IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 25, 2019, accepted February 23, 2019, date of publication March 5, 2019, date of current version March 26, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2901929

Adding Prior Knowledge in Hierarchical Attention Neural Network for Cross Domain Sentiment Classification

TU MANSHU AND WANG BING

Key Laboratory of Speech Acoustics and Content Understanding, Institute of Acoustics, Chinese Academy of Sciences, Beijing 65306, China
University of Chinese Academy of Sciences, Beijing 100190, China

Corresponding author: Tu Manshu (tumanshu_ucas@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 11590770-4, Grant 61650202, Grant 11722437, Grant U1536117, Grant 61671442, Grant 11674352, Grant 11504406, and Grant 61601453, in part by the National Key Research and Development Program under Grant 2016YFB0801203, Grant 2016YFC0800503, and Grant 2017YFB1002803, in part by the Key Science and Technology Project of the Xinjiang Uygur Autonomous Region under Grant 2016A03007-1, in part by the Foundation of Science and Technology on Information Assurance Laboratory under Grant KJ-17-102, and in part by the IACAS Young Elit Researcher Project under Grant Y854151431.

ABSTRACT Domain adaptation tasks have raised much attention in recent years, especially, the task of cross-domain sentiment classification (CDSC). Due to the domain discrepancy, a sentiment classifier trained in a source domain often performs less well, when directly applied to a target domain. Adversarial neural networks have been used in mainstream approaches for learning domain independent features, such as pivots, which are words with the same sentiment polarity in different domains. However, domain specific features can often determine sentiment in its context. In this paper, we propose a hierarchical attention network with prior knowledge information (HANP) for the CDSC task. Unlike other existing methods, the HANP can obtain both domain independent and domain specific features at the same time by adding prior knowledge. In addition, the HANP also includes a hierarchical representation layer with attention mechanism, so that the HANP can capture important words and sentences in relation to sentiment. Moreover, the proposed model can offer a direct visualization of the sentimental prior knowledge. The experiments on the Amazon review datasets demonstrate that the proposed HANP can significantly outperform the state-of-the-art methods.

INDEX TERMS Cross domain sentiment classification, HANP, prior knowledge.

I. INTRODUCTION

Sentiment classification is an important task in natural language processing (NLP). It can be used to identify user preferences, applied in the recommendation system or public opinion analysis [1]. Sentiment classification task aims to recognize sentiment polarity of documents (positive or negative). With the development of deep learning, many neural network-based sentiment analysis methods have achieved good results in public datasets such as IMDB movie review datasets [2], [3]. However, effective deep learning methods are heavily dependent on large labeled training data, which requires expensive manual labeling and is time consuming. In order to alleviate the dependence on large labeled data, cross-domain sentiment classification (CDSC) has become a promising direction. A CDSC task is to train a classification model with labeled data from a source domain, and then this model can be used to a classification task on a target domain.

Over the last decades, a number of methods have been proposed for CDSC. An appealing method is the Structural Correspondence Learning (SCL) method [4]. It uses pivot feature prediction tasks to introduce a projected feature space. A classifier obtained from this process works well for both source domains and target domains. This method is based on traditional discrete feature representations such as bag of words with linear classifiers. Along with the advances of deep neural networks in NLP, multi-layer neural network models such as Recurrent Neural Network (RNN) and Convolutional Neural Network (CNN) have achieved competitive performance in sentiment classification [2], [5]. Glorot [6] has demonstrated that a deep learning system based on Stacked Denoising Auto-Encoders(SDA) with sparse rectifier units can perform unsupervised features extraction. This mechanism pulls the

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

features of source domains and target domains close. It is highly beneficial for the domain adaptation of sentiment classifiers. Ganin *et al.* [7] proposed a Domain-Adversarial training of Neural Networks (DANN) for domain adaptation. DANN can extract a representation that preserves domain independent sentiment information, and ignore domain specific information. The disadvantage of DANN is lack of interpretability for directly identifying pivots. In order to improve the interpretability of deep models, Li *et al.* [8] proposed an end to end Adversarial Memory Network (AMN). AMN can automatically capture pivots. However, those non-pivot words that are only sentiment relevant in target domains often play an important role in CDSC. To take advantages of such a phenomenon, Li *et al.* [9] proposed a Hierarchical Attention Transfer Network (HATN) based on AMN. HTAN consist of two parts with one named P-net for pivots identification and the other named NP-net for aligning the non-pivots by using the pivots previously identified as a bridge. Nevertheless, given the processing structure of HATN, the pivots selection mechanism can propagate errors to the NP-net. Besides, HATN cannot distinguish dis-pivots. Dis-pivots are words that have different sentiment polarities in different domains. For example, the word ''sleepy'' has negative polarity in a book domain, but it often has positive polarity in a baby product domain.

N-grams, in general, have stable semantic and sentiment in different domains. Therefore, when training a classifier for CDSC, it is useful to incorporate such contextual information. CNN is often used to deal with contextual information [10], [11]. Since in CDSC the contextual information trained on source domains is applied to target domains for guiding sentiment classification. The question becomes which layers of a trained CNN should be used in target domains. In the field of image domain adaptation, Yosinski *et al.* [12] demonstrated that the features learned from the bottom of a deep CNN (DCNN) have better transferability compared with its middle and top layers [13]. Based on this study, Long *et al.* [14] proposed a Deep Adaptation Network (DAN), where DCNN is applied in the domain adaptation problems. Experimental results show that DAN constructed with bottom layers of a trained DCNN can perform better adaptation than that constructed without the pre-trained DCNN bottom layers. In this paper, we also consider this finding for solving CDSC problems.

To simultaneously harness the collective power of pivots, non-pivots and dis-pivots and incorporate contextual information, we propose a hierarchical attention network with prior knowledge information (HANP) for CDSC. The proposed HANP consists of three parts: a sentiment dictionary match (SDM) layer, a three-layer CNN, and a hierarchical attention network. The SDM layer can select important sentiment words from sentiment dictionary for further classification by establishing relations between words from a source domain and target domain. The three-layer CNN preserves contextual information from a source domain to target domain. At last, the hierarchical attention network is used to assign proper weights to sentiment related sentences and words for CDSC.

Our contributions are summarized as follows:

1) We propose to use a sentiment dictionary match layer to capture important sentiment words in context, so that all pivots, non-pivots and dis-pivots can be identified explicitly for HANP.
2) To preserve contextual information, a 3-layer CNN is incorporated into HANP.
3) We have conducted empirical experiments to test the performance of HANP on various datasets. Experimental results show that HANP can achieve state-of-the-art performance on CDSC tasks.

The rest of this paper is organized as follows: Section II presents related work. The details of HANP are given in Section III. Experimental results and discussion are presented in Section IV. We draw conclusions in Section V.

