IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received December 31, 2019, accepted February 3, 2020, date of publication February 11, 2020, date of current version February 19, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2973206

Enhanced Text Matching Based on Semantic Transformation

SHUTAO ZHANG[1,2], HAIBO TAN[1], LIANGFENG CHEN[1], AND BO LV[1]
[1]Hefei Institute of Physical Science, Chinese Academy of Sciences, Hefei 230031, China
[2]University of Science and Technology of China, Hefei 230026, China

Corresponding author: Liangfeng Chen (quinear@hfcas.ac.cn)

This work was supported by the Anhui Major Science and Technology Projects through the Project "Intelligence Service Platform for Medium, Small and Micro-Enterprises Based on Big Data" under Grant 711245801052.

ABSTRACT Text matching is the core of natural language processing (NLP) system. It's considered as a touchstone of the NLP, and it aims to find whether text pairs are equal in semantics. However, the semantic gap in text matching is still an open problem to solve. Inspired by successes of cycle-consistent adversarial network (CycleGAN) in image domain transformation, we propose an enhanced text matching method based on the CycleGAN combined with the Transformer network. Based on the proposed method, the text semantics in a source domain is transferred to a similar or different target domain, and the semantic distance between text pairs is decreased. Meanwhile, we demonstrate our method in paraphrase identification and question answer matching. The matching degree is computed by a standard text matching model to evaluate the transforming influence on narrowing the text semantic gap. The experiments show that our method achieves text domain adaptation, and the effects on different matching models are remarkable.

INDEX TERMS Cycle-consistent adversarial network, domain adaptation, semantic transformation, text matching, transformer network.

I. INTRODUCTION

Text matching is a core issue in many natural language processing (NLP) tasks, such as paraphrase identification (PI) [1], question answering (QA) [2], information retrieval (IR) [3] and many other tasks. In this paper, we focus on PI and QA matching. These tasks are to find whether text pairs are semantically equal. Due to the semantic gap problem, there is often a mismatch between text pairs. Especially in QA matching tasks, the questions may be matching in causality with the answers, rather than semantically matching such as synonym [4], which leads to a tremendous semantic distance between questions and answers.

To address the problem of semantic gap, most prior works proposed various deep matching methods. These methods depend on effective text representations, in which each word is represented as a distributed embedding vector via a deep neural network (DNN). For example, a recurrent neural network (RNN) [5] is able to learn a sequential representation for each text. Long Short-Term Memory (LSTM) network [6] is one of the popular variations of RNN and is widely used in text representation. The average of the hidden states in LSTM cells is used as text representation. A bidirectional LSTM (BiLSTM) [7] is similar to LSTM but exploits bidirectional LSTM cells on the sequence to get text representation. The convolutional neural network (CNN) [8] is also used to capture semantic features for the representation of words, which performs convolution on the text. CNN is originally invented for computer vision. While in the past years, CNN has achieved excellent results in text representation, and many studies [9]–[11] have shown better performance of CNN than RNN. Besides, some works [12]–[14] integrate RNN and CNN to combine the advantages from RNN and CNN. They usually utilize a CNN to build hidden vectors for the input texts, and then the vectors are fed into an RNN to obtain the final text representation. More recently, Transformer network [15] was proposed to learn sequential representation via a self-attention mechanism instead of the recurrent connection in RNN or convolution in CNN. Transformer network can speed up the training due to the ability of parallel computing, and it was proved to perform better than RNN and CNN in capturing semantic features [16].

The representation-based methods try to learn semantic representations for the text pairs and then conduct matching

The associate editor coordinating the review of this manuscript and approving it for publication was Ting Wang.

between the two semantic representations. Different from these methods, our approach focuses on transferring the semantic domain to improve the matching degree. To be specific, we employ a cycle-consistent adversarial network (CycleGAN) framework as the semantic transformation model. In contrast to most traditional CycleGAN architectures, we use the Transformer network as the generator. Meanwhile, a discriminator consisting of CNN estimates the probability that the transferred text comes from the target semantic domain. The generators and discriminators are adversarially trained with backpropagation. But the adversarial training is notoriously hard [17]–[19], the parameters of generators and discriminators cannot be updated in backpropagation sometimes. Thus, we modify the loss functions and use a gradient penalty to enforce the transferred texts to be semantically closer to the texts in target domain. Moreover, we conduct experiments on public datasets of paraphrase identification and question answer matching. We use different matching methods as the baselines and compare the performance between our method and baselines. The experiments on the task of paraphrase identification show that our method can achieve the similar semantic domain transformation, i.e. a question is transferred to a target question, and the matching degree is improved. For the other task, our approach transfers different domains, i.e. a question is transferred to an answer or an answer is transferred to a question. Then we achieve domain adaptation, i.e. we train the matching model in paraphrase identification for use in question answer matching. The experimental results show our method performs better than baselines.

Considering the universal application of natural language model based on semantic networks [20], the semantic transformation model is also promising for many other text approaches, such as text classification [21], text analysis [22], [23], etc.

Our main contributions are as follows:

1) The proposed text matching method focuses on the semantic transformation instead of the semantic representation from a novel perspective.

2) The proposed semantic transformation model combines Transformer network with CycleGAN framework perfectly.

3) The model is suitable for domain adaptation included the similar and different domain to narrow the text semantic gap.

