2018 IEEE International Conference on Data Mining

Exploiting Topic-based Adversarial Neural Network for Cross-domain Keyphrase Extraction

Yanan Wang, Qi Liu, Chuan Qin, Tong Xu, Yijun Wang, Enhong Chen* and Hui Xiong*
Anhui Province Key Laboratory of Big Data Analysis and Application, School of Computer Science and Technology
University of Science and Technology of China
{ynwwang,wyjun}@mail.ustc.edu.cn
{qiliuql,tongxu,cheneh}@ustc.edu.cn,{chuanqin0426,xionghui}@gmail.com

***Abstract*—Keyphrases have been widely used in large document collections for providing a concise summary of document content. While significant efforts have been made on the task of automatic keyphrase extraction, existing methods have challenges in training a robust supervised model when there are insufficient labeled data in the resource-poor domains. To this end, in this paper, we propose a novel Topic-based Adversarial Neural Network (TANN) method, which aims at exploiting the unlabeled data in the target domain and the data in the resource-rich source domain. Specifically, we first explicitly incorporate the global topic information into the document representation using a topic correlation layer. Then, domain-invariant features are learned to allow the efficient transfer from the source domain to the target by utilizing adversarial training on the topic-based representation. Meanwhile, to balance the adversarial training and preserve the domain-private features in the target domain, we reconstruct the target data from both forward and backward directions. Finally, based on the learned features, keyphrase are extracted using a tagging method. Experiments on two real-world cross-domain scenarios demonstrate that our method can significantly improve the performance of keyphrase extraction on unlabeled or insufficiently labeled target domain.**

***Index Terms*—Adversarial Network; Transfer Learning; Keyphrase Extraction**

I. Introduction

Keyphrase extraction is the task of automatically extracting a set of phrases that provide a concise summary of a document content. This task is important for many text mining applications, such as text categorization [1], recommendation [2] and opinion mining [3].

In the literature, many efforts based on supervised learning techniques have been made on the automatic keyphrase extraction task. In other words, the supervised methods typically treat keyphrase extraction as a classification problem [4], [5], where given phrases are classified as keyphrases or non-keyphrases. Although supervised methods perform well in this task, it requires a large amount of labeled data which is extremely expensive and time-consuming to collect in many application scenarios [6]. Meanwhile, if there are insufficient labeled data, unsupervised keyphrase extraction methods have been proposed, such as graph-based ranking [6]–[8], clustering [9], and language modeling [10]. Although unsupervised keyphrase extraction methods avoid the need for expensive labeled data, their performance cannot compare with supervised methods, which were developed based on a sufficient amount of labeled data [11], [12].

In real-world scenarios, when the labeled data is insufficient, there may be a large number of labeled data in the related resource-rich domains, and we can utilize such knowledge. Taking the keyphrase extraction in research paper as an example, labeled data are distributed unevenly across different domains. There are few papers with author-assigned keyphrases in the domain of Computer Graphs (CG) while there are a large number of papers with author-assigned keyphrases in Data Mining (DM) domain. In this scenario, we can leverage a large amount of labeled data in DM domain (i.e., source domain) to help keyphrase extraction in CG domain (i.e., target domain). This is the idea of transfer learning which can utilize the knowledge learned in the other related domains to improve the performance of our target task. Indeed, transfer learning has achieved success in many other domains, such as text classification [13], recommendation [14].

However, to improve the performance of keyphrase extraction task in the unlabeled or insufficiently labeled target domain by transferring knowledge from resource-rich source domain, there are still three major challenges to be solved. 1) Intuitively those words which could summarize the given documents (i.e., covering the major topics [15]) should be selected as keyphrases. Thus, how to integrate major topics when extracting features? 2) There is a distribution mismatch between the two domains as the domain knowledge and user behavior of the two domains are quite different. Due to the domain distribution mismatch, the traditional method which is trained directly on the resource-rich source domain may perform poorly in the target domain. The reason is that supervised methods show *bias* towards the domain on which they are trained and may not generalize well to new domains [16]. Thus, when leveraging resource-rich labeled data to improve the performance in target domain, how to deal with the domain distribution mismatch? 3) As keyphrases are also closely related to the domain-private knowledge, a good representation for the target domain keyphrase extraction task should contain not only the domain-invariant features but also the domain-private features. Therefore, how to learn the target domain-private features based on the target unlabeled data?

To tackle the challenges above, we investigate the problem of keyphrase extraction in a cross-domain perspective, and

* denotes the corresponding author.