II. RELATED WORK

A. SENTIMENT CLASSIFICATION

Sentiment classification (SC) is a basic task in NLP. Last decades, many researchers have studied this task and proposed many practical functions. As the performance of text classifiers heavily relies on the extracted features, early work on SC mostly focuses on designing useful features from text content [15] and sentiment lexicons [16]. With the development of neural networks, most recent studies have explored the application of deep learning in sentiment classification to avoid manual feature extraction. CNN and its variants are widely used for semantic composition by automatically capturing local and global semantics [3], [10], [11]. Sequential model like RNN or long short-term memory (LSTM) are also verified as strong approaches for semantic composition [17]–[20]. The validity of the combination of CNN and LSTM is verified. The model C-LSTM utilizes a CNN to extract a sequence of higher-level phrase representations, and then feed them into LSTM to obtain the sentence representation [21]. It achieves comparable results on four sentiment classification datasets with much fewer parameters compared with multi-layer CNN. Wang proposes to use a regional CNN-LSTM model to predict the sentiment polarity of text [22]. By combining the regional CNN and LSTM, both local information within sentences and long distances dependency across sentences can be considered in the prediction process. In this paper, we also combine CNN and LSTM in our model.

B. PRIOR KNOWLEDGE

Prior knowledge is the knowledge that learners already have before they come across new information[1]. For sentiment classification tasks, incorporating prior knowledge into neural networks is an effective way to improve model performance. Melville *et al.* [23] developed an effective framework for incorporating lexical information in supervised learning

[1] https://dictionary.cambridge.org

for text categorization. It uses the lexical information about the polarity of each word w_i to obtain probability value of $P(w_i|c_j)$, where c_j is the class. The proposed method performs better than those using either background knowledge or supervised learning in isolation. When encoding documents into vectors, Term Frequency-Inverse Document Frequency (TF-IDF) is often used by traditional methods to represent the weight of each term in a document. Ko hypothesized that a weighting scheme with additional prior knowledge (class of this document) can achieve better performance [24]. He, therefore, proposed to use prior knowledge to replace IDF. His experiments demonstrated that this scheme outperformed KNN classifiers and SVM classifiers consistently on benchmark datasets. Yu used a sentiment dictionary as prior knowledge to label pivots. This sentiment dictionary assigns high scores to most positive and negative words. Including this prior knowledge makes the embedding process in his work more effective in CDSC tasks [25]. Huang *et al.* [26] encode syntactic knowledge (i.e. part-of-speech tags) in neural networks to enhance sentence representation. By evaluating the model on two sentiment classification tasks, they demonstrate that improvements can be obtained with such syntactic knowledge encoded.

C. HIERARCHICAL ATTENTION NEURAL NETWORK

Hierarchical attention neural networks have been demonstrated to perform better than word-level attention neural networks in various NLP tasks. In HAN, the first layer is designed to capture sentence level features and the second one is for document level features [2]. Our HANP is based on this structure. For CDSC task, li [9] propose a hierarchical attention transfer network (HATN) to capture non-pivots automatically. However, in this paper, we hypothesize that non-pivots, pivots and dis-pivots can all contribute to determining target domain sentiment classification. Therefore, in HANP, all these terms are considered.

III. HANP

In this section, we introduce the details of the HANP model for CDSC tasks. The problem definition and notations are first introduced. Then, we present an overview on the HANP model, followed by details of HANP components.

A. PROBLEM DEFINITION AND NOTATION

Let us denote a source domain with D_s and a target domain with D_t. The D_s consists of labeled data $D_s^l = \{d_s^i, y_s^i\}_{i=1}^{N_s^l}$ and unlabelled data $D_s^u = \{d_s^i\}_{i=1}^{N_s^u}$. N_s^l and N_s^u denote the number of labeled data and unlabelled data in source domain. d_s refers to a document in source domain, while y refers to d's corresponding label. So the source domain $D_s = D_s^l \cup D_s^u$. In the target domain D_t, there is a set of unlabelled data $D_t = \{d_t^i\}_{i=1}^{N_t^u}$, where N_t^u is the number of unlabelled data in target domain. The CDSC task is to design a robust classifier trained on D_s^l and apply it to predict the sentiment polarity of unlabelled data in D_t.

B. AN OVERVIEW OF THE HANP MODEL

The architecture of the HANP is shown in Figure 1. It consists of several parts: a word-level representation layer, a SDM layer, a 3-layer CNN, a document-level representation layer and the sentiment classifier layer. The input of HANP model contains two parts, one is a document d, the other is a sentiment dictionary sd, which is used for obtaining sentimental information and selected by some simple rules. The output of HANP is the sentiment polarity of d. The HAN part of HANP can capture important words and sentences based on the attention mechanism for sentiment classification. The function of word level representation and document-level representation are followed by [2]. The SDM layer and the 3-layer CNN layered in HANP are designed to provide compensations for cross domain problems. The SDM layer is used to capture the sentimental information, while the 3-layer CNN is used to obtain the contextual information. We describe the details of different components in the following sections.

1) WORD-LEVEL REPRESENTATION

This part includes two sub-parts, one is a word encoder with bidirectional LSTM (BLSTM), the other is word attention for extracting important words of a sentence. The output of this part is two new representations for each word of a sentence.

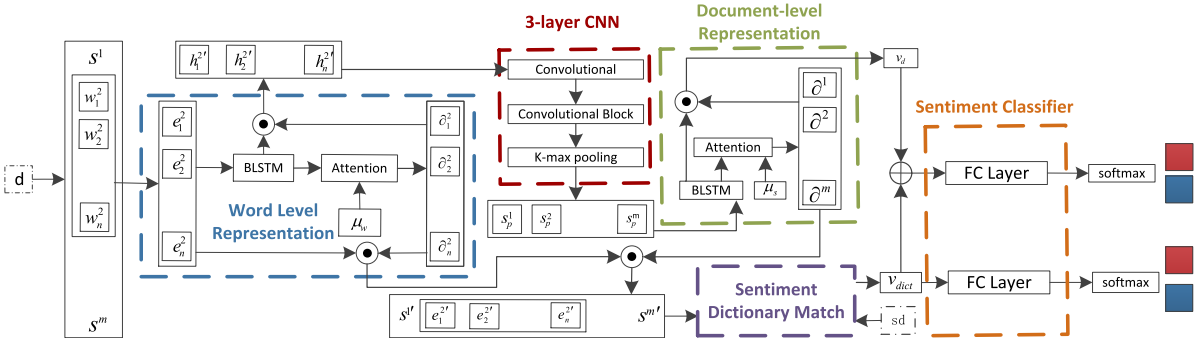

FIGURE 1. The framework of the HANP model. The dotted boxes of different colours represent different parts of the HANP. The *d* and *sd* using "-.-" around represent inputs for HANP.

