Modelling Domain Relationships for Transfer Learning on Retrieval-based Question Answering Systems in E-commerce

Jianfei Yu*
Singapore Management University
jfyu.2014@phdis.smu.edu.sg

Minghui Qiu†
Alibaba Group, China
minghui.qmh@alibaba-inc.com

Jing Jiang
Singapore Management University
jingjiang@smu.edu.sg

Jun Huang, Shuangyong Song
Alibaba Group, China
{junhuang.jh,shuangyong.ssy}@alibaba-inc.com

Wei Chu
Alibaba Group, China
weichu.cw@alibaba-inc.com

Haiqing Chen
Alibaba Group, China
haiqing.chenhq@alibaba-inc.com

ABSTRACT

Nowadays, it is a heated topic for many industries to build automatic question-answering (QA) systems. A key solution to these QA systems is to retrieve from a QA knowledge base the most similar question of a given question, which can be reformulated as a paraphrase identification (PI) or a natural language inference (NLI) problem. However, most existing models for PI and NLI have at least two problems: They rely on a large amount of labeled data, which is not always available in real scenarios, and they may not be efficient for industrial applications.

In this paper, we study transfer learning for the PI and NLI problems, aiming to propose a general framework, which can effectively and efficiently adapt the *shared* knowledge learned from a resource-rich source domain to a resource-poor target domain. Specifically, since most existing transfer learning methods only focus on learning a shared feature space across domains while ignoring the relationship between the source and target domains, we propose to simultaneously learn shared representations and domain relationships in a unified framework. Furthermore, we propose an efficient and effective hybrid model by combining a sentence encoding-based method and a sentence interaction-based method as our base model. Extensive experiments on both paraphrase identification and natural language inference demonstrate that our base model is efficient and has promising performance compared to the competing models, and our transfer learning method can help to significantly boost the performance. Further analysis shows that the *inter-domain* and *intra-domain* relationship captured by our model are insightful. Last but not least, we deploy our transfer learning model for PI into our online chatbot system, which can bring in significant improvements over our existing system.

Finally, we launch our new system on the chatbot platform Eva[1] in our E-commerce site *AliExpress*[2].

KEYWORDS

Retrieval-based Question Answering, Transfer Learning, Domain Relationships Learning, Adversarial Training

1 INTRODUCTION

Question Answering (QA) systems have been widely developed and used in many domains. Examples of industry applications include Alibaba's AliME [18, 27], Microsoft's SuperAgent [7], Apple's Siri and Google's Google Assistant. Generally speaking, there are two kinds of commonly-used techniques behind most QA systems: Information Retrieval (IR)-based models [40] and generation-based models [32]. In this work, we focus on building up an IR-based QA system for automatically answering frequently asked questions (FAQs) in the E-commerce industry.

Fig. 1 illustrates the workflow of IR-based chatbot systems, where a key component is the *Question Rerank* module which reranks candidate questions in a question-answering knowledge base (KB) to find the best matching question given a question from a user. This task can be reduced to a paraphrase identification or a natural language inference problem. Take the query question and knowledge base shown in Fig. 1 for example. If we can detect that question C1 in the KB is a paraphrase of the query question, then we can take its answer as the answer for the query. In some cases, if we allow the matching question to be more general than the query question (i.e., entailed by the query question), we can also take the answer for question C2 in the KB as the query question's answer.

In the literature, paraphrase identification (PI) and natural language inference (NLI) have been extensively studied in the last decade [3, 4, 6, 30, 43]. However, when applying existing solutions to PI and NLI in chatbot systems in the E-commerce industry, there are at least two major challenges we face: (1) *Lack of rich training data*: All these solutions rely on a large amount of labeled data. However, it is generally time-consuming and costly to manually annotate sufficient labeled data for each domain. For example, different product categories might need different training data. (2)

*Work done during the internship at Alibaba Group.
†Corresponding author.

[1] https://gcx.aliexpress.com/ae/evaenglish/portal.htm?pageId=195440
[2] http://www.aliexpress.com/

Conference'17, July 2017, Washington, DC, USA
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5581-0/18/02…$15.00
https://doi.org/10.1145/3159652.3159685

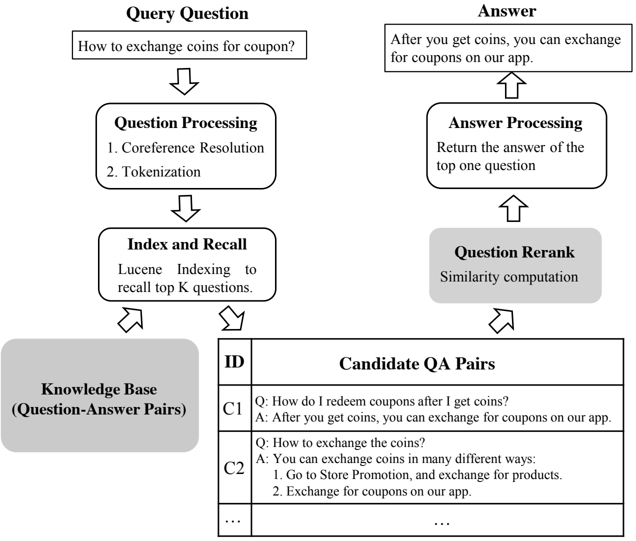
Query Question
How to exchange coins for coupon?
Question Processing
1. Coreference Resolution
2. Tokenization
Index and Recall
Lucene Indexing to recall top K questions.
Knowledge Base (Question-Answer Pairs)
Answer
After you get coins, you can exchange for coupons on our app.
Answer Processing
Return the answer of the top one question
Question Rerank
Similarity computation
ID
Candidate QA Pairs
C1
Q: How do I redeem coupons after I get coins?
A: After you get coins, you can exchange for coupons on our app.
C2
Q: How to exchange the coins?
A: You can exchange coins in many different ways:
1. Go to Store Promotion, and exchange for products.
2. Exchange for coupons on our app.
…
…

Figure 1: The workflow of IR-based QA systems.

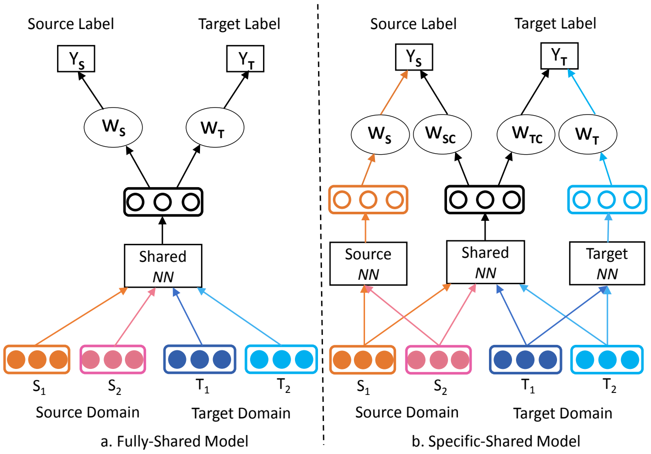

Figure 2: Existing Transfer Learning Frameworks.

Hard to reach a high QPS[3]. Most of the existing methods focus on improving the effectiveness or accuracy without paying much attention to efficiency. For real industry applications, when real-time responses are expected and a large number of customers are being served simultaneously, we need an efficient method to support a high QPS.