Label Sequence:	O	B	E	O	O	O	O	S	O	O	O	···
Text Sequence:	effective	test	generation	and	adequacy	assessment	for	javascript	based	web	applications	···

Fig. 1. An example of extracted keyphrases using sequence labeling method. The colored phrases are extracted keyphrases and corresponding labels whose meanings are detailed in Section III.

propose a novel Topic-based Adversarial Neural Network (TANN) for keyphrase extraction in the unlabeled target domain. It makes use of not only labeled data in the related resource-rich domain, but also the unlabeled data in both source and target domain. Specifically, to explicitly incorporate the global topic information into the feature representation, we first design a shared encoder in both domains with topic correlation mechanism which gives more attention to the topic-related words. Then, although there is a distribution mismatch between two domains, there are some common words like "*we introduce/we propose*" across domains, which are good indicators for the following keyphrases. To efficiently bridge the two domains with domain-invariant features (i.e., these common words), inspired by Ganin et al. [17], we propose to use adversarial training to ensure that the extracted features by the encoder are invariant to the change of domains. To prevent adversarial learning from eliminating all the domain-private information and only learning the domain-invariant features, we reconstruct the text sequence from both forward and backward directions to preserve the rich context and semantic information in target domain. Besides, our method learns a shared keyphrase tagger in two domains using sequence labeling methods [18]. Finally, experimental results on two real-world cross-domain keyphrase extraction tasks demonstrate that the proposed model is especially useful when the target domain is unlabeled or only limited annotated data is available. Our contributions can be summarized as follows:

- We investigate an under-explored problem of cross-domain keyphrase extraction. We show that it is possible to use both labeled data from resource-rich domains and unlabeled data in the source and target domains for improving the performance of keyphrase extraction in the unlabeled target domain.
- We propose a novel topic-based adversarial neural network that can learn transferable knowledge across domains efficiently by performing adversarial training. To the best of our knowledge, we are the first to exploit the adversarial learning technique for keyphrase extraction.
- We design a topic correlation layer to incorporate the topic-based representation of the document. Moreover, we also propose to reconstruct the document in the target domain from both forward and backward directions to learn the domain-private features.

II. Related Work

In this section, we briefly review two classes of closely related works: *keyphrase extraction* and *adversarial networks*.

A. Keyphrase Extraction

Various approaches to keyphrase extraction have been proposed mainly along two lines: supervised and unsupervised ones [19].

In the supervised approaches, keyphrase extraction is treated as a classification problem. A classifier is trained to discriminate keyphrase and non-keyphrase. Different machine learning methods have been used, such as Naïve Bayes [4], Condition Random Fields (CRF) [18] and Sequence Pattern Mining to search keyphrase candidates [20]. The features used in supervised methods are mainly Term Frequency-Inverse Document Frequency (TF-IDF), the first occurrence of the word, and part-of-speech tag of phrase [21], [22]. A main drawback is that these supervised methods mainly use statistical features, which are unable to capture the deep semantic information in the text. Recently, with the advanced performance of deep learning [23], [24], Deep Neural Networks (DNN) based models are used to solve keyphrase extraction problem. Zhang et al. [25] proposed a neural tagging method named joint-layer recurrent neural network to perform keyphrase extraction tasks. Meng et al. [12] generated the keyphrases directly from text with an encoder-decoder framework. Compared with traditional supervised methods with statistical features, these deep neural network based methods can grasp the deep semantic information in the text. Our work is different from these works since they are totally supervised, requiring a large amount of labeled data to train the model, while our method leverages both labeled data in the resource-rich domain and unlabeled data in the target domain. Furthermore, the above two deep neural network based methods did not consider the document topic information explicitly while we design a topic correlation mechanism to incorporate the topic information.

In the unsupervised approaches, keyphrase extraction is formulated as a ranking problem using different techniques, including graph-based ranking [7], [8], [15], [16], [26], clustering [9], and language modeling [10]. Compared with these totally unsupervised methods, our TANN method can also leverage the source labeled data.

B. Adversarial Networks

Adversarial networks were originally proposed for image generation [27], [28]. Concretely, Generative Adversarial Networks(GAN) incorporates two components: one generator and one discriminator. Generator is trained to generate real-like images while discriminator tries to discriminate the generated images from the truth.