Assume that a document d has m sentences $d = \{s^j\}_{j=1}^m$. Each sentence s^j has n words, $s^j = \{w_k^j\}_{k=1}^n$. We first map each word into its embedding vector e_k^j, $e_k^j = W_e w_k^j$, W_e is the embedding matrix. Each embedded word e_k^j is then converted into an annotation h_k^j by using a BLSTM. The BLSTM contains a forward LSTM $\overrightarrow{LSTM}$, which reads the sentence s^j from e_1^j to e_n^j and a backward LSTM $\overleftarrow{LSTM}$, which reads s^j from e_n^j to e_1^j:

$$\overrightarrow{h_k^j} = \overrightarrow{LSTM}(e_k^j) \quad k \in [1, n] \tag{1}$$

$$\overleftarrow{h_k^j} = \overleftarrow{LSTM}(e_k^j) \quad k \in [n, 1] \tag{2}$$

We eventually obtain the annotation h_k^j of w_k^j by concatenating the forward hidden state $\overrightarrow{h_k^j}$ and the backward hidden state $\overleftarrow{h_k^j}$, i.e., $h_k^j = [\overrightarrow{h_k^j}, \overleftarrow{h_k^j}]$.

Then the attention mechanism assigns weight α_k^j to each word in s^j. α_k^j is calculated as follows:

$$\mu_k^j = tanh\,(W_h h_k^j + b_h) \tag{3}$$

$$\eta_k^j = (\mu_k^j)^\top \mu_w \tag{4}$$

$$\alpha_k^j = \frac{exp(\eta_k^j)}{\sum_k exp(\eta_k^j)} \tag{5}$$

where μ_k^j is a hidden representation of h_k^j. μ_w is a word context vector. It is randomly initialized and jointly learned during later training process. The importance of word w_k^j is measured by a similarity η_k^j between μ_k^j and μ_w. The actual weight α_k^j of word w_k^j is obtained by normalizing η_k^j with a softmax function.

Then we generate two new representations for a word w_k^j. One is for dictionary match layer input, and the other is for the 3-layer CNN input. The former is calculated with the word importance weight α_k^j and the word embedding vector e_k^j as follows:

$$e_k^{j'} = \alpha_k^j e_k^j \tag{6}$$

The $e_k^{j'}$ contains original embedding information so that SMD layer can capture synonyms form sentiment dictionary, which is encoded by original embedding. The latter is calculated with the word importance weight α_k^j and the word annotation h_k^j as follows:

$$h_k^{j'} = \alpha_k^j h_k^j \tag{7}$$

We use $h_k^{j'}$ as the input of 3-layer CNN because it contains richer semantic information.

2) 3-LAYER CNN

Given the word representation $h_k^{j'}$, we can get a sentence vector s_p^j by using a 3-layer CNN. The 3-layer CNN contains a convolutional layer, a convolutional block layer and a *k_max_pooling* layer.

This structure is inspired by the success of very deep CNN (VDCNN) structure used in document classification [27], image classification [5] and transfer learning [14] [12]. CNN can learn n-gram information from source domains. For example, the phrase "not good" (negative words plus adjectives) has an inversion of sentiment polarity. CNN can preserve the polarity of such a phrase. We use the bottom layers (3-layer) of VDCNN to transfer contextual information from a source domain to a target domain, due to the fact the VDCNN has hierarchical nature when extracting features. That is, the generalized features, e.g. domain-independent features, are extracted at the bottom layers of the network. Features extracted at the middle and top layers of the VDCNN are domain private. The support of this conclusion is given in the appendix A.

The vector $h_k^{j'}$ can be seen as a new word embedding for word w_k^j. Then, a convolutional step is carried out to convert this new embedding to its corresponding feature maps. A convolution operation involves a filter ω, which is applied to a window of p words to produce a new feature. For example, a feature $c_{k:k+p-1}$ is generated from a window of words $h_{k:k+p-1}^{j'}$ ($h_{k:k+p-1}^{j'}$ means from $h_k^{j'}$ to $h_{k+p-1}^{j'}$) by

$$c_{k:k+p-1} = f(\omega \cdot h_{k:k+p-1}^{j'} + b) \tag{8}$$

where b is a bias term and f is a no-linear function, in this paper the f is ReLU activation. This filter is applied to each possible windows of new embedding in a sentence $\{h_{1:p}^{j'}, h_{2:2+p-1}^{j'}, \cdots, h_{n-p+1:n}^{j'}\}$ to produce a feature map $c = \{c_{1:p}, c_{2:p+1}, \cdots, c_{n-p+1:n}\}$. Assume we have λ feature maps, then we combine these feature maps to generate a feature matrix $fm_c^j \in \mathbb{R}^{(n-p+1)\times\lambda}$ as the input of convolutional block.

The convolutional block is a sequence of two convolutional layers, each one followed by a temporal BatchNorm layer [28]. There are λ feature maps in the convolutional block as well. The convolutional operation in the convolutional block is the same as before. It outputs a new feature matrix fm_{cb}^j. We then apply a *k_max_pooling* operation over each column of fm_{cb}^j to get a vector s_p^j. The final output of this 3-layer CNN is the vector s_p^j.

3) DOCUMENT-LEVEL REPRESENTATION

Given the sentence vectors s_p^j derived from the 3-layer CNN, we can get a document vector v_d' by a BLSTM and a sentence attention mechanism. First, we use the BLSTM to encode s_p^j into its annotation h^j. The encoding process is same as the function we mentioned in III-B1.

Similar to the word attention, we also use sentence attention mechanism to measure the importance of each sentence.

$$\mu^j = tanh(W_s h^j + b_s) \tag{9}$$

$$\eta^j = (\mu^j)^\top \mu_s \tag{10}$$

$$\alpha^j = \frac{exp((\eta^j)}{\sum_k exp(\eta^j)} \tag{11}$$

$$v_d = \sum_j \alpha^j h^j \tag{12}$$

where μ^j is a hidden representation for h^j and μ_s is a sentence level context vector. α^j is the importance weight of sentence annotation h^j. We compute the document representation v_d as a weighted sum of all the sentence annotations.

4) SENTIMENT DICTIONARY MATCH LAYER

The SDM layer aims to select important sentiment words from sentiment dictionary by establishing relations between non-pivots, dis-pivots from target domain and pivots, non-pivots from a source domain. Before we introduce SDM layer, let us first introduce the rules that we follow to select sentiment words. The sentiment dictionary *sd* consists of three parts: pivots (sd_{pivots}), dis-pivots (sd_{dis_pivots}) and non-pivots (sd_{non_pivots}), they are all important sentiment words for sentiment classification. All sentiment words in our sentiment dictionary are adjective or adverb.