In this paper, we try to address the two challenges above. Specifically, we first make an empirical comparison of both the effectiveness and efficiency of several representative methods for modeling sentence pairs and propose an effective and efficient *hybrid model* as our base model. This ensures that we can achieve a high QPS. On top of the base model, we further design a new *transfer learning (TL) framework*, which is able to efficiently improve the performance on a resource-poor target domain by leveraging knowledge from a resource-rich source domain.

A Hybrid Base Model. Observing that LSTM-based methods [3, 6] are much more time-consuming than CNN-based methods [21, 43], we focus on CNN-based methods in this study. Meanwhile, there are typically two types of CNN-based methods for the task, namely sentence encoding (SE)-based methods [22, 42] and sentence interaction (SI)-based methods [13, 24]. We argue that these two types of methods may highly complement each other, and thus we propose a hybrid CNN model by combining an SE-based method [43] and an SI-based method [24]. Specifically, we modify the SE-based method using two element-wise comparison functions inspired by [21, 36] to match the two sentence embeddings, and then concatenate them together with sentence embeddings from the SI-based method.

Transfer Learning Framework. Transfer learning aims to apply knowledge gained in a source domain to help a target domain [23]. The key issue is how to transfer the shared knowledge from the source domain to the target domain while exclude the specific knowledge in the source domain based on the *domain relationship*. Most recent studies for TL in NLP perform *multi-task feature learning* by exploiting different NN models to capture a shared feature space across domains. As illustrated in Fig. 2a and Fig. 2b, one line of work employs a fully-shared framework to learn a shared representation followed by using two different fully connected layers for each domain [22, 41], while another line of work uses a specific-shared framework to learn not only a shared representation for both domains but also a domain-specific representation for each domain [19].

However, the first line of work simply assumes that two domains share the same feature space but ignore the domain-specific feature space. Although the latter one is capable of capturing both the shared and the domain-specific representations, it does not consider any relationships between the weights of the final output layer. Generally speaking, the weights on the output layer should capture both the *inter-domain* and the *intra-domain* relationships: (1) For the shared feature space across domains, since it is expected to be domain-independent, the weights corresponding to this feature space in the two domains should be positively related to each other; (2) For the shared and the domain-specific feature spaces in each domain, since they are expected to respectively capture domain-independent and domain-dependent features, their corresponding weights should be irrelevant to each other. Motivated by such an intuition, in this paper, we propose a new transfer learning method by explicitly modeling the domain relationships via a covariance matrix, which imposes a regularization term on the weights of the output layer to uncover both the inter-domain and the intra-domain relationships. Besides, to make the shared representation more invariant across domains, we follow some recent work on adversarial networks [11, 19] and introduce an adversarial loss on the shared feature space in our method. Fig. 3 gives an outline of our full model.

To evaluate our proposed method, we conduct both intrinsic evaluation and extrinsic evaluation.

Intrinsic Evaluation. We conduct extensive experiments on both a benchmark dataset and our own dataset. (1) The hybrid CNN model is shown to be not only efficient but also effective, in comparison with several representative methods; (2) Our proposed

[3] Queries Per Second

transfer learning method can bring significant improvements over the base model without transfer learning, and outperform existing TL frameworks including the widely used fully-shared model and the recently proposed specific-shared model; (3) Further analysis on our learned correlation matrix shows that our method is able to capture the inter-domain and intra-domain relationships.

Extrinsic Evaluation. We deploy our proposed hybrid CNN-based transfer learning model into our online chatbot system, which is deployed on a real E-commerce site *AliExpress*. Both the offline and the online evaluations show that our new system can significantly outperform the existing online chatbot system. Finally, we launch our new system on Eva[4], a chatbot platform in *AliExpress*.

2 MODEL

In this section, we present our general model for paraphrase identification and natural language inference, which will be used for question reranking in our chatbot-based QA system.

2.1 Problem Formulation and Notation

Our model is designed to address the following general problem. Given a pair of sentences, we would like to identify their semantic relation. For paraphrase identification (PI), the semantic relation indicates whether or not the two sentences express the same meaning [42]; for natural language inference (NLI), it indicates whether a hypothesis sentence can be inferred from a premise sentence [3].

Formally, assume there are two sentences $\mathbf{X}_1 = (\mathbf{x}_1^1, \mathbf{x}_2^1, \ldots, \mathbf{x}_m^1)$ and $\mathbf{X}_2 = (\mathbf{x}_1^2, \mathbf{x}_2^2, \ldots, \mathbf{x}_n^2)$, where $\mathbf{x}_i^j$ denotes an l-dimensional dense embedding vector retrieved from a lookup table $\mathbf{E} \in \mathbb{R}^{l \times |\mathcal{V}|}$ for all the words in the vocabulary $\mathcal{V}$. Our task is to predict the semantic label y which indicates the relation between $\mathbf{X}_1$ and $\mathbf{X}_2$. For PI, we assume the label y to be either *paraphrase* or *not paraphrase*; for NLI, we assume y to be either *neutral*, *entailment* or *contradiction*.

We consider a transfer learning setting, where we have a set of labeled sentence pairs from a source domain and a target domain, respectively, denoted by $\mathcal{D}^{\mathrm{s}}$ and $\mathcal{D}^{\mathrm{t}}$. Note that $|\mathcal{D}^{\mathrm{s}}|$ is assumed to be much larger than $|\mathcal{D}^{\mathrm{t}}|$. We seek to use both $\mathcal{D}^{\mathrm{s}}$ and $\mathcal{D}^{\mathrm{t}}$ to train a good model so that it can work well in the target domain.

To solve such a problem, a widely used transfer learning method (as illustrated in Fig. 2a) is to use the same NN model to transform every pair of input sentences in both domains into a hidden representation $\mathbf{z_c} \in \mathbb{R}^q$, where q is the size of the hidden representations. To facilitate our discussion, let us assume $\mathbf{z_c} = f_{\Theta_c}(\mathbf{X}_1, \mathbf{X}_2)$, where f_{Θ_c} denotes the transformation function parameterized by Θ_c. Next, for the source and the target domains, we assume that two fully connected layers are separately learned to map $\mathbf{z_c}$ to label y.

$$p(y \mid \mathbf{z_c}) = \begin{cases} \mathrm{softmax}(\mathbf{W_{sc}} \mathbf{z_c} + \mathbf{b_s}) & \text{if } y \text{ from src,} \\ \mathrm{softmax}(\mathbf{W_{tc}} \mathbf{z_c} + \mathbf{b_t}) & \text{if } y \text{ from tgt,} \end{cases}$$

where $\mathbf{W_{sc}} \in \mathbb{R}^{|Y| \times q}$ and $\mathbf{W_{tc}} \in \mathbb{R}^{|Y| \times q}$ are weight matrices and $\mathbf{b_s} \in \mathbb{R}^{|Y|}$ and $\mathbf{b_t} \in \mathbb{R}^{|Y|}$ are bias vectors.

