:

- **BLSTM**: The BLSTM model calculates the sentence similarity by the Manhattan distance between the two sentence representations as Formula (2).
- **BLSTM-EHS**: Since our common feature is generated based on the elite hidden states (EHS), we treat the pooled elite hidden states as a feature, which is concatenated with the similarity score obtained by BLSTM and feed to a softmax layer for similarity prediction.

Table 2 and Table 3 show the performance of the above two baselines and our CAN model for answer selection and paraphrase identification respectively. The best result on each dataset is marked in bold. For TREC-QA and WikiQA, the percentages in parentheses indicate the improvements over the BLSTM model. In addition, we perform statistical tests based on the paired t-test at the 0.05 level. The symbols as $*$ and ▲ indicate significant improvements over BLSTM and BLSTM-EHS respectively.

Table 2: Performance of various models for answer selection on TREC-QA and WikiQA: (1) the percentages in parentheses indicate the improvements over the BLSTM model and (2) the symbols as $*$ and ▲ indicate significant improvements over BLSTM and BLSTM-EHS respectively, according to the paired t-test at the 0.05 level.

Model	TREC-QA		WikiQA	
	MAP	MRR	MAP	MRR
BLSTM	0.7528	0.8226	0.6979	0.7107
BLSTM-EHS	0.7967 (+5.83%)	0.8795 (+6.92%)	0.7116 (+1.96%)	0.7299 (+2.70)
CAN	**0.8410**$^{*▲}$ (+11.72%)	**0.9168**$^{*▲}$ (+11.45%)	**0.7303**$^{*▲}$ (+4.64%)	**0.7431**$^{*▲}$ (+4.56%)

Table 3: Performance of various models for paraphrase identification on MSRP. The symbols as $*$ and ▲ indicate significant improvements over BLSTM and BLSTM-EHS respectively, according to the paired t-test at the 0.05 level.

Model	MSRP	
	Acc (%)	F1 (%)
BLSTM	74.4	82.0
BLSTM-EHS	74.8	83.2
CAN	**77.2**$^{*▲}$	**84.0***

It is observed that we achieve significant improvements over the two baselines in most cases, by integrating our common feature extractor into the traditional BLSTM model. In particular, the maximum improvement over the BLSTM model is up to 11.72% in terms of MAP on TREC-QA. It is also notable that the BLSTM-EHS model

[1] http://nlp.stanford.edu/data/glove.6B.zip
[2] http://ttic.uchicago.edu/ wieting/

performs better than the BLSTM model, indicating the effectiveness of using the elite hidden states for boosting sentence similarity modeling. Whereas, the BLSTM-EHS model directly uses the pooled elite hidden states as additional features, which neglects their internal interactions in the hidden space. In contrast, our model extracts common features with an embedded extractor, where the interactions between the elite hidden states in two sentences are well modeled by a collaborative and adversarial game. Therefore, our proposed CAN model is superior to the BLSTM-EHS model for sentence similarity modeling on all the three datasets.

6.2 Comparison with the Recent Progress

In addition to the above two representative baselines, we compare our model with the recent progress in answer selection and paraphrase identification.

Results on TREC-QA and WikiQA. Table 4 and Table 5 report the results from prior work on TREC-QA and WikiQA respectively. The most related work to ours is [42], which explicitly modeled the similar and dissimilar parts of a sentence pair by lexical decomposition and composition. [5, 10, 37, 47] focused on the attention mechanism to capture the salient information for better sentence representations. However, the common features in a sentence pair were not well investigated in these studies. Since Wang, Liu, and Zhao [37] reported the results (marked with †) on the manually cleaned TREC-QA dataset with only 78 questions in the training set, we also present the corresponding results for a fair comparison in the last row. It is observed that we achieve substantial improvements over the recent progress on both TREC-QA and WikiQA. In particular, our proposed model outperforms the state-of-the-art approach on TREC-QA (full training set) with an absolute improvement of 0.04 in terms of MAP. For WikiQA, our proposed model is comparable to the state-of-the-art inner attention based RNN models [37] as shown in Table 5. Whereas, the inner attention based RNN models [37] did not perform very well on the cleaned TREC-QA dataset that has less training samples as indicated in Table 4. In contrast, our model is more robust by directly extracting the common features between two sentences for similarity modeling. Moreover, compared with the work which explicitly modeled the similar and dissimilar features of a sentence pair [42], our model is also much more effective in feature extraction by a collaborative and adversarial game between the generator and discriminator.