We first introduce the rules for selecting pivots. These rules are inspired by paper [25]. The pivots are sentiment words that have the same polarities in all domains, e.g., *wonderful, awful*, and *excellent*. We use a sentiment dictionary SentiWordNet3.0 [29] to select pivots. The SentiWordNet3.0 has been widely used in opinion mining applications for selecting sentiment words. Each word in this dictionary contains its part-of-speech (POS), positivity score (PosS) and negativity score (NegS). If the PosS of a word is greater than a threshold *score_threshold* and its NegS equals to zero, this word is saved to an intermediate sentiment dictionary *msd*, *vice versa*. When the *msd* is created, we select the words that appear in all domains into sd_{pivots}. At the same time, according to the SentiWordNet3.0, we also create a Neutral Sentiment Dictionary *nsd* by selecting the words with their PosS and NegS both being zero.

For creating the dictionaries $sd_{non-pivots}$ and $sd_{dis-pivots}$, we first generate a candidate word set. A POS tagging is applied to all sentences in both domains. This is done with a NLTK toolkit[2]. The outcome of this process is a set of adjectives and adverbs with their frequencies at least 5 in each domain. Non-pivots are the words in this candidate set that only appear in a single domain. Such words are selected into the dictionary $sd_{non-pivots}$, if they do not appear in sd_{pivots} or *nsd*. Dis-pivots are the words in this candidate set that appear in both domains. Such words are selected into the dictionary $sd_{dis-pivots}$, if they do not appear in sd_{pivots} or *nsd*.

Suppose that the above created sentiment dictionary has L words, $sd = \{sw_l\}_{l=1}^{L}$. For the word embedding process of *sd*, we adopt three different strategies. The words in sd_{pivots} have consistent sentiment polarities across domains, therefore their embedding can use public word2vec.

In CDSC tasks, when applied to a target domain, a classifier processes words that are often unseen in its source domain, especially sentimental ones. Ideally, when these unseen sentimental words go through embedding, they are pulled close to their synonyms that the classifier might have been trained on from its source domain. As defined above, the sentimental words that only appear in one domain are words in $sd_{non-pivots}$. To create such connections, embeddings of $sd_{non-pivots}$ words are trained with corpus from both the source domain and the target domain.

For example, in a target domain, Kitchen, a word "elastic" is a non-pivot. A classifier is trained on the Electronics source domain, "elastic" is unseen by the classifier in its training process. However, after non-pivots embedding with corpus from both Kitchen and Electronics domains, connections between "rubber" and "elastic" are built. Thereafter, even though "elastic" is unseen to the classifier, it still can be used similarly to the meaning of "rubber".

Since dis-pivots have different sentimental polarities in a different domain, it is more reasonable to generate one embedding for each domain. Therefore, for each dis-pivot, two embeddings are created by corpus from the source domain and corpus from the target domain, respectively.

The detailed word2vec pre-training for sd_{pivots} $sd_{non-pivots}$ and $sd_{dis-pivots}$ are as follows:

1) Pivot embedding: We first initialized these words with the public 300-dimensional word2vec vectors [30]. They are fine-tuned during the training process.
2) Non-Pivot embedding: we pre-train them into 300-dimensional word2vec vectors by using both D_s and D_t. They are fine-tuned during the training process.
3) Dis-pivot embedding: we pre-train each dis-pivot into two 300-dimensional word2vec vectors. One is trained with D_s, and the other is trained with D_t. They are not fine-turned during the training process.

After word embedding for *sd*, we use this output as sentimental prior knowledge. In the next step, we aim to incorporate this sentimental prior knowledge into the document d.

By section III-B1, we have a representation of a word $e_k^{j'}$ in d. The sentence level importance weight α^j is then multiplied to $e_k^{j'}$, so that the word's representation is further weighted by the sentence it belongs to.

$$w_k^{j'} = \alpha^j e_k^{j'} \tag{13}$$

We then join each word in each sentence to form a new representation for document d: $d' = \{w_k^{j'}\}_{k=1}^{m\times n}$. Next, certain words from d' are selected out to form a new set $SM_{d'}$.

Each word sw_l in dictionary *sd* and each word in document d' are aligned into an adjacent matrix $CM_{d'}$. Each element v_{kl}^d in this adjacent matrix is a similarity between sw_l and $w_k^{j'}$. This similarity is calculated with cosine distance.

$$v_{kl}^d = \frac{w_k^{j'} e_{sw_l}}{\left\| w_k^{j'} \right\| \left\| e_{sw_l} \right\|} \tag{14}$$

[2]http://www.nltk.org/

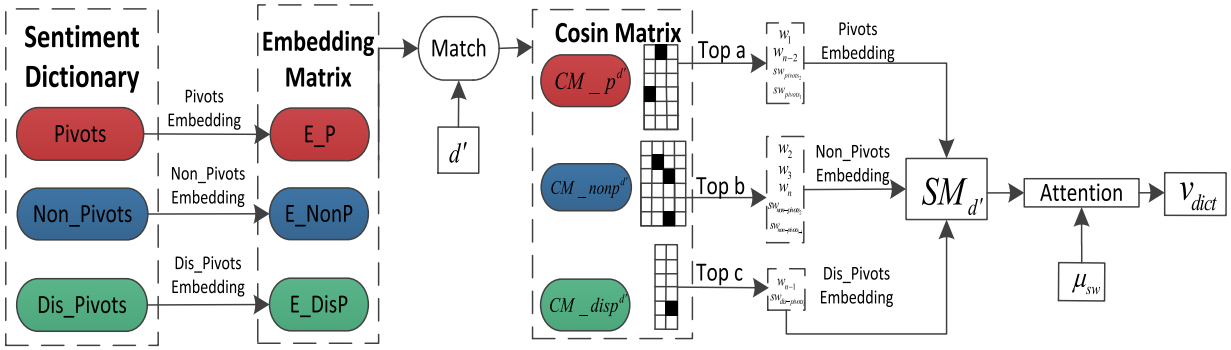

FIGURE 2. The framework of the SDM Layer. Different color represents different types of sentiment works.

$$CM_{d'} = \begin{bmatrix} v^d_{11} & \cdots & v^d_{1l} & \cdots & v^d_{1L} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ v^d_{k1} & \cdots & v^d_{kl} & \cdots & v^d_{kL} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ v^d_{m*n1} & \cdots & v^d_{m*nl} & \cdots & v^d_{m*nL} \end{bmatrix} \quad (15)$$

where the e_{sw_l} is an embedding operation on sw_l. In this matrix $CM_{d'}$, the subscript kl represents the k_{th} word in document d and l_{th} word in sd.

For each dictionary sd_{pivot}, $sd_{non-pivot}$ and $sd_{dis-pivot}$, we select a number of words from the document d and sd to form a new representation of d. Take sd_{pivot} as an example, and two hyper-parameters are set first. One is top_b, and the other is $threshold_b$. top_b decides the maximum number of words that we can select from d in accordance with sd_{pivot}, while $threshold_b$ determines which words can be selected from d in accordance with sd_{pivot}. If the similarity between a word $w^{j'}_k$ in d and a word in sd_{pivot} is higher than $threshold_b$, then the word $w^{j'}_k$ is selected into $SM_{d'}$. However, it is possible that the number of words in d that fit the above threshold rule is less than top_b. In this case, zero padding is used.