Besides, another transfer learning approach [19] was recently proposed to use a domain-shared NN model and two domain-specific NN models to obtain a shared embedding $\mathbf{z_c}$ and two domain-specific embeddings $\mathbf{z_s} = f_{\Theta_s}(\mathbf{X}_1, \mathbf{X}_2)$ and $\mathbf{z_t} = f_{\Theta_t}(\mathbf{X}_1, \mathbf{X}_2)$. Therefore, the output layers are defined as:

$$p(y \mid \mathbf{z_c}, \mathbf{z_s}) = \begin{cases} \mathrm{softmax}(\mathbf{W_{sc}} \mathbf{z_c} + \mathbf{W_s} \mathbf{z_s} + \mathbf{b_s}) & \text{if } y \text{ from src,} \\ \mathrm{softmax}(\mathbf{W_{tc}} \mathbf{z_c} + \mathbf{W_t} \mathbf{z_t} + \mathbf{b_t}) & \text{if } y \text{ from tgt.} \end{cases}$$

The main limitation of the fully-shared framework is that it ignores source-specific or target-specific features. While for the specific-shared framework, it fails to consider any inherent correlations between the weights on the output layers. Therefore, we will introduce our proposed method that explicitly incorporates such correlations into the specific-shared framework in the next session.

2.2 Proposed Transfer Learning Method

Our goal is to model the inter-domain relationship between $\mathbf{W_{sc}}$ and $\mathbf{W_{tc}}$, and the intra-domain relationship between $\mathbf{W_s}$ and $\mathbf{W_{sc}}$ as well as $\mathbf{W_t}$ and $\mathbf{W_{tc}}$. Hence, we first reshape each weight matrix into a vector $\mathbf{w_d} \in \mathbb{R}^{q|Y|}$, followed by concatenating all the four reshaped vectors to form a new matrix $\mathbf{W} \in \mathbb{R}^{q|Y| \times 4}$, where each column corresponds to one weight matrix of the output layer.

Next, to capture the domain relationships mentioned above, we introduce a covariance matrix $\Omega \in \mathbb{R}^{4 \times 4}$. Note that each element $\Omega_{\mathbf{i,j}}$ indicates the correlation between $\mathbf{W_i}$ and $\mathbf{W_j}$, where $\mathbf{i}$ and $\mathbf{j}$ are one of $\mathbf{s}$, $\mathbf{sc}$, $\mathbf{t}$ and $\mathbf{tc}$. Inspired by a general multi-task relationship learning framework as introduced in [46], we consider confining the output layer's weights with Ω by using $\mathbf{tr}(\mathbf{W}\Omega^{-1}\mathbf{W}^T)$, where $\mathbf{tr}(\cdot)$ is the trace of a square matrix. This means that if $\Omega_{\mathbf{i,j}}$ is a large positive/negative value, $\mathbf{W_i}$ and $\mathbf{W_j}$ will be positively/negatively related to each other; otherwise if $\Omega_{\mathbf{i,j}}$ is close to zero, $\mathbf{W_i}$ and $\mathbf{W_j}$ will be irrelevant to each other. Note that to the best of our knowledge, we are the first to apply the multi-task relationship learning framework into NN-based transfer learning methods.

In order to simultaneously learn our model parameters and the domain relationships in a unified framework, we formulate our loss function as follows:

$$\begin{aligned} \mathcal{L} = & \sum_{\mathbf{k} \in \mathbf{s,t}} -\frac{1}{n_{\mathbf{k}}} \sum_{i=1}^{n_{\mathbf{k}}} \log p\big(y_i \mid \mathbf{X}_1^i, \mathbf{X}_2^i\big) + \frac{\lambda_1}{2} \mathbf{tr}(\mathbf{W}\Omega^{-1}\mathbf{W}^T) \\ & + \frac{\lambda_2}{2} ||\mathbf{W}||_F^2 + \frac{\lambda_3}{2} ||\Theta_c||_F^2 + \frac{\lambda_4}{2} ||\Theta_s||_F^2 + \frac{\lambda_5}{2} ||\Theta_t||_F^2 \\ & \text{s.t.} \quad \Omega \geq 0, \quad \mathbf{tr}(\Omega) = 1. \end{aligned} \tag{1}$$

where λ_1, λ_2, λ_3, λ_4 and λ_5 are regularization parameters, Ω is required to be positive semi-definite, and $\mathbf{tr}(\Omega)$ is required to be 1 without losing generality. In the above formulation, the first term refers to the cross-entropy loss for both domains, and the second term serves as a domain-relationship regularizer to constrain the weights on the output layer. The remaining terms are standard L2-regularization terms.

2.3 Adversarial Loss

Our above transfer learning method is based on the specific-shared framework, which is assumed to well capture the shared and domain-specific feature spaces. However, as suggested by [19], the shared representation learned in this framework may still contain noisy domain-specific features. Therefore, to eliminate the noisy features, here we also consider incorporating an adversarial loss on the

[4] Eva can be accessed via the following link: https://gcx.aliexpress.com/ae/evaenglish/portal.htm?pageId=195440

shared feature space so that the trained model can not distinguish between the source and target domains on it [11].

First, we assume that the shared layer $\mathbf{z_c}$ is mapped to a binary domain label d, which indicates whether $\mathbf{z_c}$ comes from the source or the target domain:

$$p(d \mid \mathbf{z_c}) = \text{softmax}(\mathbf{W_d}\mathbf{z_c} + \mathbf{b_d}).$$

Since the goal of adversarial training is to encourage the shared feature space indiscriminate across two domains, we define the adversarial loss as minimizing the negative entropy of the predicted domain distribution, which is different from maximizing the negative cross-entropy as in [11, 19]:

$$\boldsymbol{\ell} = \sum_{\mathbf{k}\in \mathbf{s},\mathbf{t}} \Big(\frac{1}{n_{\mathbf{k}}} \sum_{i=1}^{n_{\mathbf{k}}} \sum_{j=0}^{1} p(d_{ij} \mid \mathbf{X}_1^i, \mathbf{X}_2^i) \log p(d_{ij} \mid \mathbf{X}_1^i, \mathbf{X}_2^i)\Big). \quad (2)$$

Finally, we obtain a combined objective function as follows:

$$\begin{aligned} \min_{\substack{\Omega, \mathbf{W}, \mathbf{W_d}, \Theta_c, \\ \Theta_s, \Theta_t, \mathbf{b_s}, \mathbf{b_t}, \mathbf{b_d}}} \quad & \mathcal{L} + \lambda_0 \boldsymbol{\ell} \\ \text{s.t.} \quad & \Omega \geq 0, \quad \mathbf{tr}(\Omega) = 1, \end{aligned}$$

where λ_0 is a hyper-parameter for tuning the importance of the adversarial loss. As suggested by [46], it is not easy to optimize such a semi-definite programming problem. We will present an alternating training approach in Section 2.5 for solving it efficiently.

2.4 Base Model

Although the proposed transfer learning method is general and any neural networks for modeling a pair of sentences can be applied to it, we further target at proposing an efficient and effective base model for encoding a pair of sentences.

On one hand, although various attention-based LSTM architectures have been proposed to achieve a superior performance on both PI and NLI [6, 25, 29, 36], these models are very time-consuming due to the computation of memory cells and attention weights in each time step, which may not satisfy the industry demand, especially when QPS is high. On the other hand, CNN-based models are proven to be efficient, hence are the focus of our study. Most existing CNN-based models can be categorized into two groups: sentence encoding (SE)-based methods and sentence interaction (SI)-based methods. The former aims to first learn good representations for each sentence, followed by using a comparison function to transform them into a single representation [22, 42], while the latter tries to directly model the interaction between two sentences at the beginning and then makes abstractions on top of the interaction output [13, 24]. Observing that the two lines of methods focus on different perspectives to model sentence pairs, we expect that a combination of them can capture both good sentence representations and rich interaction structures.