Recently, adversarial networks have also gained success on domain adaptation [17], [29]. Specifically, Ganin et al. [17] applied adversarial training to domain adaptation and the work was built on the theory of domain adaptation [30]. The key

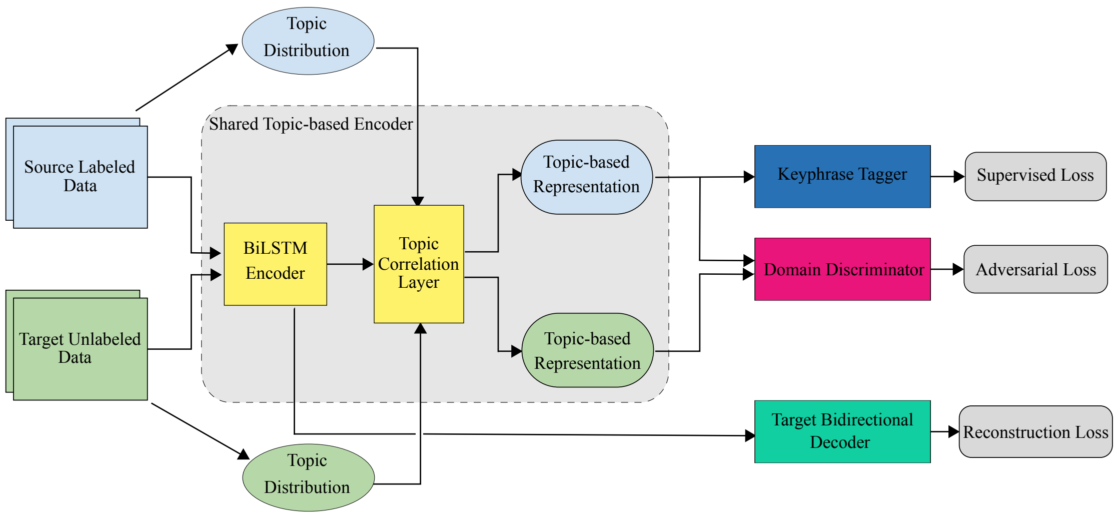


Fig. 2. Overall architecture of the TANN model for keyphrase extraction. The model is composed of four components: one shared Topic-based Encoder, Keyphrase Tagger, Domain Discriminator, and Target Bidirectional Decoder. The shared Topic-based Encoder contains one shared BiLSTM Encoder and one shared Topic Correlation layer.

idea is that a good representation for domain adaptation is the one that an algorithm cannot discriminate the origin of its domain. Adversarial training is helpful to learn transferable knowledge across domains efficiently and to address the domain distribution mismatch problem in cross-domain keyphrase extraction task. To the best of our knowledge, we are the first to exploit the adversarial learning technique for keyphrase extraction.

III. Problem Formulation

In this paper, we aim to extract keyphrase in a specific domain without labeled data, while there is a sufficient number of labeled data from related resource-rich domains. Specifically, we are given a set of labeled samples $\mathbf{X}_s^l = \{(\mathbf{x}_i^s, \mathbf{y}_i^s)\}_{i=1}^{N_s^l}$ as well as some unlabeled samples $\mathbf{X}_s^u = \{\mathbf{x}_i^s\}_{i=1}^{N_s^u}$ from source domain D_S. And a set of unlabeled samples $\mathbf{X}_t = \{\mathbf{x}_i^t\}_{i=1}^{N_t}$ are from target domain D_T. In our problem, $\mathbf{x}_i$ is the document text and $\mathbf{y}_i$ is the corresponding keyphrases.

Then, keyphrase extraction is formulated as a sequence tagging problem [18]. Given a word sequence of the text $\mathbf{x} = (x_1, x_2, ..., x_n)$, sequence tagging aims to predict labels $\mathbf{y} = (y_1, y_2, ..., y_n)$ for $\mathbf{x}$. For the i-th word x_i in text $\mathbf{x}$, its label $y_i \in \{S, B, M, E, O\}$ indicating current word is a $Single$ keyword, the $Beginning$ of a keyphrase, the $Middle$ part of the keyphrase, the End of the keyphrase, or Out of a keyphrase (i.e., the word is not any part of a keyphrase) respectively. Fig. 1 shows an example of the extracted keyphrases by sequence labeling method. As a result, we can obtain the keyphrases of the given text according to the label of each word.

IV. The TANN Model

As shown in Fig. 2, our TANN model mainly includes four components: *topic-based encoder*, *domain discriminator*, *target bidirectional decoder* and *keyphrase tagger*.