After the whole dictionary matching process, $SM_{d'}$ contains $top_a + top_b + top_c$ embedding vectors. top_a is the parameter for $sd_{non-pivot}$, top_b is the parameter for sd_{pivot}, and top_c is the parameter for $sd_{dis-pivot}$,

We use same function described in equation (5) to assign different weight α_{sw_p} for all sentiment words we picked out by using another word level context vector μ_{sw}.

$$v_{dict} = \sum_p \alpha_{sw_p} e^j_p \quad p \in [0, a+b+c] \quad (16)$$

The vector v_{dict} can be seen as a high level representation of important sentiment word in d.

5) SENTIMENT CLASSIFICATION

When we get the sentiment information representation v_{dict} and the document representation v_d, we concatenate these two vectors as the final representation for d, $v'_d = [v_d, v_{dict}]$.

For the final sentiment classification step, we use v_{dict} and v'_d as two inputs for two fully connected (FC) layers. Each FC layers contains three layers and is described as:

$$Output = Relu(w * Input + b) \quad (17)$$

Then we apply a softmax to the final output of the FC layer to obtain the prediction for each input document. Suppose that L_d denotes the loss function to minimize the cross-entropy of using all features v'_d for sentiment classification, while L_{dict} denotes the loss function using only sentimental features v_{dict} for sentiment classification.

$$L_d = -\frac{1}{S^l}\sum_{i=1}^{S^l}(y_i log\hat{y_d}^i + (1-y^i)log(1-\hat{y_d}^i)) \quad (18)$$

$$L_{dict} = -\frac{1}{S^l}\sum_{i=1}^{S^l}(y^i log^{\hat{y_{dict}}^i} + (1-y^i)log(1-\hat{y_{dict}}^i)) \quad (19)$$

where $y^i \in \{0, 1\}$ are the ground truth for the ith document. $\hat{y^i_d}$ and $\hat{y^i_{dict}}$ represent the sentiment classification result by using all features and only using sentiment features, respectively. We combine these two losses with a regularization to constitute the overall objective function:

$$L = L_d + L_{dict} + \rho L_{reg} \quad (20)$$

where ρ is a regularization parameter to balance the regularization term and other terms. The regularization term L_{reg} is responsible for avoiding over-fitting by placing the squared l_2 regularization to parameters of two sentiment classifiers. All the parameters are optimized jointly with the standard back-propagation algorithm.

IV. EXPERIMENT

In this section, we present the performance of the proposed HANP model on Amazon public datasets.

A. DATESETS

The experiments are conducted on the Amazon reviews datasets [31], which has been wildly used for CDSC tasks. These datasets contain reviews from five domains, and they are Books (B), DVD (D), Electronics (E), Kitchen (K) and Video (V). The details of the datasets are listed in the Table 1.

Table 1. Statistics of the Amazon reviews dataset including the number of training, valuation and unlabeled datasets for each domain. The last two columns represent the average number of words in each sentence and the average number of sentences in each review.

Domain	Train	Val	Unlab.	Avg_L	Avg_SN
Books	5600	400	9750	20.47	7.97
DVD	5600	400	11843	20.31	8.37
Electronics	5600	400	17009	16.59	6.44
Kitchen	5600	400	13856	15.49	6.04
Video	5600	400	30180	19.40	7.67

The ratio of positive and negative reviews are 1:1 in all domains for training data and valuation data.

By following li's work [9], we construct twenty CDSC tasks, for example, $B \rightarrow D$, where the dataset B before the arrow refers to the source domain and the dataset D refers to the target domain. The proposed model is trained on the source dataset and tested on the target dataset.

B. IMPLEMENTATION DETAILS

The embedding method for the sentiment dictionary consists of three parts. Details are given in III-B4.

The hyper-parameters of HANP are tuned on the validation set. The hyper-parameters in the SDM layer is listed in Table 2 last two columns. The *score_threshold* for selecting pivots are set to 0.75. We set the maximum number of words in one sentence to 20 and the maximum number of sentences in one document to 10 according to the average number of words and sentences in each domain in Table 1. The weights in HANP are randomly initialized by using a uniform distribution $U[-0.01, 0.01]$. The size of *LSTM* in word representation and document representation are both 100. The sizes of attention vector μ_w,μ_{sw} and μ_s are set to 50. The filter size in all convolutional operation is 3, and the numbers of filters for the convolutional layer and the convolutional block are 8 and 16. The pooling size is 8 for *k_max_pooling* in CNN.

Table 2. The second to fifth columns are the specific numbers of povits, non-povits and dis-pivots in *sd* for each domain. The last two columns record the max number of sentiment words in one document we select and the threshold value of each kind of *pivots*. For example, 5 is the value of top_c, 0.60 is the value of $threshold_c$.

	B	D	E	K	V	top_a top_b top_c	$threshold_a$ $threshold_b$ $threshold_c$
povits	870					20	0.70
non-povits	104	22	82	77	542	10	0.65
dis-pivots	75	231	60	48	306	5	0.60

For training, we use a mini-batch size of 50. The HANP is trained with the Adaptive Moment Estimation (Adam) with the learning rate set to 0.001 for all tasks. We perform early stopping on the validation set during the training process.

C. PERFORMANCE COMPARSION

The baseline methods in the comparison include:

1) **HAN** [2]: a hierarchical attention network for document classification. It progressively builds a document vector aggregating important words into a sentence and then aggregating important sentences vector to document vectors.
2) **CNN-aux** [25]: it is based on the CNN model proposed by Kim [10]. CNN-aux is a domain adaptation method for CDSC task based on sentence embeddings. The sentence embedding and sentiment classifier are jointly trained.
3) **AMN** [8]: it learns domain-shared representation based on memory networks and adversarial training. It can capture some pivots by using two shared memory networks with attention mechanism.
4) **HATN**^{h} [9]: it can capture some non-pivots based on AMN. It uses an "NP-net" to predict whether a sentence has positive pivots or negative pivots, and it contains a hierarchical positional encoding.
5) **HAN+CNN**: this model is the proposed HANP modified for ablation experiments. It adds the CNN structure into HAN to verify the effect of the contextual prior knowledge.
6) **HAN+CNN+Pivots**: this model adds the pivots (a part of sentiment words) on the basis of HAN+CNN to verify the effect of the prior knowledge of pivots.
7) **HAN+CNN+Pivots+Non-pivots**: this model is to confirm the effect of non-pivots information. It adds the non-pivots on the basis of HAN+CNN+Pivots.

Table 3 reports the classification accuracies of different methods on the Amazon reviews datasets. We evaluate our method on 20 transfer dataset pairs, and we use 5-fold cross-validation on each transfer dataset pair.