The rest of the paper is organized as follows. Section II covers the related work on text matching and CycleGAN. Our proposed method is presented in Section III. In section IV, we conduct experiments on paraphrase identification and question answer matching. Finally, the conclusions are illustrated in Section V.

II. RELATED WORK

A. TEXT MATCHING

Many text matching methods focus on finding good semantic representations from the texts using DNN. There are a lot of works based on text representation. Huang *et al.* proposed a deep structured semantic model (DSSM) [24] for document retrieval. It is the earliest research that applies a DNN on text matching. DSSM uses a multilayer perceptron (MLP) [25] to map the texts to feature vectors in a low dimensional semantic space and then compares the similarity between documents and queries. Shen *et al.* further improved DSSM and proposed the convolutional DSSM (CDSSM) [26]. CDSSM replaces the MLP with CNN when learning text representations. Similarly, Palangi *et al.* proposed DSSM-LSTM [27] that uses LSTM to capture contextual semantics. Severyn and Moschitti proposed two distributional sentence models based on convolutional neural network (ConvNet) [28] for QA matching. Zhou *et al.* [29] also used a neural network to learn the semantic representations for questions and answers. Yin *et al.* proposed attention-based CNN (ABCNN) [30] for answer selection, PI and textual entailment tasks. This approach introduces three attention mechanisms that integrates mutual influence between texts into CNN. Chen *et al.* proposed an enhanced sequential inference model (ESIM) [31] for natural language inference. ESIM uses a BiLSTM to encode the premise and hypothesis. Tan *et al.* proposed Attention-Based LSTM [32]. The model utilizes neural attention mechanism with LSTM to capture the most important semantic information in a text. Kusner *et al.* proposed the Word Mover's Distance (WMD) [33] to measure the distance between text documents. Liu *et al.* proposed a hierarchical model for sentences matching and extended WMD to the Ordered WMD [34] that considers the positions of words in a sentence. Song *et al.* also proposed a positional convolution neural model (P-CNN) [35] that takes the sequential structures of sentences into account.

B. CYCLE-CONSISTENT ADVERSARIAL NETWORK

A CycleGAN [36] basically consists of two generative adversarial networks (GANs) [37]. It trains two generators coupled with two discriminators to learn two bidirectional mappings across image domains. The default generators are residual networks (ResNets) [38], and the default discriminators are PatchGANs [36]. The CycleGAN framework was proposed for image to image transformation (e.g. it transfers a zebra to a horse). It is a recent successful approach to learn a transformation between image domains, and the general idea of CycleGAN has been further developed and improved. Based on CycleGAN, Wu *et al.* [39] generated a cartoon face that transferred from a person face. Lu *et al.* [40] extended the CycleGAN to conditional CycleGAN for face image generation. It generates a high-res face image from the low-res image and allows control of the appearance of the generated face via the input attributes. Chang *et al.* [41] used the CycleGAN to generate handwritten Chinese characters. The CycleGAN consists of DenseNet blocks [42]. Brunner *et al.* [43] adapted CycleGAN to transfer a piece of music from a source genre to a target genre. The model adds additional discriminators to enforce the generators to keep the structure of the original music mostly intact. Engin *et al.* [44] proposed an enhanced CycleGAN by com-

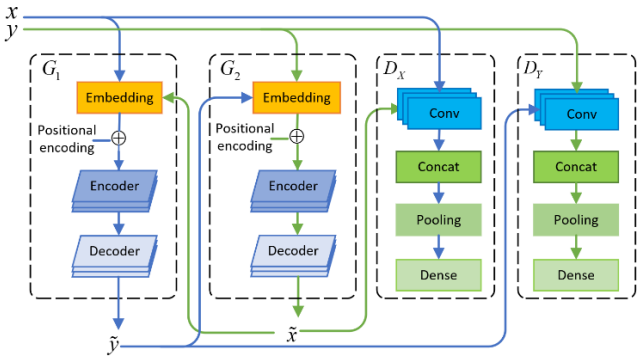


FIGURE 1. Architecture of semantic transformation model. Blue and green arrows denote the transformations across domains in both directions.

bining cycle consistency loss and perceptual loss to generate haze-free images. Wang *et al.* [45] translated the SAR patches to optical image patches. Gorti and Ma [46] used the Cycle-GAN to generate images from text.

III. APPROACH

A. SEMANTIC TRANSFORMATION

Our semantic transformation model is shown in Fig. 1. It contains two generators and two discriminators. The generators are responsible for transferring text from source domain to a target domain, while the discriminators are responsible for differentiating the transferred text from the text in target domain.

In particular, for text pair x and y that respectively from domain X and Y, the generator G_1 transfers text x to a new text $\tilde{y}$, such that the discriminator D_Y cannot tell whether the text $\tilde{y}$ comes from domain Y or another. At the same time, the generator G_2 tries to transfer text y to a new text $\tilde{x}$, while the discriminator D_X attempts to distinguish whether the text $\tilde{x}$ comes from domain X. Moreover, the transferred text $\tilde{x}$ and $\tilde{y}$ should be fed back to the generators to continue the transformations. The two transformations are defined as below:

$$\tilde{y} = G_1(x), \quad \tilde{x} = G_2(y) \tag{1}$$

The generator G_1 and G_2 employ Transformer network, which is the encoder-decoder architecture. For both the encoder and decoder, the generators use a stack of self-attention and feed-forward networks. The generators are illustrated in more detail. Firstly, the input text is encoded as an embedding matrix $M \in \mathbb{R}^{n \times d}$. The length of text is n and the dimension of word embedding is d. Then the matrix is combined with a position encoding to expose position information to the model. In addition, the self-attention in encoder employ m attention heads. Each attention head operates on the embedding matrix and computes a latent representation. Moreover, self-attention and the feed-forward networks are followed by dropout. Finally, the decoder receives the latent representation and use the self-attention mechanism to reconstruct it to a new text.