In our model, the *topic-based encoder*, the *domain discriminator* and the *target bidirectional decoder* jointly play the role of learning a text representation which incorporates the global topic information and captures both domain-invariant and domain-private features. Specifically, we first leverage the *topic-based encoder* to read the input text and build the topic-based representation. Then, to deal with the domain distribution mismatch problem, TANN learns domain-invariant features using the *domain discriminator* with adversarial training. At the same time, we use *target bidirectional decoder* to reconstruct the input text in both forward and backward directions to learn the domain-private knowledge. Finally, using the learned topic-based representation, the *keyphrase tagger* predicts the label of each token in the text.

A. Topic-based Encoder

The role of the *topic-based encoder* is to incorporate the document topic information into the text representation which can give more attention to the topic-related words. We first read the input sentence using a Bidirectional Long-Short Term Memory Network (BiLSTM) [31], [32] encoder and then get the topic-based representation leveraging the topic-correlation layer.

1) BiLSTM Encoder: BiLSTM is used to capture the sequential information. Given the input text $\mathbf{x} = (x_1, x_2, ..., x_n)$, where n is the length of the text. We first map the text to its word embedding $\mathbf{E} = (\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_n)$ by looking up word embedding from a embedding matrix $\mathbf{W}_e \in \mathbb{R}^{d_e \times V}$, where V is the size of the vocabulary and d_e is the dimension of the word embedding. BiLSTM is used to process the sequence to incorporate the context information in both directions:

$$\overrightarrow{\mathbf{h}}_i = \text{LSTM}(\mathbf{e}_i, \overrightarrow{\mathbf{h}}_{i-1}), \tag{1}$$

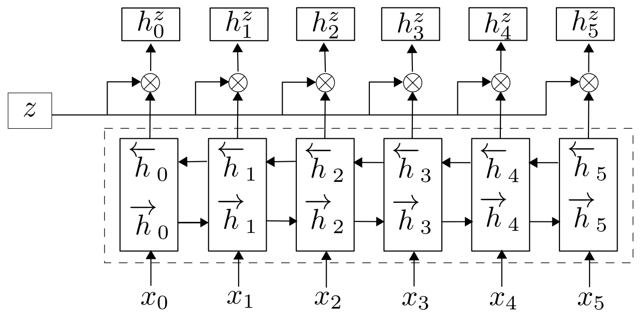

h_0^z
h_1^z
h_2^z
h_3^z
h_4^z
h_5^z
z
$\overleftarrow{h}_0$ $\overrightarrow{h}_0$
$\overleftarrow{h}_1$ $\overrightarrow{h}_1$
$\overleftarrow{h}_2$ $\overrightarrow{h}_2$
$\overleftarrow{h}_3$ $\overrightarrow{h}_3$
$\overleftarrow{h}_4$ $\overrightarrow{h}_4$
$\overleftarrow{h}_5$ $\overrightarrow{h}_5$
x_0
x_1
x_2
x_3
x_4
x_5

Fig. 3. An illustration of the Topic Correlation Mechanism in Topic-based Encoder.

$$\overleftarrow{\mathbf{h}}_i = \mathrm{LSTM}(\mathbf{e}_i, \overleftarrow{\mathbf{h}}_{i+1}). \tag{2}$$

The final hidden representation of position i is:

$$\mathbf{h}_i = [\overrightarrow{\mathbf{h}}_i; \overleftarrow{\mathbf{h}}_i]. \tag{3}$$

Each LSTM cell can be computed as follows:

$$\begin{bmatrix} \mathbf{i}_i \\ \mathbf{o}_i \\ \mathbf{f}_i \\ \tilde{\mathbf{c}}_i \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ \sigma \\ \phi \end{bmatrix} \left(\mathbf{W}_g{}^{\intercal} \begin{bmatrix} \mathbf{e}_i \\ \mathbf{h}_{i-1} \end{bmatrix} + \mathbf{b}_g \right), \tag{4}$$

$$\mathbf{c}_i = \mathbf{c}_{i-1} \odot \mathbf{f}_i + \tilde{\mathbf{c}}_i \odot \mathbf{i}_i, \tag{5}$$

$$\mathbf{h}_i = \mathbf{o}_i \odot \phi(\mathbf{c}_i), \tag{6}$$

where $\mathbf{i}, \mathbf{o}, \mathbf{f}, \mathbf{c}$ are the input gate, output gate, forget gate, and memory cell respectively. $\mathbf{W}_g \in \mathbb{R}^{(d_e+d_h)\times 4d_h}$, $\mathbf{b}_g \in \mathbb{R}^{4d_h}$ are trainable parameters. d_h is the hyper-parameter indicating the size of LSTM hidden unit. σ and ϕ are the sigmoid and tanh function respectively. $\odot$ is the element-wise multiplication operator to control the information flow.