Hence, we propose a hybrid CNN (hCNN) model, which are based on some minor modifications of two existing models: a SE-based BCNN model [43] and a SI-based Pyramid model [24]. Fig. 3 depicts our full transfer learning framework, which contains one shared hCNN and two domain-specific hCNNs. Below we briefly go through the architecture of hCNN. Note that in our implementation and the model description below, we pad the two input sentences to the same length m.

Modified BCNN: Following the original BCNN [43], we first use two separate 1-D convolutional (conv) and 1-D max-pooling layers to encode the two input sentences into two sentence embeddings:

$$\mathbf{h_1} = \mathit{CNN}(\mathbf{X}_1); \quad \mathbf{h_2} = \mathit{CNN}(\mathbf{X}_2).$$

Furthermore, as suggested by [21, 36] that element-wise comparison can work well on the problem, we use two comparison functions to match the two sentence embeddings, and then concatenate them together with the sentence embeddings as the sentence pair representation: $\mathbf{H_b} = \mathbf{h_1} \oplus \mathbf{h_2} \oplus (\mathbf{h_1} - \mathbf{h_2}) \oplus (\mathbf{h_1} \cdot \mathbf{h_2})$, where $-$ and $\cdot$ refer to element-wise subtraction and element-wise multiplication, and $\oplus$ refers to concatenation. Note that this setting is different from the original BCNN, which yields better performance in our empirical experiments.

Pyramid: As shown in the rightmost part of Fig 3, we first produce an interaction matrix $\mathbf{M} \in \mathbb{R}^{m \times m}$, where $\mathbf{M_{i,j}}$ denotes the similarity score between the i^{th} word in $\mathbf{X}_1$ and the j^{th} word in $\mathbf{X}_2$. Following [24], we use dot-product to compute the similarity score.

Next, by viewing the interaction matrix as an image, we stack two 2-D convolutional layers and two 2-D max-pooling layers on it to obtain the hidden representation $\mathbf{H_p}$.

Finally, we concatenate the two hidden representations as the final representation for each input sentence pair: $\mathbf{z} = \mathbf{H_b} \oplus \mathbf{H_p}$.

2.5 Inference

In our combined objective function, we have nine parameters Ω, $\mathbf{W}$, $\mathbf{W_d}$, $\mathbf{b_s}$, $\mathbf{b_t}$, $\mathbf{b_d}$, Θ_c, Θ_s and Θ_t, and it is not easy to optimize them at the same time. Following the practice in [46], we employ an alternating stochastic method, i.e., first optimizing the other eight parameters by fixing Ω, and then alternatively optimizing Ω by fixing the others in each iteration. The details are given as below:

Updating $\mathbf{W}$, $\mathbf{W_d}$, $\mathbf{b_s}$, $\mathbf{b_t}$, $\mathbf{b_d}$, Θ_c, Θ_s and Θ_t. While fixing Ω, the optimization problem becomes:

$$\begin{aligned} \min_{\substack{\mathbf{W}, \mathbf{W_d}, \Theta_c, \Theta_s, \\ \Theta_t, \mathbf{b_s}, \mathbf{b_t}, \mathbf{b_d}}} & \sum_{\mathbf{k}\in \mathbf{s},\mathbf{t}} \Big(\frac{1}{n_{\mathbf{k}}} \sum_{i=1}^{n_{\mathbf{k}}} \big(-\log p(y_i \mid \mathbf{X}_1^i, \mathbf{X}_2^i) \\ & + \lambda_0 \sum_{j=0}^{1} p(d_{ij} \mid \mathbf{X}_1^i, \mathbf{X}_2^i) \log p(d_{ij} \mid \mathbf{X}_1^i, \mathbf{X}_2^i)\big)\Big) + \frac{\lambda_1}{2} \mathbf{tr}(\mathbf{W}\Omega^{-1}\mathbf{W}^T) \\ & + \frac{\lambda_2}{2} ||\mathbf{W}||_F^2 + \frac{\lambda_3}{2} ||\Theta_c||_F^2 + \frac{\lambda_4}{2} ||\Theta_s||_F^2 + \frac{\lambda_5}{2} ||\Theta_t||_F^2 \end{aligned}$$

Since it is a smooth function, we can easily compute its partial derivatives with respect to the eight parameters.

Updating Ω. After fixing the eight parameters, the optimization problem is as follows:

$$\begin{aligned} \min_{\Omega} \quad & \mathbf{tr}(\mathbf{W}\Omega^{-1}\mathbf{W}^T) \\ \text{s.t.} \quad & \Omega \geq 0, \quad \mathbf{tr}(\Omega) = 1. \end{aligned}$$

As proved by [46], the above optimization problem has an analytical solution $\Omega = \frac{(\mathbf{W}^T\mathbf{W})^{\frac{1}{2}}}{\mathbf{tr}\left((\mathbf{W}^T\mathbf{W})^{\frac{1}{2}}\right)}$.

Finally, we present the whole procedure for training our full model as in Algorithm 1. Note that we only update Ω when we scan all the target training instances once.

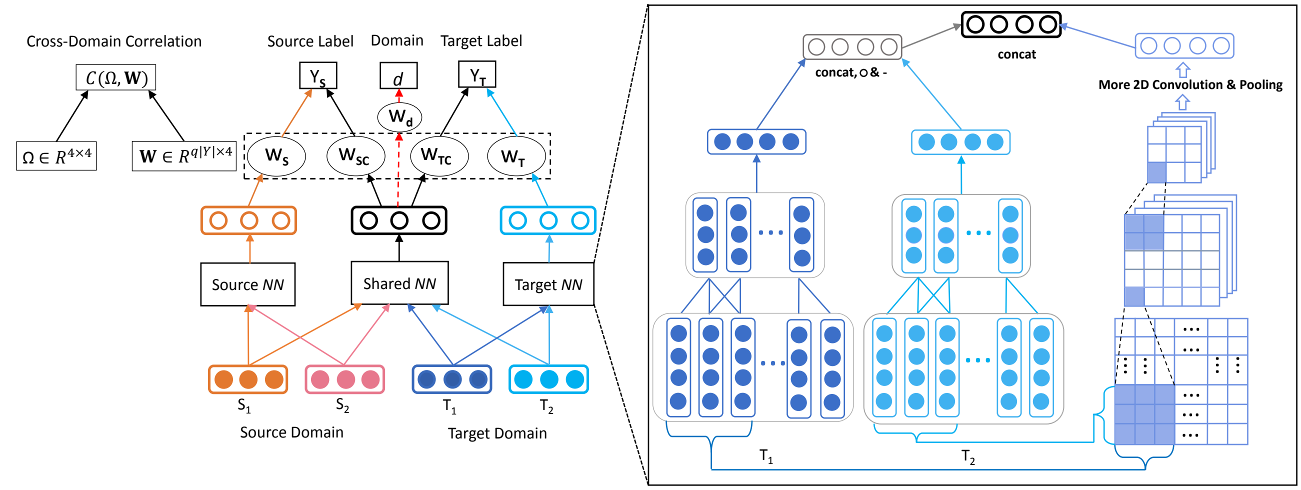

Figure 3: Our Full Transfer Learning Model for Paraphrase Identification and Natural Language Inference.