Table 4: Performance comparisons on TREC-QA. Note that the work marked with † uses the manually cleaned training set which has less questions.

System	MAP	MRR
Wang and Nyberg 2015 [38]	0.7134	0.7913
Wang, Liu, and Zhao 2016 [37] †	0.7369	0.8208
Wang and Ittycheriah 2015 [41]	0.7460	0.8200
Santos et al. 2016 [10]	0.7530	0.8511
Wang, Mi, and Ittycheriah 2016 [42]	0.7714	0.8447
Chen et al. 2017 [5]	0.7814	0.8513
Zhang et al. 2017 [49]	0.7864	0.8325
Rao, He, and Lin 2016 [32]	0.8010	0.8770
CAN	**0.8410**	**0.9168**
CAN †	0.7759	0.8403

Table 5: Performance comparisons on WikiQA.

System	MAP	MRR
Yang, Yih, and Meek 2015 [44]	0.6520	0.6652
Santos et al. 2016 [10]	0.6886	0.6957
Yin et al. 2015 [47]	0.6921	0.7108
(Rao, He, and Lin 2016 [32]	0.7010	0.7180
Wang, Mi, and Ittycheriah 2016 [42]	0.7058	0.7226
Chen et al. 2017 [5]	0.7212	0.7312
Wang, Liu, and Zhao 2016 [37]	**0.7341**	0.7418
CAN	0.7303	**0.7431**

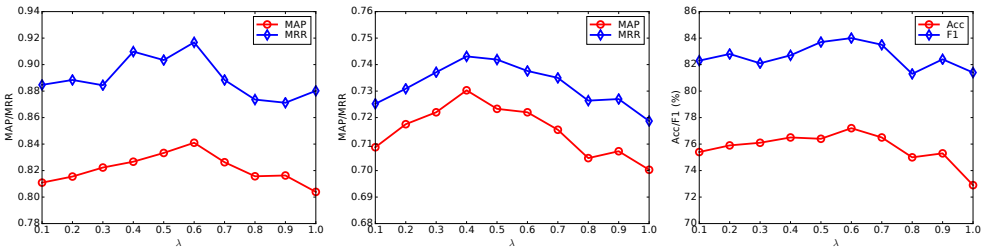
Table 6: Performance comparisons on MSRP.

	System	Acc (%)	F1 (%)
non-NN	Blacoe and Lapata 2012 [3]	73.0	82.3
	Wan et al. 2006 [36]	75.6	83.0
	Madnani et al. 2012 [25]	77.4	84.1
	Ji and Eisenstein 2013 [22]	**80.4**	**86.0**
NN	Hu et al. 2014 [20]	69.9	80.9
	Socher et al. 2011 [33]	76.8	83.6
	Yin and SchÅąÂźtze 2015 [46] - without pre-training	72.5	81.4
	Yin and SchÅąÂźtze 2015 [46] - with pre-training	78.1	84.4
	He, Gimpel, and Lin 2015 [18] - without POS embeddings	77.8	N/A
	He, Gimpel, and Lin 2015 [18] - with POS and Para. embeddings	78.6	84.7
	Wang, Mi, and Ittycheriah 2016 [42]	78.4	84.7
	CAN	77.2	84.0