2) Topic Correlation Layer: While the BiLSTM encoder above only considers the local context of each word, it is suggested that good keyphrases should be relevant to the global information of the document, i.e., the major topics of the given document [6], [15]. So, it is necessary to pay more attention to the words related to the documents topics. Therefore, to take the major topics of the document into consideration and get good representation for the keyphrase extraction task, we design a *topic correlation mechanism* to get the topic-based representation of the document. As shown in Fig. 3, the *topic correlation mechanism* explicitly pays attention to the words that are more relevant to the major topics of the given document by considering the correlation score between the topic vector $\mathbf{z}$ and the text words.

Concretely, we first get the topic vector $\mathbf{z}$ by computing the topic distributions of the source and target document using pre-trained Latent Dirichlet Allocation (LDA) [33] model, where $\mathbf{z} \in \mathbb{R}^k$ and k is the hyper-parameter indicating the number of topics in the document.

Then, the topic correlation gate $\mathbf{t}_i$ between the topic vector $\mathbf{z}$ and the hidden vector $\mathbf{h}_i$ is:

$$\mathbf{t}_i = \tanh(\mathbf{W}_z\mathbf{h}_i + \mathbf{U}_z\mathbf{z} + \mathbf{b}_z), \tag{7}$$

where $\mathbf{W}_z \in \mathbb{R}^{2d_h\times 2d_h}, \mathbf{U}_z \in \mathbb{R}^{2d_h\times k}$ and $\mathbf{b}_z \in \mathbb{R}^{2d_h}$ are trainable parameters. $\mathbf{t}_i$ controls the topic influence on the i-th word's hidden representation $\mathbf{h}_i$:

$$\mathbf{h}_i^z = \mathbf{h}_i \odot \mathbf{t}_i. \tag{8}$$

Finally, we get the topic-based representation $(\mathbf{h}_1^z, \mathbf{h}_2^z, ..., \mathbf{h}_n^z)$ of the text sequence.

The parameters in *topic-based encoder* are denoted as $\boldsymbol{\theta}_{enc}$.

B. Domain Discriminator

To deal with domain distribution mismatch problem, the *domain discriminator* in Fig. 2 is introduced to learn domain-invariant features to bridge these domains using domain adversarial loss. We denote the parameters of the *domain discriminator* as $\boldsymbol{\theta}_d$. During training, the *domain discriminator* is optimized to *minimize* the classification loss on $\boldsymbol{\theta}_d$ to correctly discriminate the document's representation from source and target domains. In converse, the *topic-based encoder*'s parameters $\boldsymbol{\theta}_{enc}$ are optimized to *maximize* the *domain discriminator*'s loss, which works adversarially to fool the *domain discriminator* and reduce its accuracy. Finally, this adversarial process forces the learned features in the *topic-based encoder* to be more domain-invariant, which can generalize well across domains.

In this work, we use Convolution Neural Network (CNN) as our *domain discriminator* since CNN has shown its effectiveness in various sequence classification tasks [34], [35]. Specifically, the topic-based representation $\mathbf{h}^z \in \mathbb{R}^{n\times 2d_h}$ is the input to the *domain discriminator*, where n is the text sequence length and d_h is the LSTM hidden dimension size. Then applying one convolution operation with one filter $\mathbf{W}_q \in \mathbb{R}^{c\times 2d_h}$ to a window size of c words produces a new feature map :

$$\mathbf{q} = f(\mathbf{W}_q * \mathbf{h}^z + \mathbf{b}_q), \tag{9}$$

where $*$ is the convolution operator, $\mathbf{q} \in \mathbb{R}^{n-c+1}$, $\mathbf{b}_q \in \mathbb{R}^{n-c+1}$. f is the nonlinear activation function and here we use ELU function [36]. Finally, we use a max-over-time pooling operation over the feature map and take its maximum value: $\hat{q} = max\{\mathbf{q}\}$.

The above process is for one filter. In this work, we use a number of filters n_q with different window size c to get multiple features of the text sequence. After extracting the features, we use a softmax layer to predict the domain $d \in \{0, 1\}$, where 0 and 1 indicate the source and target domain respectively.

The *domain discriminator* aims to discriminate the domain label using source and target samples. It tries to minimize the following objective:

$$\mathcal{L}_d = -\sum_{i=1}^{N_s^l+N_s^u+N_t} d_i \log \hat{d}_i + (1-d_i)\log(1-\hat{d}_i), \tag{10}$$

where $\hat{d}_i$ is the predicted probability of domain label for the i-th sample, d_i is the ground truth domain label $d_i \in \{0, 1\}$.