Algorithm 1 Training Procedure for our Full Model

```
Input: source training instances $\mathcal{D}^s$, target training instances $\mathcal{D}^t$ ($|\mathcal{D}^s| \gg |\mathcal{D}^t|$) .
Output: $\Omega$, $\mathbf{W}$, $\mathbf{W_d}$, $\mathbf{b_s}$, $\mathbf{b_t}$, $\mathbf{b_d}$, $\Theta_c$, $\Theta_s$, $\Theta_t$
Initialize $\mathbf{W}$, $\mathbf{W_d}$, $\mathbf{b_s}$, $\mathbf{b_t}$, $\mathbf{b_d}$, $\Theta_c$, $\Theta_s$, $\Theta_t$ with random values
Initialize $\Omega = \frac{1}{4}\mathbf{I}_4$, where $\mathbf{I}$ is an identity matrix
epoch = 0
while epoch ≤ MaxEpoch do
    $c_s$, $c_t$ = 0, 0
    while $c_s$ ≤ src_batches do
        read the $c_s$^th mini-batch from the source domain
        Update $\Theta_c$, $\Theta_s$, $\mathbf{W}$, $\mathbf{W_d}$, $\mathbf{b_s}$ and $\mathbf{b_d}$
        $c_s$ += 1
        if $c_t$ == tgt_batches then
            $c_t$ = 0
            Update $\Omega = \frac{(\mathbf{W}^T\mathbf{W})^{\frac{1}{2}}}{\mathbf{tr}\left((\mathbf{W}^T\mathbf{W})^{\frac{1}{2}}\right)}$
        end if
        read the $c_t$^th mini-batch from the target domain
        Update $\Theta_c$, $\Theta_t$, $\mathbf{W}$, $\mathbf{W_d}$, $\mathbf{b_t}$ and $\mathbf{b_d}$
        $c_t$ += 1
    end while
    epoch = epoch + 1
end while
```

2.6 Implementation Details

In our full transfer learning model, we initialize the lookup table $\mathbf{E}$ with the pre-trained vectors from *GloVe* [26] by setting l as 300. For **BCNN**, the window size and activation function are set to be 4 and ReLU, and the feature map sizes are set to be 50 and 100 for PI and NLI; for the two convolution layers of **Pyramid** in both PI and NLI, the feature map sizes are set to be 8 and 16, the strides are set to be 1 and 3, and the kernel sizes are set to be 6 × 6 and 4 × 4; for the two max-pooling layers of **Pyramid**, the strides are set to be 4 and 2, and the pooling sizes are set to be 4 × 4 and 2 × 2. Besides, for λ_0 and λ_1, we set them as 0.05 and 0.0008; while for λ_2, λ_3, λ_4 and λ_5, we set them as 0.0004. AdaGrad [10] is used to train our model with an initial learning rate of 0.08.

3 ONLINE SYSTEM

As introduced in Section 1, our online chatbot system is based on traditional information retrieval techniques, where the goal is to obtain the nearest question in the knowledge base for a given customer question [14]. Fig. 1 depicts the whole system architecture.

Specifically, we first build an indexing for all the questions in our knowledge base (KB) using *Apache Lucene* [5]. Next, given a query question, we employ TF-IDF ranking algorithm [39] in Lucene to compute its similarities to all the questions in the KB, and call back the top-K candidate questions. We then use a reranking algorithm to compute the similarities between the query and the K candidates, and obtain the most similar candidate. Finally we return the answer of the selected candidate to answer the query question. Note that in this paper, we only consider formulating our question rerank module as a PI task, but one can also model it as an NLI task.

Our existing reranking method is based on this ensemble method for the Answer Selection task [34]. But instead of using the output of the time-consuming LSTM model, we feed another three features, namely, Word Mover's Distance [16], keywords features [34] and the cosine distance of sentence embeddings [37] to a gradient boosted regression tree (GBDT).

To combine our model with the existing ranking method, we treat the probability of being *paraphrases* predicted by our model as an additional feature, and feed all features to GBDT for reranking.

4 EXPERIMENTS

In this section, we describe a qualitative evaluation of our proposed methods from the following perspectives: (1) From Section 4.1 to Section 4.4, we perform an **intrinsic** evaluation by utilizing a benchmark dataset and our own dataset to show the efficiency and effectiveness of our proposed base model and transfer learning framework; (2) In Section 4.5, we deploy our full model into our chatbot system, and conduct an **extrinsic** evaluation to show that

[5] https://lucene.apache.org/core/

our full model can bring in significant improvements to our existing online chatbots.

4.1 Experiment Settings

Datasets: In this section, we evaluate our methods on both Paraphrase Identification (PI) and Natural Language Inference (NLI).

For PI, we used a recently released large-scale dateset[6] by Quora as the source domain, and our E-commerce dataset as the target domain. Based on our historical data, we constructed a question answering KB, which consists of around 15,000 frequently asked QA pairs. To create labeled question pairs, we first collected all the query questions from the chat log of conversations between clients and our staff from May 22 to May 28, 2017. For each query question, we then used *Lucene* indexing to retrieve several of its similar questions, and obtained 45,075 question pairs. Finally, we asked a business analyst to annotate all the question pairs.

For NLI, we employed a large-scale multi-genre corpus [38], which contains an image captioning domain (SNLI) and another five distinct genres/domains about written and spoken English (MultiNLI)[7]. Since the number of sentence pairs in SNLI is much larger than that in the other five domains, we took SNLI as the source domain, and the others as the target domains.

Table 1 and Table 2 summarize the statistics of our datasets. Note that in Table 1, the number before and after the slash for Q-Q pairs denote respectively the total number of question pairs and the number of positive question pairs (i.e., paraphrases), while the two numbers for #Query-Q respectively denote the total number of query questions and the number of questions with paraphrasing candidates. Besides, for #Candi-Q, we refer to the average number of candidate questions for each query.

Compared Methods: For base models, we compared our hCNN model with the following models:

- **BCNN** is the left component of our hCNN model, which incorporates element-wise comparisons on top of the base model proposed in [43].
- **Pyramid** is the right component of our hCNN model based on sentence interactions as in [24].
- **ABCNN** is the attention-based CNN model by [43].
- **BiLSTM** is similar to **BCNN**, but uses LSTM instead of CNN to encode each sentence as in [3].
- **ESIM** is one of the state-of-the-art attention-based LSTM models on SNLI proposed by [6].
- **hCNN** is our hybrid CNN model as introduced in Section 2.4.

For evaluating the proposed transfer learning framework, we employed the following compared systems:

- **Tgt-Only** is the baseline trained in the target domain.
- **Src-Only** is another baseline trained in the source domain.
- **Mixed** is to simply combine the labeled data in the two domains to train the hCNN model.
- **Fine-Tune** is a widely used TL method, where we first train a model on the source data, and then use the learned parameters to initialize the model parameters for training another model on the target data.
- **FS** and **SS** are the fully-shared and specific-shared frameworks as detailed in Section 2.1.
- **DRSS** is our proposed model of learning domain relationships based on **SS** as in Section 2.2.
- **SS-Adv** and **DRSS-Adv** denote adding the adversarial loss into **SS** and **DRSS** as in Section 2.3.

All the methods in this paper are implemented with Tensorflow and are trained on machines with NVIDIA Tesla K40m GPU.