Results on MSRP. The results from recent work on MSRP are summarized in Table 6, which can be divided into 2 groups: non-neural network (non-NN) based and neural network (NN) based. We observe that the non-NN based method [22] achieved the best performance. However, it relied on various hand-crafted features, which were not available in many cases. Recently, the NN based methods tend to attract more attention. To be specific, Yin and SchÅąÂźtze [46] presented a CNN based deep learning architecture, which modeled interaction features at multiple levels of granularity. However, their model relied much on the pre-training step. In [18], the CNN model was used for feature extraction at a multiplicity of perspectives, and achieved the best performance among the NN based methods. Whereas, their model was dependent on various components, such as the part-of-speech (POS) embeddings, the PARAGRAM (Para.) embeddings, multiple widths and similarity layers. The model which learned sentence similarity by lexical decomposition and composition [42] achieved similar performance as [18]. Regarding to our proposed CAN model, it performs slightly worse than the best NN based approaches [18, 42] on MSRP. However, our model does not need to tune various factors, such as the window size, perspective granularity [18], and various decomposition functions [42]. It is also notable that our model is comparable to the one that contains all the components except the POS embeddings in [18], indicating the effectiveness of using our common feature extractor for enhancing sentence similarity modeling.

6.3 Influence of λ

As described in previous studies [6, 29], the final loss in adversarial learning for multi-domain or multi-criteria is usually a linear combination of the task loss, generator loss and discriminator loss. To balance the various losses, a weight parameter is often used for the combination. In this paper, we use a parameter λ to control the



(a) TREC-QA (b) WikiQA (c) MSRP

Figure 3: Influence of the weight λ that controls the generator loss

generator loss as shown in Formula (10). Since the loss weight has not well been studied in previous work, we investigate the influence of the parameter in this section. Figure 3 shows the performance of our model with different settings for λ over three datasets. The value of λ ranges from 0.1 to 1.0 with a step of 0.1.

We observe that the general tendency for each evaluation metric is similar when λ grows on all the three datasets. To be specific, the performance improves with the increasing λ value at the beginning, and then declines. Although there is not a common optimal value for λ across all the datasets, a general value between 0.3 and 0.6 is recommended to be a reliable setting to control the generator loss as observed in our experiments. In this paper, we obtain the optimal value of λ according to the best performance on the development set if not otherwise specified. Since the weight for the task loss and discriminator loss is set to 1 as shown in Formula (10), it is indicated that a less weight for the generator loss can yield better performance, which is consistent with the empirical settings ($\lambda = 0.05$) in [6].

6.4 Influence of Collaborative Learning

As described in Section 4.3, our generator plays a collaborative and adversarial game with the discriminator for common feature extraction, which is quite different from the previous work that only involves adversarial learning [6, 16, 24, 29]. Particularly, the loss of our generator is defined according to the similarity label in the training set as shown in Formula (9). For a similar sentence pair ($c = 1$), the generator plays against the discriminator by generating more common features that are hard to tell which sentence they are generated from (i.e., *adversarial learning*). While given a dissimilar sentence pair ($c = 0$), the generator tries to make the generated features far away from the other sentence. In other words, it collaborates with the discriminator to facilitate the feature discrimination (i.e., *collaborative learning*). To further investigate the influence of the collaborative learning in our proposed model, we re-define the generator loss to make the model only incorporates adversarial learning into feature extraction, namely adversarial network (AN):

$$\max_{\Theta^g} \mathcal{J'}_g(\Theta^g) = \sum_{(X,Y,c)\in\mathcal{D}} \log p_d(l_{\mathbf{F}_G} = 1|\mathbf{F}_G) \tag{11}$$

Table 7 shows the performance of the models with and without collaborative learning. We observe that our proposed CAN model is superior to the AN model that only involves adversarial learning. In particular, compared with our CAN model, the performance of the AN model decreases by 0.0209 on TREC-QA in terms of MRR. This corresponds to our intuition that the generator needs to collaborate with the discriminator to generate less common features for the dissimilar sentence pairs. Furthermore, it is notable that the only difference between the two models lies in that we utilize additional collaborative learning for common feature extraction. Therefore, it has been confirmed that the collaborative learning plays an important role in our proposed model, which works together with the adversarial learning to extract better common features for sentence similarity modeling.