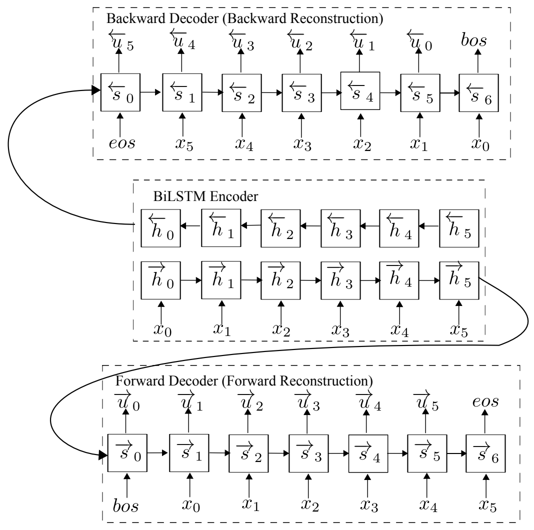

Backward Decoder (Backward Reconstruction)
BiLSTM Encoder
Forward Decoder (Forward Reconstruction)
bos
eos

Fig. 4. An illustration of bidirectional reconstruction loss using bidirectional decoder.

C. Target Bidirectional Decoder

The above adversarial process by domain discriminator mainly aims to learn the domain-invariant features and tries to eliminate all the domain-private information, which may be harmful to the target keyphrase extraction task. To preserve the domain-private information in the target domain, we propose to use a *bidirectional decoder* with bidirectional reconstruction loss in the target domain. This unsupervised objective can encourage model to preserve the main semantic information in the target domain and can efficiently utilize the unlabeled target domain data. As shown in Fig. 2, we use *target bidirectional decoder* to reconstruct the input sequence based on the output of BiLSTM encoder. Specifically, Fig. 4 shows two separate decoders to reconstruct both the forward and the backward sequences of the input text to preserve the domain-private information from both sides. The forward and backward decoders are initialized with the last hidden state in the forward and backward encoder of BiLSTM Encoder respectively.

For the forward decoder, the output is $(\overrightarrow{u}_0, ..., \overrightarrow{u}_n)$. the output probability distribution over the vocabulary for the predicted word $\overrightarrow{u}_i$ in the i-th time stamp is:

$$p(\overrightarrow{u}_i|x_{<i}) = \text{softmax}(\mathbf{W}_r^\intercal \overrightarrow{\mathbf{s}}_i + \mathbf{b}_r), \tag{11}$$

where $\mathbf{W}_r \in \mathbb{R}^{d_h \times V}, \mathbf{b}_r \in \mathbb{R}^V$ are trainable parameters. And $x_{<i}$ denotes the input words $\{x_0, ..., x_{i-1}\}$ before the i-th time stamp. The hidden state $\mathbf{s}_i$ is computed using LSTM based on the ground truth word's representation $\mathbf{e}_{i-1}$ in the previous time stamp:

$$\overrightarrow{\mathbf{s}}_i = \text{LSTM}(\mathbf{e}_{i-1}, \overrightarrow{\mathbf{s}}_{i-1}), \tag{12}$$

where $\overrightarrow{\mathbf{s}}_0 = \overrightarrow{\mathbf{h}}_n$ is the initial state. We use the last hidden state of forward encoder to initialize the forward decoder.

Then, we get each word's predicted probability distribution $(p(\overleftarrow{u}_1), ..., p(\overleftarrow{u}_n))$ over the vocabulary from the backward side similarly as the forward decoder and we initialize the backward decoder with the last hidden state $\overleftarrow{\mathbf{h}}_n$ of the backward encoder.

Finally, the target bidirectional reconstruction loss is:

$$\mathcal{L}_{recon} = -\sum_{m=1}^{N_t} \sum_{i=1}^{l_m} (x_i \log p(\overrightarrow{u}_i) + x_i \log p(\overleftarrow{u}_i)), \tag{13}$$

where l_m is length of the i-th text sequence. x_i is the one-hot vector of the i-th word, $p(\overrightarrow{u}_i)$ is predicted probability distribution over the vocabulary for the i-th word by the forward decoder, $p(\overleftarrow{u}_i)$ is the respective predicted probability distribution by the backward decoder. The parameters in the *bidirectional decoder* are denoted as $\boldsymbol{\theta}_{dec}$.