The discriminator D_X and D_Y are the regular CNN model, which consists of several convolutional layers, several max-pooling layers and several fully-connected layers. The inputs of discriminators include the original texts and the transferred texts. Firstly, the texts are encoded to matrixes (not shown in Fig. 1). Then the convolutional layer involves a filter $W \in \mathbb{R}^{k \times h}$, in which $k \times h$ is the filter size. The convolutional operation maps the matrix to a latent feature map $c = \sigma(W * x + b)$, where $\sigma(\cdot)$ is the activation function, $*$ is the convolutional operation, and b is the corresponding bias. Then these feature maps are concatenated together, followed with a max-pooling layer to take the maximum value. Finally, we use a full-connected layer that consists of DenseNet blocks to estimate the outputting results.

B. ADVERSARIAL TRAINING

To learn the two transformations in (1), we adopt an adversarial training strategy. In adversarial training, the generators and discriminators compete with each other. The adversarial training is accomplished by propagating the gradients of loss functions back to the generators and discriminators.

For the generators, the loss functions are shown as below:

$$L_G(D_Y, \tilde{y}) = -D_Y(\tilde{y}) + \gamma L_{cycle}(G_1, y, \tilde{x}) \tag{2}$$
$$L_G(D_X, \tilde{x}) = -D_X(\tilde{x}) + \gamma L_{cycle}(G_2, x, \tilde{y}) \tag{3}$$

where $D_X(\tilde{x})$ denotes the probability that text $\tilde{x}$ comes from domain X, and $D_Y(\tilde{y})$ denotes the probability of text $\tilde{y}$ coming from domain Y. L_{cycle} is the cycle consistency loss, γ is the corresponding weight.

The cycle consistency loss makes sure that the text is transferred back to the original text after passing through both generators, i.e. $x = G_2(G_1(x))$ and $y = G_1(G_2(y))$. In other words, the output text $\tilde{y}$ of the generator G_1 can be used as input to the generator G_2, and the output of the generator G_2 should be identical to the original text. Likewise, the output text $\tilde{x}$ of the generator G_2 can be fed as input to the generator G_1, and the output of the generator G_1 should be identical to the input of the generator G_2. To that end, we use the cross-entropy loss [47] to measure the difference between the text $G_1(\tilde{x})$ and the original text y, and vice versa.

$$L_{cycle}(G_1, y, \tilde{x}) = -\sum_{i=1}^{V} q_i(y) \log(q_i(G_1(\tilde{x}))) \tag{4}$$
$$L_{cycle}(G_2, x, \tilde{y}) = -\sum_{i=1}^{V} q_i(x) \log(q_i(G_2(\tilde{y}))) \tag{5}$$

where V is the vocabulary size, $q_i(\cdot)$ is probability that each word belongs to the i-th word in vocabulary. In function $q_i(\cdot)$, we use the one-hot label encoding [48] to encode the text, then the label smoothing regularization [49] is used to softly distribute the labels.

For the discriminators, the loss functions are shown as below:

$$L_D(D_X, x, \tilde{x}) = -D_X(x) + D_X(\tilde{x}) + \lambda p(D_X, x, \tilde{x}) \tag{6}$$
$$L_D(D_Y, y, \tilde{y}) = -D_Y(y) + D_Y(\tilde{y}) + \lambda p(D_Y, y, \tilde{y}) \tag{7}$$

where $D_X(x)$ denotes the probability that text x comes from domain X, and $D_Y(y)$ denotes the probability of text y coming from domain Y. $p(\cdot)$ is the gradient penalty function. λ is the penalty coefficient.

The gradient penalty is an additional regularization term for the discriminators to increase training stability further. The gradient penalty function is shown as below:

$$p(D_X, x, \tilde{y}) = (\|\nabla D_X(x + \alpha(x - \tilde{x}))\|_2 - 1)^2 \quad (8)$$

$$p(D_Y, y, \tilde{y}) = \left(\|\nabla D_Y(y + \alpha(y - \tilde{y}))\|_2 - 1\right)^2 \quad (9)$$

where ∇ denotes the gradient of discriminators, and α is a random number with uniform distribution in [0, 1].

The training target is to minimize the loss functions of the generators and discriminators. For generator G_1 and G_2, we freeze the discriminators, and then minimize the loss functions L_G, i.e. we should maximize the probability $D_Y(\tilde{y})$ and $D_X(\tilde{x})$ respectively. For discriminator D_X and D_Y, we freeze the generators, and then minimize the loss functions L_D, i.e. we should maximize the probability $D_X(x)$ and $D_Y(y)$ for the original text, and minimize the probability $D_X(\tilde{x})$ and $D_Y(\tilde{y})$ for the transferred text. The generators and discriminators are iteratively trained. Hence, G_1 and G_2 in our semantic transformation model is used to transfer the text semantic domain.