6.5 Influence of Generator Source

In the answer selection task, the answer sentence usually contains more information than the question sentence due to the larger length, thus we apply the generator upon the *answer* for common feature extraction, namely $\mathbf{F_G}$**-A** as demonstrated in Section 5.2. As to paraphrase identification, there is not a significant difference between the length of two sentences in a pair. For the model consistency, we also apply the generator on the *right* sentence for feature extraction, namely $\mathbf{F_G}$**-R**. To have a better understanding, we investigate the influence of different generator sources in this section. In other words, the generator will be applied on the *question* (or *left*) sentence for common feature extraction, namely $\mathbf{F_G}$**-Q** (or $\mathbf{F_G}$**-L**) in the answer selection (or paraphrase identification) task.

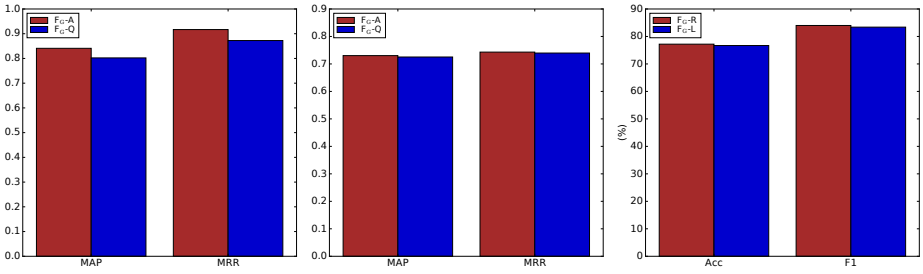
Figure 4 shows the performance of our proposed model with various generator sources on each dataset. It is observed that there is a slight performance decline in the TREC-QA dataset when using the generator upon the question, i.e., $\mathbf{F_G}$**-Q**. Regarding to the other two datasets, namely WikiQA and MSRP, there are no obvious differences between the models with different generator sources. These observations have further proved the effectiveness of our proposed model for common feature extraction. To be specific, although the generator is applied only on one of the two sentences in a pair, it can indeed extract the common features between two sentences by playing the collaborative and adversarial game with the discriminator. Therefore, our proposed model is very robust in common feature extraction with different generator sources.

6.6 Quality of Extracted Common Feature

In general, the good common feature should cover the common characteristics of a sentence pair, and the similarities with the individual features of two sentences are expected to be as close as possible. To have an insight of the quality of our extracted common

Table 7: Performance of the models with and without collaborative learning. The last row denotes the absolute performance decrease of the AN model that only involves adversarial learning compared with our CAN model.

Description	Model	TREC-QA		WikiQA		MSRP	
		MAP	MRR	MAP	MRR	Acc (%)	F1 (%)
Collaborative and adversarial learning	CAN	0.8410	0.9168	0.7303	0.7431	77.2	84.0
Adversarial learning	AN	0.8321	0.8959	0.7151	0.7302	76.0	82.2
Absolute change	-	-0.0089	-0.0209	-0.0152	-0.0129	-1.2	-1.8