D. Keyphrase Tagger

As shown in the top right part of Fig. 2, the *keyphrase tagger* takes the output representation of the *topic-based encoder* as input to predict the label of each word in the source text. Note that we train one shared *keyphrase tagger* between two domains. Keyphrase tagging is a task where there are strong connections between the labels. For example, label M is impossible to be followed by label B. Hence, instead of tagging them independently, we model the sequence jointly using Conditional Random Field (CRF) [37].

Formally, given topic-based sequence representation $\mathbf{h}^z = (\mathbf{h}_1^z, \mathbf{h}_2^z, ..., \mathbf{h}_n^z)$ and the respective labels $\mathbf{y} = (y_1, y_2, ..., y_n)$, the conditional probability for sequence $\mathbf{y}$ is:

$$p(\mathbf{y}|\mathbf{h}^z) = \frac{\exp(g(\mathbf{h}^z, \mathbf{y}))}{\sum\limits_{\mathbf{y}' \in \mathcal{Y}(\mathbf{h}^z)} \exp(g(\mathbf{h}^z, \mathbf{y}'))}, \tag{14}$$

$$g(\mathbf{h}^z, \mathbf{y}) = \sum_{i=1}^{n} \mathbf{P}_{i,y_i} + \sum_{i=0}^{n} \mathbf{A}_{y_i, y_{i+1}}, \tag{15}$$

$$\mathbf{P} = \mathbf{W}_g^\intercal \mathbf{h}^z + \mathbf{b}_g, \tag{16}$$

where $\mathcal{Y}(\mathbf{h}^z)$ is all the possible label sequences for $\mathbf{h}^z$, $\mathbf{P} \in \mathbb{R}^{n \times |\mathcal{L}|}$ is the matrix of scores where $|\mathcal{L}|$ is the number of the labels. $\mathbf{P}_{i,y_i}$ is the score of assigning the i-th word with label y_i. $\mathbf{A}_{y_i,y_{i+1}}$ is the label transition matrix and we only consider the interactions between two successive labels, where $\mathbf{A}_{i,j}$ indicates the transition score from label i to label j. y_0 and y_{n+1} is the $start$ and end tags of the sequence. So A is a square matrix of size $k+2$, k is the number of labels, which is 5. $\mathbf{W}_g \in \mathbb{R}^{2d_h \times |\mathcal{L}|}$ and $\mathbf{b}_g \in \mathbb{R}^{|\mathcal{L}|}$ are trainable parameters. We denote the parametes of the Keyphrase Tagger as $\boldsymbol{\theta}_{tag}$.

For CRF's training, we minimize the negative log-likelihood over the source labeled samples:

$$\mathcal{L}_{tagger} = -\sum_{i=1}^{N_s^l} \log p(\mathbf{y}|\mathbf{h}^z). \tag{17}$$

During test, we search the label sequence $\mathbf{y}^*$ with the highest conditional probability:

$$\mathbf{y}^* = \underset{\mathbf{y} \in \mathcal{Y}(\mathbf{h}^z)}{\text{argmax}}\, p(\mathbf{y}|\mathbf{h}^z),$$

where $\mathbf{y}^*$ can be efficiently computed using Viterbi algorithm [38].

Algorithm 1 Adversarial Training Procedure for TANN Model.

```
Input: Training data $\{\{(\mathbf{x}_i^s, \mathbf{y}_i^s)\}_{i=1}^{N_s^l}, \{\mathbf{x}_i^s\}_{i=1}^{N_s^u}, \{\mathbf{x}_i^t\}_{i=1}^{N_t}\}$
       Parameters: $\alpha, \beta$ – hyper-parameters of the loss weight
1: Initialize model's parameters $\{\boldsymbol{\theta}_{enc}, \boldsymbol{\theta}_{tag}, \boldsymbol{\theta}_{dec}, \boldsymbol{\theta}_d\}$
2: repeat
3:     sample a half mini-batch from source domain $\{\mathbf{x}_i^s\}_{i=1}^{N_s^l+N_s^u}$ and a half mini-batch from target domain $\{\mathbf{x}_i^t\}_{i=1}^{N_t}$. Update $\boldsymbol{\theta}_d$ using gradient descent $\nabla\mathcal{L}_d$ in Eq.(10)
4:     sample a mini-batch source data $\{(\mathbf{x}_i^s, \mathbf{y}_i^s)\}_{i=1}^{N_s^l}$, sample a mini-batch from target $\{\mathbf{x}_i^t\}_{i=1}^{N_t}$. Update $\boldsymbol{\theta}_{enc}, \boldsymbol{\theta}_{tag}, \boldsymbol{\theta}_{dec}$ using gradient descent $\nabla\mathcal{L}_f$ in Eq.(19)
5: until convergence
Output: Topic-based Encoder ($\boldsymbol{\theta}_{enc}$) and Keyphrase Tagger ($\boldsymbol{\theta}_{tag}$) for prediction in the target domain
```