C. TEXT MATCHING

Suppose that we have two texts x and y that respectively from semantic domain X and Y. For text x, our semantic transformation model transfers text x to a new text $\tilde{y}$, and then we conduct matching between text $\tilde{y}$ and text y. The reverse is also true: we transfer text y to text $\tilde{x}$, and then we match text $\tilde{x}$ with text x. The text matching model learns the semantic representation of text and should decrease the semantic distance between text pairs as much as possible. To that end, in this paper, we use the semantic transformation model to transfer the texts to the same semantic domain. Given a fixed matching model F, the degree of matching is typically measured by the matching model on the representation of each text:

$$match(x, y) = F(\Phi(ST(x)), \Phi(y)) \quad (10)$$

$$\text{or } match(x, y) = F(\Phi(x), \Phi(ST(y))) \quad (11)$$

where $\Phi(\cdot)$ is a function that encodes text to an embedding matrix, ST denotes our semantic transformation model.

A potential interpretation for the text matching method is that it decreases the semantic distance between text pairs. The semantic distance between text x and text y is defined as:

$$SD(x, y) = \|\Phi(x) - \Phi(y)\|_2 \quad (12)$$

As text $\tilde{y}$ and text y come from the same semantic domain, and text $\tilde{x}$ and text x is also in the same semantic domain, the semantic distance between text $\tilde{y}$ and text y is less than the semantic distance between text x and text y, and the semantic distance between text $\tilde{x}$ and text x is also less than the semantic distance between text x and text y, i.e. $SD(\tilde{y}, y) < SD(x, y)$ and $SD(\tilde{x}, x) < SD(x, y)$. As a result, the matching degree is enhanced after transformation.

IV. EXPERIMENTS

We evaluate our method and baselines on two text matching tasks, i.e. paraphrase identification and question answer matching.

A. BASELINES

For comparison, the following representation-based text matching models are evaluated as baselines to demonstrate the superiority of our method.

DSSM: It is a semantic model based on DNN. A non-linear projection is performed to map the text pairs to a common semantic space. Then the matching degree of the texts is calculated as the cosine similarity between the vectors in the semantic space. In this paper, we set the parameters of DSSM as follows: the batch size is 1000, the epoch is 50, the embedding size is 100, the learning rate is 0.0005, and the dropout rate is 0.7.

ConvNet: It is composed of a single convolutional layer followed by a non-linearity and simple max-pooling layer. The convolutional operation is used to learn an embedding vector, the resulting vector representations can be used to calculate the similarity score between text pairs. In this paper, we set the parameters of ConvNet as follows: the batch size is 1024, the epoch is 50, the embedding size is 100, the learning rate is 0.0005, the dropout rate is 0.7, and the filter size is 3×100.

ABCNN: It learns semantic representations using CNN with attention mechanism. The model adds attention mechanism in the input layer to utilize mutual information between text pairs. In this paper, we set the parameters of ABCNN as follows: the batch size is 200, the epoch is 50, the embedding size is 100, the learning rate is 0.001, the dropout rate is 0.7. Besides, the model uses two convolutional layers, and the filter size is 3×100.

ESIM: It is an enhanced LSTM model. It was proposed for natural language inference. Here, we modify the ESIM network to fit the text matching task. The model uses a BiLSTM to encode the text pairs, followed by a matrix attention layer, a local inference layer, another BiLSTM inference composition layer, and finally a pooling operation before the output layer. In this paper, we set the parameters of ConvNet as follows: the batch size is 1024, the epoch is 50, the embedding size is 512, the learning rate is 0.0005, and the dropout rate is 0.7.

B. ARCHITECTURE PARAMETERS AND TRAINING

Since the adversarial training of our semantic transformation model is notoriously hard and requires a significant amount of hyperparameter tuning and training tricks. In this paper, we apply the following parameters for the generators and discriminators.

For generators, we use six encoder and decoder layers, the attention head number $m = 8$, the maximum length of text

$n = 15$, the dimension of word embedding $d = 512$, and a dropout rate is 0.1.

For discriminators, we use four convolutional layers, one max-pooling layer and two full-connected layers. The filter sizes in convolutional layers are 2×512, 3×512, 4×512, and 5×512. We use ReLu [50] as the activation function in convolutional layers.

We crossly train the generators and discriminators, the step for training the generators is one, and the step for training the discriminators is five. Both the generators and discriminators are trained for 30 epochs, and the batch size for each step is 32. We apply Adam optimizer [51] for the optimization of our model. The learning rate of generators is 0.0001, and the learning rate of discriminators is 0.0005. We set γ as 10 in loss function (2) and (3). Also, we set $\lambda = 10$ in loss function (6) and (7).

C. PARAPHRASE IDENTIFICATION

Paraphrase identification aims to match two questions that are semantically similar. This experiment is conducted to test our methods on matching homogenous texts. Here, we conduct our experiment on LCQMC dataset [52] and STS benchmark dataset [53]. The LCQMC dataset was released by Harbin Institute of Technology at COLING 2018. The dataset is a collection of question pairs and contains 125K positive instances (the question pairs are labeled as matching) and 125K negative instances (the question pairs are labeled as not matching). The STS benchmark dataset is a collection of text pairs with human-annotated similarity scores that range from 0 to 5. Since the paraphrase identification task is a binary classification problem, we modify STS benchmark to fit the text matching task. The modification is being done by collapsing the scores in range 0-1 to the label 0 (the text pairs are matching) and the scores in range 4-5 to the label 1 (the text pairs are not matching). We omit all the ambiguous text pairs associated with scores between 1 and 4. The modified STS benchmark dataset contains 2.0K positive instances and 1.8K negative instances.