The proposed model HANP consistently achieves the best performance on all transfer pairs. Compared to the sentiment dictionary based approach CNN-aux, HANP outperforms CNN-aux by 5.78% on average. Our approach exceeds HATN^h by 1.10% on average in capturing pivots and non-pivots.

To validate the effectiveness of each part of the prior knowledge in HANP, we list the accuracy of adding each part to HAN.

First, we can see the HAN+CNN outperforms the HAN by 3.02% on accuracy. It shows that the CNN equipped with capturing n-gram information functionality affects on domain adaptation. It also means that the contextual information is important for the CDSC tasks. The HAN+CNN+Pivots outperforms HAN+CNN by 0.58% on accuracy. It has a slight improvement because the word level representation in HAN can pay higher attention to some pivots. Comparing HAN+CNN+Pivots+Non-pivots with HAN+CNN+Pivots, the former is 1.84% higher than the latter on accuracy. It shows that Non-pivots are important for CDSC tasks, and the sentiment dictionary that we have designed for incor-

Table 3. The accuracy of classification on the Amazon reviews dataset. * and ** indicates that our HANP method is significantly better than HATN^h with $p < 0.001$ and $p < 0.005$ based on T-test, respectively.

S	T	HAN	CNN-aux	AMN	HATN^h	HAN+CNN	HAN+CNN Pivots	HAN+CNN Pivots+Non-pivots	HANP
B	D	82.12	84.42	85.62	87.07	85.02	85.58	87.48	**88.12***
B	E	78.90	80.63	80.55	85.75	81.67	82.98	84.66	**85.81**
B	K	81.67	83.38	81.88	87.03	85.01	84.85	87.22	**88.91***
B	V	82.57	84.43	87.25	87.80	86.56	86.89	87.96	**89.21***
D	B	81.78	83.07	84.53	87.78	86.53	86.99	87.55	**89.18***
D	E	79.27	80.35	80.42	86.32	82.12	83.41	85.23	**86.87****
D	K	81.95	81.68	81.67	87.47	83.38	84.79	87.88	**88.54***
D	V	85.33	85.87	87.40	89.12	88.97	89.33	90.98	**91.25***
E	B	76.21	77.38	77.52	84.03	81.53	81.94	84.21	**85.67***
E	D	79.12	79.07	80.53	84.32	80.87	81.19	84.15	**85.29****
E	K	84.09	87.15	87.83	90.08	89.54	89.88	90.87	**91.08***
E	V	81.85	78.78	82.12	84.55	82.12	83.03	85.31	**85.96***
K	B	78.45	78.47	79.05	84.88	82.68	82.68	83.79	**85.04**
K	D	78.26	79.07	79.5	84.72	80.13	81.91	84.29	**86.47***
K	E	85.76	86.73	86.68	89.33	88.64	88.70	89.09	**90.43***
K	V	81.03	78.82	82.15	85.42	81.84	82.58	84.12	**85.93****
V	B	81.49	81.48	83.5	87.10	85.02	84.62	87.22	**88.94***
V	D	84.77	85.25	86.88	87.9	85.64	85.97	87.87	**88.54****
V	E	78.18	82.32	79.68	85.98	81.93	82.47	84.57	**86.11**
V	K	78.62	81.23	80.98	86.45	82.69	83.68	85.74	**87.21***
Average		81.07	81.98	82.79	86.66	84.09	84.67	86.51	**87.76**

porating prior knowledge is also practical. HANP obtains 1.25% increase, compared with HAN+CNN+Pivots+Non-pivots. It shows the validity and necessity of dis-pivots in CDSC tasks. In conclusion, each part of the prior knowledge in HANP has a positive impact on improving model accuracy.

To give a more profound analysis of the impact of sentimental prior knowledge, we show the relationship between accuracy and the number of pivots, non-pivots and dis-pivots in Figure 3. This figure shows the results under the best threshold values, $threshold_a = 0.70$ $threshold_b = 0.65$ $threshold_c = 0.60$.

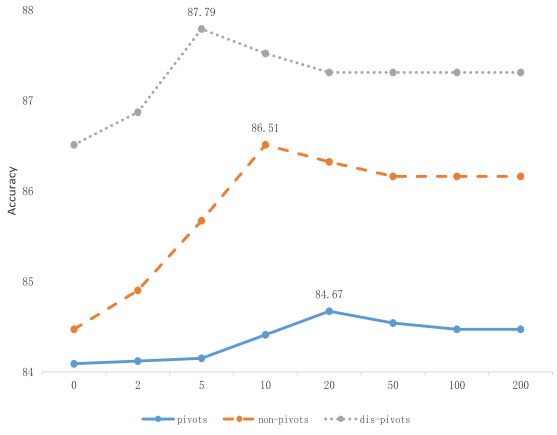

FIGURE 3. This line chart is about the accuracy of the different number of pivots, non-pivots and dis-pivots we select. The line of non-pivots is based on the best result of pivots, i.e., $top_a = 20$. The line of dis-pivots is based on the best result of non-pivots, i.e., $top_b = 10$.

As the number of sentiment words increases, the model performance on accuracy also increases. The highest accuracy value appears at $top_a = 20$, $top_b = 10$ and $top_c = 5$. This can be observed in Figure 3. After peaks appear, each line decreases a bit and then stays relatively stable. This shows that only a certain set of sentiment words are useful for the CDSC task. The decrease of accuracy after the peak values is not significant shows that more sentiment words to HANP do not bring too much noise to HANP.

D. CASE STUDY

To obtain a better understanding of our model, we conduct a case study where the source domain is Electronics and the target domain is Kitchen.

As shown in Figure 4, our HAN+CNN structure tends to pay more word attention to some sentiment words like *perfect, poor, disappointed*, and *flimsy*. However, HAN+CNN cannot pay enough word attention to the non-pivots like *stereo, flavourful* and *roomy*. The SDM layer can capture the non-pivots and their synonyms of non-pivots and pay more attention to some of them. Although HAN+CNN pay more word attention to dis-pivots like *flimsy*, it can not distinguish the different meanings of

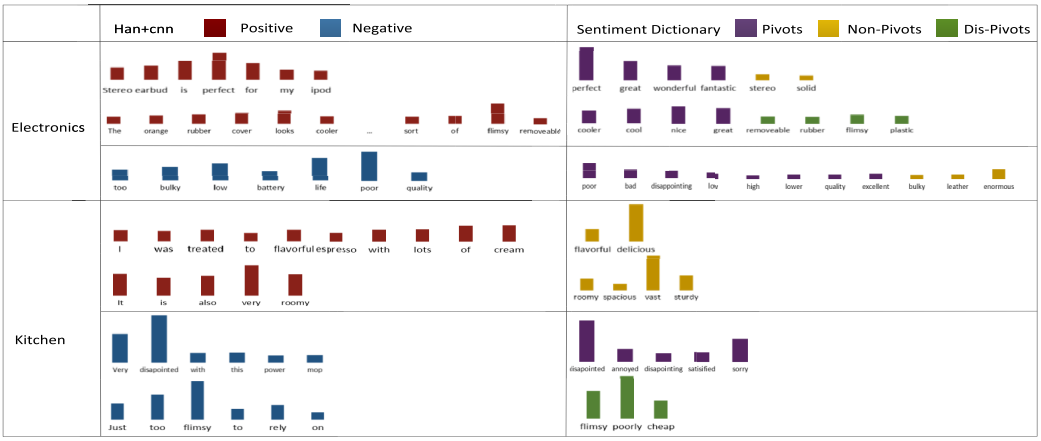

FIGURE 4. Visualization of attention on some examples of HANP in the $E \rightarrow K$ task is shown in this figure. The height of rectangles represents the attention value. Red rectangle denotes samples with positive sentiment polarity, and blue rectangle denotes samples with negative sentiment polarity. Purple, yellow and green rectangles represent the attention value of pivots, non-pivots and dis-pivots in SDM layer, respectively.