Evaluation Metrics: For PI, since our goal is to retrieve the most similar candidate for each query question, we use our model to predict each candidate's probability of being *paraphrase* as its similarity score, and then rank all the candidates. To evaluate the ranking performance, we use Precision@1, Recall@1, F_1@1 as metrics; to evaluate the classification performance for all question pairs, we employ two metrics: the Area under the Receiver Operating Characteristic curve (AUC) score [5] and the classification accuracy (ACC). For NLI, we only use ACC as the evaluation metric.

4.2 Comparisons Between Base Models

In Table 3, we compared different models for classifying sentence pairs with hCNN in both efficiency and effectiveness. Note that to fairly evaluate the efficiency of each model, we compute the total time of predicting all the test sentence pairs on CPU by setting the mini-batch size to 1, and report the average time. Also, for feature map sizes in BCNN, ABCNN and the BCNN component in hCNN, we set them as 50 for PI and 300 for NLI.

First, we can find that LSTM-based methods are generally much slower than CNN-based methods. Especially for ESIM, although it can outperform all CNN-based models, its computational time for each sentence pair is 32.2ms for our dataset and 79.5ms for SNLI, which is 6-11 times of CNN-based models. This means that most existing state-of-the-art models can only support low QPS, and therefore hard to be applied to industry. Second, clearly for both tasks, hCNN performs better than the other CNN-based methods, which indicates that BCNN and Pyramid are complementary to each other, and can work better when combined. Moreover, we verified that the improvements of hCNN over the other methods are significant with $p < 0.05$ based on McNemar's paired significance test [12]. Finally, while the computational cost of hCNN is slightly higher than BCNN and Pyramid, it can still serve 233 question pairs per second, which is able to satisfy the current demand of our industrial bot.

4.3 Comparisons Between TL Methods

We further evaluated the performance of our transfer learning method in Table 4 and Table 5.

We can observe from Table 5 that for all the five target domains, Src-Only perform much worse than Tgt-Only, and the average performance of Mixed is even worse than Tgt-Only. This implies that the source domain is quite different from all the target domains, and simply mixing the training data in two domains may lead to the model overfitting the source data since $|\mathcal{D}^s|$ is much larger

[6] https://www.kaggle.com/c/quora-question-pairs

[7] For the MultiNLI dataset, we use Version 0.9 in this paper. Note that since the label of the original test set is unavailable, we treat its development set as our test set, and randomly choose 2000 sentence pairs from its training set as our development set.

Table 1: Statistics of Paraphrase Identification Data

		Train	Dev	Test
AliExpress	Q-Q Pairs	29,884/8,624	7,622/1,968	7,569/2,133
	#Query-Q	3202/2414	781/578	777/584
	#Candi-Q	9.33	9.76	9.74
	#words /Query-Q	11.71	11.73	12.04
	#words /Candi-Q	8.37	8.37	8.54
Quora	Q-Q Pairs	404,290/149,265	N.A.	N.A.

Table 3: A Comparison Between Different Base Models

	E-Commerce			SNLI	
	AUC	ACC	Test Time(ms)	ACC	Test Time(ms)
BCNN	81.0	77.5	2.8	81.0	3.3
Pyramid	77.8	77.0	3.8	77.7	5.3
ABCNN	81.0	78.2	5.2	81.8	12.3
hCNN	82.2†	79.2†	4.3	83.2†	6.4
BiLSTM	79.9	77.8	7.1	80.6	19.6
ESIM	84.2	79.8	32.2	86.7	79.5

Table 2: Statistics of Natural Language Inference Data

	SNLI	Fiction	Travel	Slate	Telephone	Government
Train	550,125	77,348	77,350	77,306	83,348	77,350
Dev	10,000	2,000	2,000	2,000	2,000	2,000
Test	10,000	2,000	2,000	2,000	2,000	2,000

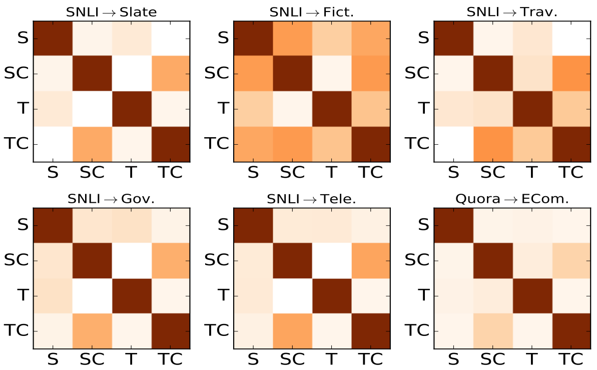

Figure 4: Learnt Correlation Matrix. A darker color means a larger entry value. S:Source, T:Target, SC:Source-shared, TC: Target-shared.

than $|\mathcal{D}^t|$. In addition, it is observed that the widely used Fine-Tune method can perform slightly better than Tgt-Only in most cases, which shows that pre-training the model parameters on a related source domain is better than randomly initializing them. Moreover, in all the five domains, the performance of two existing transfer learning frameworks FS and SS are both 1.9% better than that of Tgt-Only, which proves their usefulness. Furthermore, our proposed method DRSS improves the average performance of SS to 0.665, and the improvements are significant over all the tasks with $p < 0.05$ based on McNemar's paired significance test. This suggests that capturing the relationship between domains is generally useful for transfer learning. Finally, we can see that the incorporation of adversarial loss into SS and DRSS further boosts their performance, and DRSS-Adv can achieve the best accuracy across all the methods. Similar trends can be also observed for the PI task from Table 4. Interestingly, by comparing Table 3 and Table 4, we find that with the help of training data from the source domain, the performance of DRSS-Adv is even better than that of ESIM. These observations demonstrate the effectiveness of our transfer learning method.

Apart from the effectiveness, we also measure the efficiency of each method. Since the first five methods only use a single hCNN model for prediction, the computational time is the same as hCNN. As for SS, DRSS and their adversarial extensions, the computational time is 6.9ms, which is slightly longer than the other five methods but still much shorter than LSTM-based methods as in Table 3.

4.4 Domain Relationships

After obtaining the covariance matrix Ω for each source/target pair, we can derive their corresponding correlation matrices. For better comparison, here we show the square root of the correlation matrices for DRSS.

As shown in Fig. 4 that across all the six source/target pairs, $\mathbf{W_{sc}}$ and $\mathbf{W_{tc}}$ are positively related with each other. This is intuitive as the shared network is supposed to learn shared features between the source and the target domains, thus the learned $\mathbf{W_{sc}}$ and $\mathbf{W_{tc}}$ should be close to each other. This also shows the learned correlation matrix helps to capture the inter-domain relationship between $\mathbf{W_{sc}}$ and $\mathbf{W_{tc}}$.

In Fig.4, we can also see that for most source/target pairs except SNLI→Fict, the correlation between $\mathbf{W_s}$ and $\mathbf{W_{sc}}$ and that between $\mathbf{W_t}$ and $\mathbf{W_{tc}}$ learnt by our model are with small values. This indicates that in most cases, the shared feature space and the domain-specific feature space learnt by SS tend to be different from each other, and our model can help to reveal such intra-domain relationships.