We use accuracy and F1-score [54] as the metrics to evaluate the model performance. For brevity, the text pairs in each dataset are respectively from domain Q1 and Q2. Our semantic transformation model is trained on positive instances and applied to transfer the texts in Q1 to the texts in Q2. Then the baselines are used to evaluate the original text pairs and the transferred text pairs. The experimental results are listed in Table 1 and Table 2.

In Table 1 and Table 2, it can be seen that our method improves the accuracy and F1-score performance. Compared with baselines, our method achieves an accuracy of 64.50% (+2.85), 75.24% (+0.06), 76.92% (+0.46) and 80.43% (+1.5) over LCQMC dataset. On STS benchmark dataset, we achieve an accuracy of 59.29% (+1.24), 64.40% (+0.16), 63.37% (+1.56) and 64.71% (+0.82). The improvement of F1-score is 2.91%/2.37%, 0.35%/0.24%, 0.25%/0.87% and 1.44%/0.16% respectively. As a result, our method outperforms baselines slightly in terms of accuracy and F1-score.

TABLE 1. Results on LCQMC.

Model	Acc. (%)	F1 (%)
DSSM	61.65	68.60
DSSM + ST	**64.50**	**71.51**
ConvNet	75.18	77.45
ConvNet + ST	**75.24**	**77.80**
ESIM	76.46	80.16
ESIM + ST	**76.92**	**80.41**
ABCNN	78.93	81.26
ABCNN + ST	**80.43**	**82.70**

TABLE 2. Results on STS benchmark.

Model	Acc. (%)	F1 (%)
DSSM	58.05	59.57
DSSM + ST	**59.29**	**61.94**
ConvNet	64.24	59.12
ConvNet + ST	**64.40**	**59.36**
ESIM	61.81	59.72
ESIM + ST	**63.37**	**60.59**
ABCNN	63.89	63.34
ABCNN + ST	**64.71**	**63.50**

TABLE 3. Comparative results on LCQMC and DBQA.

Model	LCQMC		DBQA	
	Acc. (%)	F1 (%)	Acc. (%)	F1 (%)
DSSM	61.65	68.60	52.64	66.15
ConvNet	75.18	77.45	64.92	65.14
ESIM	76.46	80.16	50.34	44.65
ABCNN	78.93	81.26	66.01	67.14

D. QUESTION ANSWER MATCHING

The goal of QA matching is to determine whether the answer matches with question. This experiment is conducted to test our methods on matching heterogeneous texts. We conduct our experiment on dataset NLPCC-ICCPOL 2016 DBQA [55]. The dataset is a collection of QA pairs, and each pair is labeled as matching or not matching. It contains 15K positive instances and 290K negative instances. The data is seriously unbalanced, which can lead to an unreliable result in the binary classification task. Therefore, we choose 14K positive instances and 14K negative instances for the task.

The evaluation metrics are also accuracy and F1-score. We compare the performance of baselines that are respectively trained on LCQMC and DBQA. Table 3 summarizes the evaluation results.

From Table 3, we can see the accuracy and F1-score on DBQA are much less than the accuracy and F1-score on LCQMC. The accuracy is almost reduced by 10%. For example, in ESIM, the accuracy and F1-score even decrease to 50.34% (−26.12) and 44.65% (−35.51). This might be caused by the large semantic distance between the questions and answers in DBQA.

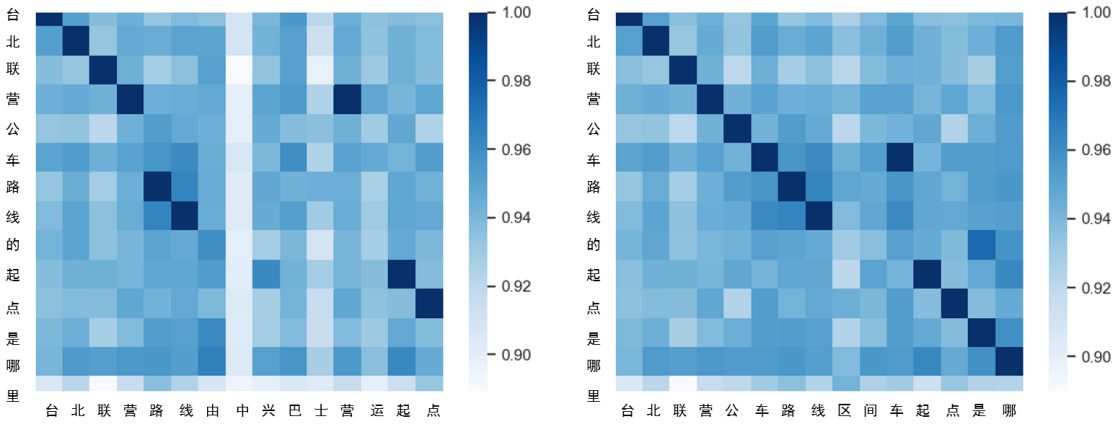


FIGURE 2. An example of QA matching degree. (Left) the original question and answer, (Right) the original question and transferred answer.

TABLE 4. Results on DBQA.