Table 4. Samples of pivots, dis-pivots and non-pivots we select in sentiment dictionary. The third column and last column are the synonyms for words in the second and fourth columns.

	Electronics Domain		Kitchen Domain	
Pivots	love good best reliable wonderful kind enjoy like vivid pleased charming happiness lucky nice woefully bad stubborn hard unacceptable worst hate terrible worse sadly silly horribly			
Non pivots	snug	comfortable securely tight	freshly	fresh minted smelling
	stereo	surround sound audio	elastic	fitted rubber snug sliding
Dis pivots	solidly	designed excellent like	solidly	well-made stylish built
	flimsy	plastic outer	flimsy	break poorly cheap
	portable	affordable walkman	portable	stylish pros compact

dis-pivots in different domains. The SDM layer selects different synonyms for the same word *flimsy* in different domains.

As shown in Table 4, we list some examples of pivots, non-pivots and dis-pivots and their synonyms found by our word2vec vectors. For non-pivots, although our model has not seen them in the source domain, it can learn the meaning of these words by finding out their synonyms. Take the word *snug* as an example, and it is a positive word according to its synonyms *comfortable*. Take the word *flimsy* as an example, it is a neutral word and a negative word in domain E and domain K, respectively. This shows that HANP can distinguish the different sentiment polarities of the same word in different domains.

V. CONCLUSION

In this paper, we propose the HANP method for CDSC task. The proposed HANP can pay more attention to important words and sentences for sentiment classification. It can also capture important pivots, non-pivots and dis-pivots by sentiment dictionary match layer. The HANP can obtain the meaning of a non-pivot or a dis-pivot by finding out its synonyms. Experiments on the Amazon review dataset show the effectiveness of HANP. We obtain the state-of-the-art accuracy on this dataset.

APPENDIX A
TRANSFER EXPERIMENTS

In this appendix, we describe the experiment and conclusions about transferability of model TVDCNN in details.

We use VDCNN as our base model. According to [27], nine layers network structure is suitable for transfer experiments. So we conduct transfer experiments based on nine layers VDCNN structure, we abbreviate as TVDCNN.

As shown in Fig. 5, **A** and **B** represent two different datasets. **A** represents the source domain and **B** represents the target domain. Label **A** and **B** represent the classification results for the dataset **A** and **B**, respectively.

1) BaseA & BaseB: The BaseA and BaseB networks are trained with standard backward propagation on the source domain **A** and the target domain **B**. (Fig. 5, first two rows)
2) A2B: the weight of first two layers are copied from baseA and frozen. The rest 7 higher layers (3-9 layer) are randomly initialized and trained on dataset **B**. Intuitively, we use the first 2 layers from the network trained on dataset **A** and then studied higher layer features on top of them to classify a new target dataset **B**. If the A2B has the same performance on dataset **B**, there is an evidence that the second-layer features are general. If the performance decrease, there is an

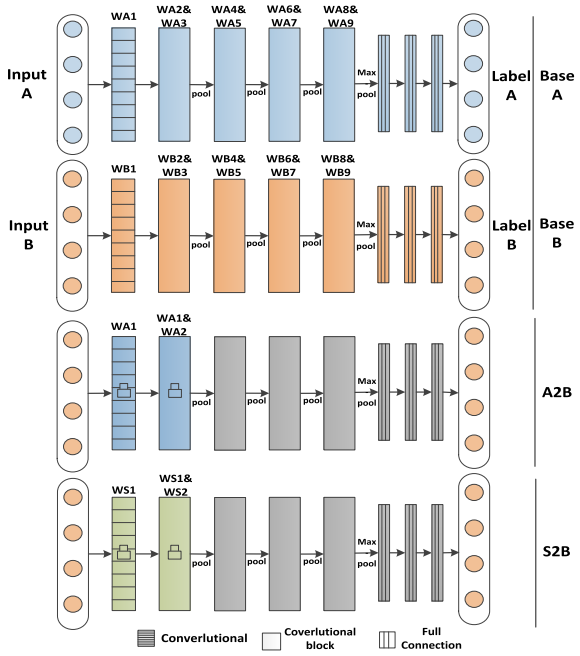

FIGURE 5. The framework of TVDCNN_n. *n* represents which layers are transferred in TVDCNN. In this figure $n = 2$, i.e., we transfer first two layers from source domain to target domain. The TVDCNN consist of four networks. All of them contain one converlutional layer, four converlutional block layers and four *k_max_pooling* layers. These layers have the same setting with section III-B2. The labelled rectangles (e.g. W_{A2}&W_{A3}) represent the weight vector learned from which layer and the colour of rectangles indicates which dataset the layer was originally trained on. The closed lock represents this layer is not involved in training.

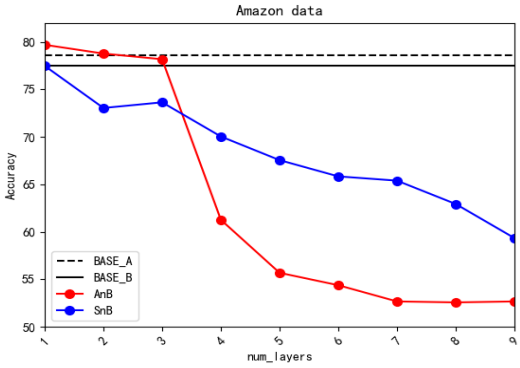

FIGURE 6. The results of this transferability experiment. Each point in this figure represents the average accuracy over the test set. The number on abscissa is the number of layers for transferring. The line with red circles is the performance of AnB network, while the line with blue triangles is SnB. The solid black line presents the accuracy of BaseB, and the dashed black line presents the accuracy of BaseA.

evidence that the second-layer features are specific to **A**. (Fig. 5, row 3)

3) A stochastic network S2B: the first 2 layers are stochastic and frozen. The rest 7 higher layers (3-9) are randomly initialized and trained on dataset **B**. If the S2B perform as well as BaseB, there is an evidence the second-layer has no contribution to the classifier. If performance suffers, it means the second-layer has contribution to the classifier. (Fig. 5, row 4)

We repeated this process for all n, $n \in [1, 2, \cdots, 9]$.