Finally, to help us get a deeper insight on the helpfulness of the adversarial training, we perform comparisons on the correlation matrices learnt by DRSS and DRSS-Adv. We first show the result of SNLI→Fict. in Table 6. As we can see, for DRSS, the correlation between $\mathbf{W_s}$ (or $\mathbf{W_t}$) and $\mathbf{W_{sc}}$ (or $\mathbf{W_{tc}}$) is relatively large, while for DRSS-Adv, the correlation is relatively small. For the other subtasks, we find that the learnt matrices of DRSS-Adv are similar to those of DRSS, but we still observe that the intra-domain correlations of DRSS-Adv are generally smaller than those of DRSS. This shows that adding the adversarial loss can encourage the shared feature space to capture more domain-independent features, and further make the shared and domain-specific feature spaces more different. Therefore, the adversarial training can lead our model to better satisfy our assumption on the domain relationships, and finally improve the performance. All the above observations demonstrate that our model can capture the *inter-domain* and *intra-domain* relationship as mentioned in Section 2.2.

4.5 Extrinsic Evaluations

As mentioned in Section 3, for the online reranking algorithm, we propose to train GBDT by treating the prediction score of our DRSS model as another feature. To achieve this, we first took out

Table 4: The Result of Paraphrase Identification Task

	Prec@1	Rec@1	F1@1	ACC	AUC
Tgt-Only	0.717	0.551	0.623	0.792	0.822
Src-Only	0.619	0.368	0.461	0.719	0.686
Mixed	0.735	0.532	0.618	0.788	0.810
Fine-Tune	0.713	0.567	0.632	0.790	0.825
FS	0.734	0.595	0.657	0.797	0.831
SS	0.744	0.601	0.665	0.800	0.837
SS-Adv	0.743	0.603	0.666	0.808	0.842
DRSS	**0.757**	0.608	0.674†	**0.812**†	0.847
DRSS-Adv	0.753	**0.620**	**0.680**†	0.809	**0.849**

Table 5: The Classification Result of NLI Task

	Fict.	Trav.	Gov.	Tele.	Slate	AVG
Tgt-Only	0.647	0.658	0.692	0.644	0.579	0.644
Src-Only	0.520	0.516	0.540	0.520	0.488	0.517
Mixed	0.647	0.647	0.675	0.648	0.580	0.639
Fine-Tune	0.653	0.652	0.684	0.651	0.591	0.646
FS	0.662	0.671	0.704	0.657	0.588	0.656
SS	0.653	0.668	0.700	0.668	0.592	0.656
SS-Adv	0.666	0.666	0.701	0.664	0.597	0.659
DRSS	0.665	**0.674**†	0.706†	0.673†	0.605†	0.665
DRSS-Adv	**0.676**†	0.673	**0.707**†	**0.675**†	**0.607**†	**0.668**

Table 6: Correlation Matrices on SNLI→Fict.

	DRSS				DRSS-Adv			
	$\mathbf{W_s}$	$\mathbf{W_{sc}}$	$\mathbf{W_t}$	$\mathbf{W_{tc}}$	$\mathbf{W_s}$	$\mathbf{W_{sc}}$	$\mathbf{W_t}$	$\mathbf{W_{tc}}$
$\mathbf{W_s}$	1.000	0.242	0.101	0.205	1.000	0.090	0.055	0.062
$\mathbf{W_{sc}}$	0.242	1.000	0.008	0.247	0.090	1.000	0.024	0.221
$\mathbf{W_t}$	0.101	0.008	1.000	0.127	0.055	0.024	1.000	0.043
$\mathbf{W_{tc}}$	0.205	0.247	0.127	1.000	0.062	0.221	0.043	1.000

the prediction scores of our DRSS model on the validation set. Then, we combined them together the other features as introduced in Section 3, and trained GBDT on the validation set. The model performance on the test set is reported in Table 7. Note that the test time here denotes the average serving time (including the Response Time), which is different from the reported test time in Table 3.

As we can see from the offline test, the GBDT model with the feature derived from our DRSS model (referred to as GBDT-DRSS) is respectively 26.3% and 7.1% better than our existing online model (referred to as GBDT) and the GBDT model with the feature derived from hCNN (refered to as GBDT-hCNN) in F_1@1. Although adding our DRSS feature leads to more computational time, the total prediction time is 80.7ms for each query question (i.e., QPS of 12), which is acceptable for our chatbots.

For online serving, to accelerate the computation, we set the number of candidates returned by *Lucene* as 30, and bundle the 30 candidates into a mini-batch to feed into our model for prediction. For online evaluation, we randomly sampled 2750 questions, where 1317 questions are answered by GBDT and 1433 questions are answered by GBDT-DRSS. Then, we asked one business analyst to annotate if the nearest question returned by models expresses the same meaning as the query question, and compared their precision at top-1. As shown in Table 7, the Prec@1 of GBDT-DRSS is 18.8% higher than that of GBDT.

Table 7: The Performance of Online Serving

	Offline		Online Evaluation
	F_1@1	Time(ms per query)	Prec@1
GBDT	0.539	20.1	0.614
GBDT-hCNN	0.636	69.9	-
GBDT-DRSS	**0.681**	80.7	**0.729**

5 RELATED WORK

Paraphrase Identification and Natural Language Inference: Recent years have witnessed great successes of applying different neural networks, including Recursive Neural Networks (ReNN), Reccurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN), into Paraphrase Identification and Natural Language Inference [3, 30, 43]. Although all these models have been shown to significantly outperform the traditional methods without deep learning, most of them only focus on improving the performance of standard in-domain setting, and therefore require a large amount of labeled data to train a robust model. However, in practice, it will be time-consuming and costly to manually annotate much labeled data for each target domain we are interested in. Hence, in this paper, our focus is to apply an efficient and effective NN model into a transfer learning framework so that we can leverage the large amount of labeled data from a related source domain to train a robust model for a resource-poor target domain, which can benefit our chatbot-based question answering system.

Transfer Learning: Transfer learning (TL) has been extensively studied in the last decade [20]. Most existing studies for TL can be generally categorized into two groups. The first line of work assumes that we have enough labeled data from a source domain and also a little labeled data from a target domain [9], and the second line assumes that we only have labeled data from source domain but may also have some unlabeled data from a target domain [2, 45]. Our study belongs to the first line of work, which is also referred to as *supervised domain adaptation.*

For supervised domain adaptation, a majority of previous work belong to two clusters: instance-based and feature-based transfer learning. The former focuses on mining from the source labeled data to find those instances that are similar to the distribution of the target domain, and combine them together with the target labeled data [8, 15]. The core idea of the latter line of work is to find a shared feature space, which can reduce the divergence between the distribution of the source and the target domains [1, 17, 28, 33]. Our work follows the latter one, and tries to leverage NN models to learn a shared hidden representation for sentence pairs across domains.

Deep Transfer Learning: With the recent advances of deep learning, different NN-based TL frameworks have been proposed for image processing [44] and speech recognition [35] as well as NLP [19, 22, 41]. A simple but widely used framework is referred to as *fine-tuning* approaches, which first use the parameters of the well trained models on the source domain to initialize the model parameters of the target domain, and then fine tune the parameters based on labeled data in the target domain [22, 44]. Another line of work can be referred to as *multi-task feature learning* approaches, which bears the same intuition behind the feature-transfer methods as mentioned above. Among this line of work, one typical framework is to simply use a shared NN to learn a shared feature space [22, 41], while another representative framework is to employ a shared NN and two domain-specific NNs to respectively derive a shared feature space and two domain-specific feature space [19]. Motivated by the observation that both methods fail to consider the domain relationship, in this paper, we propose to jointly learn the shared feature representations and domain relationships in a unified model. Moreover, inspired by the recent success of applying adversarial networks into unsupervised domain adaptation [11, 31] and multi-task learning [19], we also incorporate the adversarial training into our transfer learning model in order to learn a more robust shared feature space across domains.