Model	Acc. (%)	F1 (%)
DSSM	48.85	50.84
DSSM + ST (Q→A)	**63.28**	**66.09**
DSSM + ST (A→Q)	**63.33**	**66.33**
ConvNet	50.29	22.77
ConvNet + ST (Q→A)	**66.37**	**56.11**
ConvNet + ST (A→Q)	**66.64**	**60.20**
ESIM	50.36	17.45
ESIM + ST (Q→A)	**82.64**	**82.36**
ESIM + ST (A→Q)	**80.57**	**80.20**
ABCNN	49.81	13.56
ABCNN + ST (Q→A)	**87.06**	**85.94**
ABCNN + ST (A→Q)	**86.69**	**85.30**

Therefore, we use the baselines trained on LCQMC to evaluate the text pairs in DBQA. Our semantic transformation model is trained on positive instances in DBQA. Then the baselines trained on LCQMC are used to evaluate the transferred text pairs. The results are shown in Table 4.

In Table 4, Q→A denotes that our semantic transformation model transfers questions in DBQA to answers, and then we conduct matching on answers and answers. Similarly, A→Q denotes that our model transfers answers in DBQA to questions, then we match questions with questions. As we can see, the performance of baselines that are trained on LCQMC is really poor, and the accuracy is even less than 50%. After transformation, the accuracy and F1-score are significantly enhanced. More specifically, when we transfer questions to answers, our method achieves an accuracy of 63.28% (+14.43), 66.37% (+16.18), 82.64% (+32.28) and 87.06% (+37.25%). When we transfer answer to question, our method achieves an accuracy of 63.33% (+14.48), 66.64% (+16.35), 80.57% (+30.21) and 86.69% (+36.80). From Table 3 and Table 4, the performance of our method is also improved. The relative improvement is about 10.64%/10.69%, 1.45%/1.72%, 32.3%/ 30.23% and 21.05%/20.68%. The F1-score is also improved significantly. As a result, our method can achieve different semantic domain transformation and enhanced the matching degree.

Furthermore, we take a pair of texts as an example (selected from the DBQA dataset) and illustrate the matching degree of the model with transformation and without transformation. The matching degree is visualized as Fig. 2.

Fig. 2 shows the heatmap of the semantic similarity matrix, which displays the matching degrees of original texts and transferred texts. The more semantically similar the word pair is, the deeper the color in the heatmap is. From the heatmap, we can observe the matching degree of transferred text pairs is more than the matching degree of original text pairs.

E. RESULT ANALYSIS

We have compared our method with baselines on paraphrase identification and question answer matching to evaluate their influence on performance. Table 1 to Table 4 show the comparison results on LCQMC, STS Benchmark and DBQA.

In the similar domain, Table 1 and Table 2 show that our method improves the text matching degree slightly. The best improvement of our method over the baselines is about 2.85% and 2.91% in terms of accuracy and F1-score. It corroborates that the transferred text is more similar to the text in domain Q2.

In the different domain, Table 3 and Table 4 show that our method outperforms the baselines significantly. The best improvement is about 32.3% and 37.71% in terms of accuracy and F1-score. In this case, our model achieves different domain transformation and the semantic transformation does produce positive impact for the text matching models.

In addition, in Table 4, we observe that our model achieves domain adaptation. The baselines without semantic transformation achieve the poor accuracy and F1-score, as they are evaluated on DBQA but trained on LCQMC. With semantic transformation, our model learns the semantic features and transfers the questions and answers on DBQA to the same domain as LCQMC.

V. CONCLUSION

In this paper, we present that text matching performance can be enhanced by semantic transformation of text pairs from a new perspective. We propose a semantic transformation model combined Transformer network with CycleGAN framework to achieve similar domain transformation and different domain transformation. In the model training phase, the loss functions of the generators and the discriminators are modified to improve the training stability. The experiments demonstrate that the semantic gap is narrowed, and the semantic matching is improved significantly. Meanwhile, the results show that the model can be used in different semantic domains to implement domain adaptation.

REFERENCES

[1] B. Dolan, C. Quirk, and C. Brockett, "Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources," in *Proc. 20th Int. Conf. Comput. Linguistics*, 2004, p. 350.

[2] E. M. Voorhees, "The TREC-8 question answering track report," *Trec*, vol. 99, pp. 77–82, Nov. 1999.

[3] H. Li and J. Xu, "Semantic matching in search," *Found. Trends Inf. Retr.*, vol. 7, no. 5, pp. 343–469, 2014.

[4] X. Yang, M. Khabsa, M. Wang, W. Wang, M. Khabsa, A. Awadallah, D. Kifer, and C. Lee Giles, "Adversarial training for community question answer selection based on multi-scale matching," 2018, *arXiv:1804.08058*. [Online]. Available: http://arxiv.org/abs/1804.08058

[5] J. L. Elman, "Finding structure in time," *Cognit. Sci.*, vol. 14, no. 2, pp. 179–211, Mar. 1990.

[6] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[7] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, 1997.

[8] Y. Kim, "Convolutional neural networks for sentence classification," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, Doha, Qatar, Oct. 2014, pp. 1746–1751.