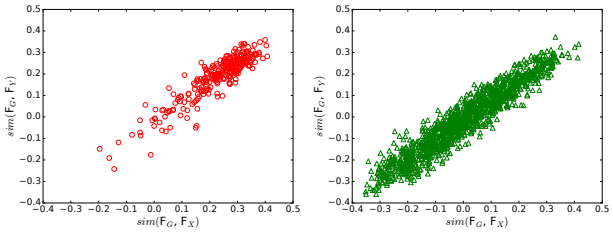


(a) TREC-QA (b) WikiQA (c) MSRP

Figure 4: Influence of the different generator sources

feature, we first calculate the similarities between the extracted common feature and the individual feature of each sentence on the test set of each dataset. The cosine metric is utilized, which has been widely used for word similarity calculation in the vector space [27]. Then, the similarities are visualized in a scatter plot, where the x axis denotes the similarity between the common feature and the individual feature of sentence X (i.e., $sim(\mathbf{F}_G, \mathbf{F}_X)$), and the y axis represents the similarity between the common feature and the individual feature of sentence Y (i.e., $sim(\mathbf{F}_G, \mathbf{F}_Y)$). For the space of limit, we only show the similarities on TREC-QA in Figure 5, and similar distributions can be observed on the other two datasets. In addition, we use different symbols to distinguish the similar and dissimilar sentence pairs as shown in the figure.

We observe that all the similarities are distributed quite closely to the line of $y = x$. In other words, the similarities between the extracted common feature and the individual feature of each sentence are nearly equal, indicating that the common characteristics have been captured by our common feature extractor. It is also worth noting that most of the red circles that denote the similar sentence pairs (in Figure 5(a)) are distributed in the upper right corner, while a large portion of the green triangles which denote the dissimilar pairs (in Figure 5(b)) are located in the bottom left corner. This indicates that the similarities corresponding to the similar sentence pairs are usually larger than that of the dissimilar ones. Moreover, it provides an intuitive understanding of why our extracted common features can help enhance sentence similarity modeling. All these observations have justified the good quality of our extracted common features.



(a) TREC-QA (similar) (b) TREC-QA (dissimilar)

Figure 5: Visualization of the similarities between the extracted common feature and the individual feature of each sentence on the test set of TREC-QA, namely $sim(\mathbf{F}_G, \mathbf{F}_X)$ and $sim(\mathbf{F}_G, \mathbf{F}_Y)$. Here X and Y denotes the question and answer sentence representatively, and $\mathbf{F}_G$ is generated based on the answer. The red circle denotes the similar sentence pairs, and the green triangle denotes the dissimilar ones.

7 CONCLUSIONS AND FUTURE WORK

In this paper, we propose a Collaborative and Adversarial Network (CAN) for enhancing sentence similarity modeling. In particular, our CAN model includes a common feature extractor, which focuses on extracting the common features between two sentences via both collaborative and adversarial learning. The experimental results show that our proposed CAN model is comparable to if not better than the state-of-the-art neural network based approaches for answer selection and paraphrase identification. It is also interesting to find that the generator source does not have a significant effect on the quality of the extracted common features. Furthermore, by comparing our proposed model with the one that only incorporates adversarial learning into common feature extraction, the effectiveness of using additional collaborative learning can be confirmed. In the future, we will investigate the effect of our proposed model on more tasks, such as biomedical IR [21], Web search [51] and textual entailment [26]. Another interesting research direction is to study the influence of the generator source on more tasks.

ACKNOWLEDGEMENTS

This research was funded by the National Nature Science Foundation of China (No. 61602179) and the Shanghai Municipal Commission of Economy and Informatization (No. 170513). The computation was performed in the Supercomputer Center of East China Normal University, China. This research was also supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada, an NSERC CREATE award in ADERSIM[3], the York Research Chairs (YRC) program and an ORF-RE (Ontario Research Fund-Research Excellence) award in BRAIN Alliance[4]. All the work was done when the first author was a visiting Ph.D. student at the Information Retrieval and Knowledge Management Research Lab, York University, Canada. Her visit was supported by an NSERC discovery grant. Part of the work was done when the second author was a professor at East China Normal University, China.