6 CONCLUSIONS

In this paper, we systematically evaluated different base methods and transfer learning techniques for modelling sentence pairs, with the goal of proposing an effective and efficient transfer learning framework for PI and NLI. Specifically, we first proposed a hybrid CNN model on the basis of two existing models, and then further proposed a general transfer learning framework, which can simultaneously perform the shared feature learning and domain relationship learning in an end-to-end mode. Evaluations on both a benchmark dataset and our own dataset showed that (1) our hybrid CNN model is both effective and efficient in comparison with several representative models; (2) our transfer learning framework can outperform all the existing frameworks across six source/target pairs. We further deployed our transfer learning model in our online chatbot system, and showed that it can improve the performance of the existing system by a large margin.

ACKNOWLEDGEMENTS

The authors would like to give great thanks to Feng Ji, Wei Zhou, Weipeng Zhao and Xu Hu for their helpfulness during the project, and the anonymous reviewers for their constructive comments.

REFERENCES

[1] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. 2007. Multi-task feature learning. In *NIPS*.

[2] John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain Adaptation with Structural Correspondence Learning. In *EMNLP*.

[3] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*.

[4] Samuel R. Bowman, Jon Gauthier, Abhinav Rastogi, Raghav Gupta, Christopher D. Manning, and Christopher Potts. 2016. A Fast Unified Model for Parsing and Sentence Understanding. In *ACL*.

[5] Andrew P Bradley. 1997. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern recognition* 30, 7 (1997), 1145–1159.

[6] Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced LSTM for Natural Language Inference. In *ACL*.

[7] Lei Cui, Shaohan Huang, Furu Wei, Chuanqi Tan, Chaoqun Duan, and Ming Zhou. 2017. SuperAgent: A Customer Service Chatbot for E-commerce Websites. In *ACL 2017, Demo*.

[8] Wenyuan Dai, Qiang Yang, Gui-Rong Xue, and Yong Yu. 2007. Boosting for transfer learning. In *ICML*.

[9] Hal Daume III. 2007. Frustratingly Easy Domain Adaptation. In *ACL*.

[10] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12, Jul (2011), 2121–2159.

[11] 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.

[12] Laurence Gillick and Stephen J Cox. 1989. Some statistical issues in the comparison of speech recognition algorithms. In *ICASSP*.

[13] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *NIPS*.

[14] Jiwoon Jeon, W Bruce Croft, and Joon Ho Lee. 2005. Finding similar questions in large question and answer archives. In *CIKM*.

[15] Jing Jiang and ChengXiang Zhai. 2007. Instance Weighting for Domain Adaptation in NLP. In *ACL*.

[16] Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. In *ICML*.

[17] Su-In Lee, Vassil Chatalbashev, David Vickrey, and Daphne Koller. 2007. Learning a meta-level prior for feature relevance from multiple related tasks. In *ICML*.

[18] Feng-Lin Li, Minghui Qiu, Haiqing Chen, Xiongwei Wang, Xing Gao, Jun Huang, Juwei Ren, Zhongzhou Zhao, Weipeng Zhao, Lei Wang, and Guwei Jin. 2017. AliMe Assist: An Intelligent Assistant for Creating an Innovative E-commerce Experience. In *CIKM 2017. Demo*.

[19] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial Multi-task Learning for Text Classification. In *ACL*.

[20] Mingsheng Long, Jianmin Wang, Guiguang Ding, Sinno Jialin Pan, and S Yu Philip. 2014. Adaptation regularization: A general framework for transfer learning. *IEEE Transactions on Knowledge and Data Engineering* 26, 5 (2014), 1076–1089.

[21] Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan, and Zhi Jin. 2016. Natural Language Inference by Tree-Based Convolution and Heuristic Matching. In *ACL*.

[22] Lili Mou, Zhao Meng, Rui Yan, Ge Li, Yan Xu, Lu Zhang, and Zhi Jin. 2016. How Transferable are Neural Networks in NLP Applications?. In *EMNLP*.

[23] Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering* 22, 10 (2010), 1345–1359.

[24] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text Matching as Image Recognition. In *AAAI*.

[25] Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A Decomposable Attention Model for Natural Language Inference. In *EMNLP*.

[26] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *EMNLP*.

[27] Minghui Qiu, Feng-Lin Li, Siyu Wang, Xing Gao, Yan Chen, Weipeng Zhao, Haiqing Chen, Jun Huang, and Wei Chu. 2017. AliMe Chat: A Sequence to Sequence and Rerank based Chatbot Engine. In *ACL*.

[28] Minghui Qiu, Peilin Zhao, Ke Zhang, Jun Huang, Xing Shi, Xiaoguang Wang, and Wei Chu. 2017. A Short-Term Rainfall Prediction Model using Multi-Task Convolutional Neural Networks.. In *ICDM*.

[29] Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, and Phil Blunsom. 2015. Reasoning about entailment with neural attention. *ICLR* (2015).

[30] 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*.

[31] Yaniv Taigman, Adam Polyak, and Lior Wolf. 2017. Unsupervised cross-domain image generation. *ICLR* (2017).

[32] Oriol Vinyals and Quoc Le. 2015. A neural conversational model. *arXiv preprint arXiv:1506.05869* (2015).

[33] Chang Wang and Sridhar Mahadevan. 2008. Manifold alignment using procrustes analysis. In *ICML*.

[34] Di Wang and Eric Nyberg. 2015. A Long Short-Term Memory Model for Answer Sentence Selection in Question Answering. In *ACL-IJCNLP*.

[35] Dong Wang and Thomas Fang Zheng. 2015. Transfer learning for speech and language processing. In *Signal and Information Processing Association Annual Summit and Conference (APSIPA)*.

[36] Shuohang Wang and Jing Jiang. 2017. A Compare-Aggregate Model for Matching Text Sequences. *ICLR* (2017).

[37] John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016. Towards universal paraphrastic sentence embeddings. *ICLR* (2016).

[38] Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. *arXiv preprint arXiv:1704.05426* (2017).

[39] Ho Chung Wu, Robert Wing Pong Luk, Kam Fai Wong, and Kui Lam Kwok. 2008. Interpreting tf-idf term weights as making relevance decisions. *ACM Transactions on Information Systems (TOIS)* 26, 3 (2008), 13.

[40] Rui Yan, Yiping Song, and Hua Wu. 2016. Learning to respond with deep neural networks for retrieval-based human-computer conversation system. In *SIGIR*.

[41] Zhilin Yang, Ruslan Salakhutdinov, and William W Cohen. 2017. Transfer learning for sequence tagging with hierarchical recurrent networks. *ICLR* (2017).

[42] Wenpeng Yin and Hinrich Schütze. 2015. Convolutional Neural Network for Paraphrase Identification.. In *NAACL-HLT*.

[43] Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. 2016. ABCNN: Attention-Based Convolutional Neural Network for Modeling Sentence Pairs. *Transactions of ACL* 4 (2016), 259–272.

[44] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks?. In *NIPS*.

[45] Jianfei Yu and Jing Jiang. 2016. Learning Sentence Embeddings with Auxiliary Tasks for Cross-Domain Sentiment Classification. In *EMNLP*.

[46] Yu Zhang and Dit-Yan Yeung. 2010. A convex formulation for learning task relationships in multi-task learning. In *UAI*.