[9] N. Thang Vu, H. Adel, P. Gupta, and H. Schutze, "Combining recurrent and convolutional neural networks for relation classification," 2016, *arXiv:1605.07333*. [Online]. Available: http://arxiv.org/abs/1605.07333

[10] K. Xu, Y. Feng, S. Reddy, S. F. Huang, and D. Y. Zhao, "Question answering on freebase via relation extraction and textual evidence," 2016, *arXiv:1603.00957*. [Online]. Available: https://arxiv.org/abs/1603.00957

[11] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," in *Proc. 34th Int. Conf. Mach. Learn.*, Sydney, NSW, Australia, vol. 70, Aug. 2017, pp. 933–941.

[12] S. Lai, L. Xu, K. Liu, and J. Zhao, "Recurrent convolutional neural networks for text classification," in *Proc. 29th AAAI Conf. Artif. Intell.*, Austin, TX, USA, Jan. 2015, pp. 2267–2273.

[13] C. Zhou, C. Sun, Z. Liu, and F. C. M. Lau, "A C-LSTM neural network for text classification," 2015, *arXiv:1511.08630*. [Online]. Available: http://arxiv.org/abs/1511.08630

[14] S. T. Hsu, C. Moon, P. Jones, and N. Samatova, "A hybrid CNN-RNN alignment model for phrase-aware sentence classification," in *Proc. 15th Conf. Eur. Chapter Assoc. Comput. Linguistics*, vol. 2, 2017, pp. 443–449.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017, *arXiv:1706.03762*. [Online]. Available: https://arxiv.org/abs/1706.03762

[16] G. Tang, M. Muller, A. Rios, and R. Sennrich, "Why self-attention? A targeted evaluation of neural machine translation architectures," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2018, pp. 4263–4272.

[17] K. Kurach, M. Lucic, X. Zhai, M. Michalski, and S. Gelly, "The GAN landscape: Losses, architectures, regularization, and normalization," in *Proc. Int. Conf. Mach. Learn.*, 2018. [Online]. Available: https://arxiv.org/abs/1807.04720

[18] N. Kodali, J. Abernethy, J. Hays, and Z. Kira, "On convergence and stability of GANs," 2017, *arXiv:1705.07215*. [Online]. Available: http://arxiv.org/abs/1705.07215

[19] K. Roth, A. Lucchi, S. Nowozin, and T. Hofmann, "Stabilizing training of generative adversarial networks through regularization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 2018–2028.

[20] J. Cong and H. Liu, "Approaching human language with complex networks," *Phys. Life Rev.*, vol. 11, no. 4, pp. 598–618, Dec. 2014.

[21] H. F. De Arruda, L. D. F. Costa, and D. R. Amancio, "Using complex networks for text classification: Discriminating informative and imaginative documents," *Europhys. Lett.*, vol. 113, no. 2, p. 28007, Jan. 2016.

[22] C. Akimushkin, D. R. Amancio, and O. N. Oliveira, "Text authorship identified using the dynamics of word co-occurrence networks," *PLoS ONE*, vol. 12, no. 1, Jan. 2017, Art. no. e0170527.

[23] D. R. Amancio, F. N. Silva, and L. D. F. Costa, "Concentric network symmetry grasps authors' styles in word adjacency networks," *Europhys. Lett.*, vol. 110, no. 6, Jun. 2015, Art. no. 68001.

[24] P. S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck, "Learning deep structured semantic models for Web search using clickthrough data," in *Proc. 22nd ACM Int. Conf. Conf. Inf. Knowl. Manage.*, San Francisco, CA, USA, 2013, pp. 2333–2338.

[25] D. W. Ruck, S. K. Rogers, and M. Kabrisky, "Feature selection using a multilayer perceptron," *J. Neural Netw. Comput.*, vol. 2, no. 2, pp. 40–48, 1990.

[26] Y. L. Shen, X. D. He, J. F. Gao, L. Deng, and G. Mesnil, "A latent semantic model with convolutional-pooling structure for information retrieval," in *Proc. 23rd ACM Int. Conf. Inf. Knowl. Manage.*, 2014, pp. 101–110.

[27] H. Palangi, L. Deng, Y. Shen, J. Gao, X. He, J. Chen, X. Song, and R. Ward, "Semantic modelling with long-short-term memory for information retrieval," 2014, *arXiv:1412.6629*. [Online]. Available: http://arxiv.org/abs/1412.6629

[28] A. Severyn and A. Moschitti, "Learning to rank short text pairs with convolutional deep neural networks," in *Proc. 38th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr. (SIGIR)*, New York, NY, USA, 2015, pp. 373–382.

[29] G. Zhou, Y. Zhou, T. He, and W. Wu, "Learning semantic representation with neural networks for community question answering retrieval," *Knowl.-Based Syst.*, vol. 93, pp. 75–83, Feb. 2016.

[30] W. Yin, H. Schütze, B. Xiang, and B. Zhou, "ABCNN: Attention-based convolutional neural network for modeling sentence pairs," *Trans. Assoc. Comput. Linguistics*, vol. 4, pp. 259–272, Dec. 2016.

[31] Q. Chen, X. D. Zhu, Z. H. Ling, S. Wei, H. Jiang, and D. Inkpen, "Enhanced LSTM for natural language inference," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2017, pp. 1657–1668.

[32] M. Tan, C. dos Santos, B. Xiang, and B. Zhou, "LSTM-based deep learning models for non-factoid answer selection," 2015, *arXiv:1511.04108*. [Online]. Available: http://arxiv.org/abs/1511.04108

[33] M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger, "From word embeddings to document distances," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 957–966.