We use Amazon datasets as experimental data and to classify this data into two classes, positive or negative. The Amazon dataset consist of comments on different products. We random divided these different products into two different parts, i.e., part **A** and part **B**. This operation is repeated ten times.

A. EXPERIMENTAL RESULTS AND DISCUSSION

The results of all A/B learning experiments on randomly split datasets are shown in Fig. 6. The results yield many differences conclusions. In each of the following interpretations, we compare the performance from different perspectives.

SnB sets to verify whether the weights, which are learned from BaseA network are really work on dataset **B**. As shown in Fig 6, at the first 3 layers, the accuracy of SnB is lower than AnB. It means at least the features be obtained through first 3 layers are valid for classification.

AnB shows the transferability of the weights from BaseA at each layer. According to the result, the accuracy of A3B, A2B, and A1B is higher than BaseB 2% to 3%. It means the weights on first three layers can obtain general features. Layer four to layer nine show a more significant drop in performance, it means the features get from these layers are not general at all.

Therefore, according to the above conclusions, we use 3-layer CNN to capture contextual information and transfer these knowledges form source domain to target domain in HANP.

REFERENCES

[1] B. Liu and L. Zhang, *A Survey of Opinion Mining and Sentiment Analysis*. New York, NY, USA: Springer, 2012.

[2] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2016, pp. 1480–1489.

[3] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 649–657.

[4] J. Blitzer, R. McDonald, and F. Pereira, "Domain adaptation with structural correspondence learning," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Association for Computational Linguistics, 2006, pp. 120–128.

[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Represent.*, 2014. [Online]. Available: https://arxiv.org/abs/1409.1556

[6] X. Glorot, A. Bordes, and Y. Bengio, "Domain adaptation for large-scale sentiment classification: A deep learning approach," in *Proc. 28th Int. Conf. Mach. Learn. (ICML)*, 2011, pp. 513–520.

[7] Y. Ganin *et al.*, "Domain-adversarial training of neural networks," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 2096–2030, 2016.

[8] Z. Li, Y. Zhang, Y. Wei, Y. Wu, and Q. Yang, "End-to-end adversarial memory network for cross-domain sentiment classification," in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, 2017, pp. 2237–2243.

[9] Z. Li, Y. Wei, Y. Zhang, and Q. Yang, "Hierarchical attention transfer network for cross-domain sentiment classification," in *Proc. 32nd AAAI Conf. Artif. Intell., AAAI*, New Orleans, LA, USA, Feb. 2018, pp. 1–8.

[10] Y. Kim. (2014). "Convolutional neural networks for sentence classification." [Online]. Available: https://arxiv.org/abs/1408.5882

[11] W. Yin and H. Schütze. (2016). "Multichannel variable-size convolution for sentence classification." [Online]. Available: https://arxiv.org/abs/1603.04513

[12] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27. 2014, pp. 3320–3328.
[13] Y. Jia *et al.*, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. 22nd ACM Int. Conf. Multimedia*, 2014, pp. 675–678.
[14] M. Long, Y. Cao, J. Wang, and M. I. Jordan. (2015). "Learning transferable features with deep adaptation networks." [Online]. Available: https://arxiv.org/abs/1502.02791
[15] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up?: Sentiment classification using machine learning techniques," in *Proc. ACL-02 Conf. Empirical Methods Natural Lang. Process.-Volume 10*. Association for Computational Linguistics, 2002, pp. 79–86.
[16] K. Cheng, J. Li, J. Tang, and H. Liu, "Unsupervised sentiment analysis with signed social networks," in *Proc. AAAI*, 2017, pp. 1–7.
[17] D. Tang, B. Qin, and T. Liu, "Document modeling with gated recurrent neural network for sentiment classification," in *Proc. 2015 Conf. Empirical Methods Natural Lang. Process.*, 2015, pp. 1422–1432.
[18] J. Li, M.-T. Luong, D. Jurafsky, and E. Hovy. (2015). "When are tree structures necessary for deep learning of representations?" [Online]. Available: https://arxiv.org/abs/1503.00185
[19] X. Wang, Y. Liu, S. U. N. Chengjie, B. Wang, and X. Wang, "Predicting polarities of tweets by composing word embeddings with long short-term memory," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics 7th Int. Joint Conf. Natural Lang. Process. (Volume 1: Long Papers)*, vol. 1, 2015, pp. 1343–1353.
[20] M. Al-Smadi, B. Talafha, M. Al-Ayyoub, and Y. Jararweh, "Using long short-term memory deep neural networks for aspect-based sentiment analysis of arabic reviews," *Int. J. Mach. Learn. Cybern.*, pp. 1–13, 2018.
[21] Y. Xiao and K. Cho. (2016). "Efficient character-level document classification by combining convolution and recurrent layers." [Online]. Available: https://arxiv.org/abs/1602.00367
[22] J. Wang, L.-C. Yu, K. R. Lai, and X. Zhang, "Dimensional sentiment analysis using a regional cnn-lstm model," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics (Volume 2: Short Papers)*, vol. 2, 2016, pp. 225–230.
[23] P. Melville, W. Gryc, and R. D. Lawrence, "Sentiment analysis of blogs by combining lexical knowledge with text classification," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 1275–1284.
[24] Y. Ko, "A study of term weighting schemes using class information for text classification," in *Proc. 35th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2012, pp. 1029–1030.
[25] J. Yu and J. Jiang, "Learning sentence embeddings with auxiliary tasks for cross-domain sentiment classification," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 236–246.
[26] M. Huang, Q. Qian, and X. Zhu, "Encoding syntactic knowledge in neural networks for sentiment classification," *ACM Trans. Inf. Syst.*, vol. 35, no. 3, p. 26, 2017.
[27] A. Conneau, H. Schwenk, L. Barrault, and Y. Lecun. (2016). "Very deep convolutional networks for text classification." [Online]. Available: https://arxiv.org/abs/1606.01781
[28] S. Ioffe and C. Szegedy. (2015). "Batch normalization: Accelerating deep network training by reducing internal covariate shift." [Online]. Available: https://arxiv.org/abs/1502.03167
[29] S. Baccianella, A. Esuli, and F. Sebastiani, "Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining," in *Lrec*, vol. 10, no. 2010, 2010, pp. 2200–2204.
[30] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.
[31] J. Blitzer, M. Dredze, and F. Pereira, "Biographies, bollywood, boomboxes and blenders: Domain adaptation for sentiment classification," in *Proc. 45th Annu. Meeting Assoc. Comput. Linguistics*, 2007, pp. 440–447.

Authors' photographs and biographies not available at the time of publication.