REFERENCES

[1] Eneko Agirre, Mona Diab, Daniel Cer, and Aitor Gonzalez-Agirre. 2012. Semeval-2012 task 6: a pilot on semantic textual similarity. In *SemEval*. 385–393.
[2] Weijie An, Qin Chen, Wei Tao, Jiacheng Zhang, Jianfeng Yu, Yan Yang, Qinmin Hu, Liang He, and Bo Li. 2017. ECNU at 2017 LiveQA track: learning question similarity with adapted long short-term memory networks. In *TREC*. 1–9.
[3] William Blacoe and Mirella Lapata. 2012. A comparison of vector-based representations for semantic composition. In *EMNLP-CoNLL*. 546–556.
[4] Qin Chen, Qinmin Hu, Jimmy Xiangji Huang, and Liang He. 2018. CA-RNN: using context-aligned recurrent neural networks for modeling sentence similarity. In *AAAI*. 9 pages.
[5] Qin Chen, Qinmin Hu, Jimmy Xiangji Huang, Liang He, and Weijie An. 2017. Enhancing recurrent neural networks with positional attention for question answering. In *SIGIR*. 993–996.
[6] Xinchi Chen, Zhan Shi, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial multi-criteria learning for Chinese word segmentation. In *ACL*. 1193–1203.
[7] Emily L Denton, Soumith Chintala, Rob Fergus, et al. 2015. Deep generative image models using a Laplacian pyramid of adversarial networks. In *NIPS*. 1486–1494.
[8] Bill Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised construction of large paraphrase corpora: exploiting massively parallel news sources. In *COLING*. 350–356.
[9] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. 2017. Adversarial feature learning. In *ICLR*. 1–18.
[10] Cıcero Nogueira dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. 2016. Attentive pooling networks. *arXiv preprint arXiv:1602.03609* (2016).
[11] Alexey Dosovitskiy, Jost Tobias Springenberg, and Thomas Brox. 2015. Learning to generate chairs with convolutional neural networks. In *CVPR*. 1538–1546.
[12] Ishan Durugkar, Ian Gemp, and Sridhar Mahadevan. 2017. Generative multi-adversarial networks. In *ICLR*. 1–14.
[13] Hanyin Fang, Fei Wu, Zhou Zhao, Xinyu Duan, Yueting Zhuang, and Martin Ester. 2016. Community-based question answering via heterogeneous social network learning. In *AAAI*. 122–128.
[14] Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *ICML*. 1180–1189.
[15] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *Journal of Machine Learning Research* 17, 59 (2016), 1–35.
[16] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *NIPS*. 2672–2680.
[17] Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks* 18, 5 (2005), 602–610.
[18] Hua He, Kevin Gimpel, and Jimmy J Lin. 2015. Multi-perspective sentence similarity modeling with convolutional neural networks.. In *EMNLP*. 1576–1586.
[19] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780.
[20] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *NIPS*. 2042–2050.
[21] Xiangji Huang and Qinmin Hu. 2009. A bayesian learning approach to promoting diversity in ranking for biomedical information retrieval. In *SIGIR*. 307–314.
[22] Yangfeng Ji and Jacob Eisenstein. 2013. Discriminative improvements to distributional sentence similarity. In *EMNLP*. 891–896.
[23] Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. 2015. An empirical exploration of recurrent network architectures. In *ICML*. 2342–2350.
[24] Zheng Li, Yu Zhang, Ying Wei, Yuxiang Wu, and Qiang Yang. 2017. End-to-end adversarial memory network for cross-domain sentiment classification. In *IJCAI*. 2237–2243.
[25] Nitin Madnani, Joel Tetreault, and Martin Chodorow. 2012. Re-examining machine translation metrics for paraphrase identification. In *NAACL HLT*. 182–190.
[26] Marco Marelli, Luisa Bentivogli, Marco Baroni, Raffaella Bernardi, Stefano Menini, and Roberto Zamparelli. 2014. Semeval-2014 task 1: evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. In *SemEval*. 1–8.
[27] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).