[34] B. Liu, T. Zhang, F. X. Han, D. Niu, K. Lai, and Y. Xu, "Matching natural language sentences with hierarchical sentence factorization," in *Proc. Int. World Wide Web Conf. Committee*, 2018, pp. 1237–1246.

[35] Y. Song, Q. V. Hu, and L. He, "P-CNN: Enhancing text matching with positional convolutional neural network," *Knowl.-Based Syst.*, vol. 169, pp. 67–79, Apr. 2019.

[36] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," 2016, *arXiv:1611.07004*. [Online]. Available: http://arxiv.org/abs/1611.07004

[37] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 2014, pp. 2672–2680.

[38] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[39] R. Wu, X. Gu, X. Tao, X. Shen, Y.-W. Tai, and J. iaya Jia, "Landmark assisted CycleGAN for cartoon face generation," 2019, *arXiv:1907.01424*. [Online]. Available: http://arxiv.org/abs/1907.01424

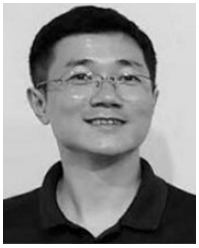
[40] Y. Lu, Y.-W. Tai, and C.-K. Tang, "Attribute-guided face generation using conditional CycleGAN," 2017, *arXiv:1705.09966*. [Online]. Available: http://arxiv.org/abs/1705.09966

[41] B. Chang, Q. Zhang, S. Pan, and L. Meng, "Generating handwritten chinese characters using CycleGAN," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, Mar. 2018, pp. 199–207.

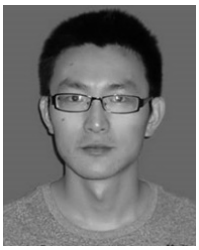
[42] F. Iandola, M. Moskewicz, S. Karayev, R. Girshick, T. Darrell, and K. Keutzer, "DenseNet: Implementing efficient convnet descriptor pyramids," 2014, *arXiv:1404.1869*. [Online]. Available: http://arxiv.org/abs/1404.1869

[43] G. Brunner, Y. Wang, R. Wattenhofer, and S. Zhao, "Symbolic music genre transfer with CycleGAN," in *Proc. Int. Conf. Tools Artif. Intell.*, 2018, pp. 786–793.
[44] D. Engin, A. Genc, and H. K. Ekenel, "Cycle-dehaze: Enhanced CycleGAN for single image dehazing," in *Proc. CVPR*, May 2018, pp. 825–833.
[45] L. Wang, X. Xu, Y. Yu, R. Yang, R. Gui, Z. Xu, and F. Pu, "SAR-to-optical image translation using supervised cycle-consistent adversarial networks," *IEEE Access*, vol. 7, pp. 129136–129149, 2019.
[46] S. Krishna Gorti and J. Ma, "Text-to-image-to-text translation using cycle consistent adversarial networks," 2018, *arXiv:1808.04538*. [Online]. Available: http://arxiv.org/abs/1808.04538
[47] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer-Verlag, 2006, p. 229.
[48] M. Cassel and F. L. Kastensmidt, "Evaluating one-hot encoding finite state machines for SEU reliability in SRAM-based FPGAs," in *Proc. IEEE Int. Line Test. Symp.*, Aug. 2006, pp. 139–144.
[49] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. CVPR*, 2016, pp. 2818–2826.
[50] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. Int. Conf. Mach. Learn.*, Haifa, Israel, 2010, pp. 807–814.
[51] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980
[52] X. Liu, Q. C. Chen, C. Deng, H. J. Zeng, J. Chen, D. F. Li, and B. Z. Tang, "LCQMC: A large-scale chinese question matching corpus," in *Proc. Int. Conf. Comput. Linguistics*, 2018, pp. 1952–1962.
[53] D. Cer, M. Diab, E. Agirre, N. Lopez-Gazpio, and L. Specia, "SemEval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation," in *Proc. 11th Int. Workshop Semantic Eval. (SemEval)*, 2017, pp. 1–14.
[54] M. Sokolova, N. Japkowicz, and S. Szpakowicz, "Beyond accuracy, F-score and ROC: A family of discriminant measures for performance evaluation," in *Proc. Australas. Joint Conf. Artif. Intell.*, 2006, pp. 1015–1021.
[55] N. Duan, "Overview of the NLPCC-ICCPOL 2016 shared task: Open domain chinese question answering," in *Proc. Int. Conf. Comput. Process. Oriental Lang.*, 2016, pp. 942–948.

SHUTAO ZHANG received the B.S. degree in computer science and technology from Jiangnan University, Wuxi, China, in 2018. He is currently pursuing the M.S. degree in computer applied technology with the University of Science and Technology of China, Hefei, China. His current research interests include artificial intelligence, natural language processing, and big data.



HAIBO TAN is currently a Professor with the Hefei Institute of Physical Science, Chinese Academy of Sciences, Hefei, China. His research interests include network security, artificial intelligence, and big data.



LIANGFENG CHEN received the Ph.D. degree in control theory from the University of Science and Technology of China, Hefei, China, in 2016. He is currently an Assistant Professor with the Hefei Institutes of Physical Science, Chinese Academy of Sciences, Hefei. His research interests include network security, natural language processing, deep learning, and big data.

BO LV is currently an Assistant Professor with the Hefei Institute of Physical Science, Chinese Academy of Sciences, Hefei, China. His research interests include artificial intelligence and big data.