[28] Jonas Mueller and Aditya Thyagarajan. 2016. Siamese recurrent architectures for learning sentence similarity. In *AAAI*. 2786–2792.
[29] Zhongyi Pei, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. 2018. Multi-adversarial domain adaptation. In *AAAI*. 1–8.
[30] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: global vectors for word representation. In *EMNLP*. 1532–1543.
[31] Alec Radford, Luke Metz, and Soumith Chintala. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015).
[32] Jinfeng Rao, Hua He, and Jimmy Lin. 2016. Noise-contrastive estimation for answer selection with deep neural networks. In *CIKM*. 1913–1916.
[33] Richard Socher, Eric H Huang, Jeffrey Pennin, Christopher D Manning, and Andrew Y Ng. 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *NIPS*. 801–809.
[34] Ming Tan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2015. LSTM-based deep learning models for non-factoid answer selection. *arXiv:1511.04108* (2015).
[35] Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. 2015. Simultaneous deep transfer across domains and tasks. In *ICCV*. 4068–4076.
[36] Stephen Wan, Mark Dras, Robert Dale, and Cécile Paris. 2006. Using dependency-based features to take the "para-farce" out of paraphrase. In *ALTW*. 131–138.
[37] Bingning Wang, Kang Liu, and Jun Zhao. 2016. Inner attention based recurrent neural networks for answer selection. In *ACL*. 1288–1297.
[38] Di Wang and Eric Nyberg. 2015. A long short-term memory model for answer sentence selection in question answering. In *ACL*. 707–712.
[39] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. IRGAN: a minimax game for unifying generative and discriminative information retrieval models. In *SIGIR*. 515–524.
[40] Mengqiu Wang, Noah A Smith, and Teruko Mitamura. 2007. What is the Jeopardy model? A quasi-synchronous grammar for QA. In *EMNLP-CoNLL*. 22–32.
[41] Zhiguo Wang and Abraham Ittycheriah. 2015. FAQ-based question answering via word alignment. *arXiv preprint arXiv:1507.02628* (2015).
[42] Zhiguo Wang, Haitao Mi, and Abraham Ittycheriah. 2016. Sentence similarity learning by lexical decomposition and composition. *arXiv preprint arXiv:1602.07019* (2016).
[43] John Wieting, Mohit Bansal, Kevin Gimpel, Karen Livescu, and Dan Roth. 2015. From paraphrase database to compositional paraphrase model and back. *arXiv preprint arXiv:1506.03487* (2015).
[44] Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. WikiQA: a challenge dataset for open-domain question answering. In *EMNLP*. 2013–2018.
[45] Wen-tau Yih, Ming-Wei Chang, Chris Meek, and Andrzej Pastusiak. 2013. Question answering using enhanced lexical semantic models. In *ACL*. 1744–1753.
[46] Wenpeng Yin and Hinrich Schütze. 2015. Convolutional neural network for paraphrase identification. In *NAACL HLT*. 901–911.
[47] Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. 2015. ABCNN: attention-based convolutional neural network for modeling sentence pairs. In *arXiv preprint arXiv:1512.05193*.
[48] Matthew D Zeiler. 2012. ADADELTA: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701* (2012).
[49] Haotian Zhang, Jinfeng Rao, Jimmy Lin, and Mark D Smucker. 2017. Automatically extracting high-quality negative examples for answer selection in question answering. In *SIGIR*. 797–800.
[50] Xiaodong Zhang, Sujian Li, Lei Sha, and Houfeng Wang. 2017. Attentive interactive neural networks for answer selection in community question answering. In *AAAI*. 3525–3531.
[51] Jiashu Zhao, Jimmy Xiangji Huang, and Ben He. 2011. CRTER: using cross terms to enhance probabilistic information retrieval. In *SIGIR*. 155–164.
[52] Zhou Zhao, Hanqing Lu, Vincent W Zheng, Deng Cai, Xiaofei He, and Yueting Zhuang. 2017. Community-based question answering via asymmetric multi-faceted ranking network learning. In *AAAI*. 3532–3539.

[3] http://www.yorku.ca/adersim
[4] http://brainalliance.ca