E. Adversarial Training Procedure

In this section, we will detail the training procedure of our model. The overall training objective can be seen as a min-max game played among the *topic-based encoder* (including BiLSTM Encoder and Topic Correlation Layer), *keyphrase tagger*, *target bidirectional decoder* and *domain discriminator*:

$$\min_{\boldsymbol{\theta}_{enc}, \boldsymbol{\theta}_{tag}, \boldsymbol{\theta}_{dec}} \max_{\boldsymbol{\theta}_d} \mathcal{L}_{tagger} + \alpha\mathcal{L}_{recon} - \beta\mathcal{L}_d, \tag{18}$$

where α, β are the weights of the reconstruction loss and reverse domain classification loss respectively.

Specifically, we update the parameters of *domain discriminator* $\boldsymbol{\theta}_d$ directly by minimizing $\mathcal{L}_d$ using Eq.(10).

To fool the *domain discriminator* and learn the domain-invariant features, we use the reverse gradient of the domain loss $\mathcal{L}_d$ to update the parameters of $\boldsymbol{\theta}_{enc}$. And we also combine the above objectives Eq.(13), (17) and minimize the total loss $\mathcal{L}_f$ to update the parameters of $\boldsymbol{\theta}_{enc}$:

$$\mathcal{L}_f = \mathcal{L}_{tagger} + \alpha\mathcal{L}_{recon} - \beta\mathcal{L}_d. \tag{19}$$

Finally, we present the adversarial training algorithm for the TANN model in Algorithm 1. The training procedure is optimized alternatively between Eq.(10) for $\boldsymbol{\theta}_d$ and Eq.(19) for $\boldsymbol{\theta}_{enc}, \boldsymbol{\theta}_{tag}, \boldsymbol{\theta}_{dec}$. All parameters are optimized using standard backpropagation.

V. Experimental Setup

A. Dataset

To the best of our knowledge, there are no publicly-available cross-domain keyphrase extraction datasets for keyphrase extraction tasks, so we construct a dataset of research papers from different domains in computer science domain since research papers are widely studied in previous works [12], [15], [18].

The selection of the domains is according to the recommended international academic conferences and journals [1] by the China Computer Federation (CCF), which were already categorized into different domains. We crawled the corresponding paper abstracts in these domains. Here, we selected three different domains where the number of abstracts is larger than other domains: Data Mining (DM), Software and Language (SL) and Computer Graphics (CG). Some basic statistics of the crawled dataset are summarized in Table I. Table I describes the number of labeled (i.e., with author-assigned keyphrases) and unlabeled abstracts in each domain where AvgKp is the average number of the author-assigned keyphrases per abstract, and AvgLength is the average number of words per abstract.

TABLE I
The statistics of the dataset.

Domain	Labeled	Unlabeled	AvgKp	AvgLength
DM	10,550	2,146	3.73	180
SL	1,476	8,639	4.55	162
CG	1,423	6,716	4.91	155

TABLE II
Jaccard distance between the phrases set of two different domains.

Domain Pairs	Jaccard Distance		
	unigrams	bigrams	trigrams
DM and SL	0.817	0.827	0.928
DM and CG	0.814	0.850	0.940

As there are a large number of labeled abstracts in domain DM, we denote the DM domain as source domain S and the other two domains as target domain T. For each transfer pair $S \rightarrow T$, we randomly sampled 500 abstracts from Labeled data in domain DM as validation data. For the test dataset in the domain SL and CG, we randomly sampled 500 abstracts from the labeled data in domain SL and CG respectively. Note that in the main settings, we did not use any labeled data from the target domain T during training. To carry the experiments with limited data in Section VI-D, we also randomly sampled the rest 800 abstracts as additional target domain training data from the labeled data in domain SL and CG respectively.

Table II shows the discrepancy between different domains. Specifically, We calculated the Jaccard Distance [2] of two sets of phrases of different length in the domain D_1 and D_2 to estimate the overlap degree of the two sets phrases in the document:

$$J_V(D_1, D_2) = 1 - \frac{|V_1 \cap V_2|}{|V_1 \cup V_2|}, \tag{20}$$

where V_1, V_2 are the phrases for D_1 and D_2 respectively. Specifically, we use n-grams which is consecutive words of length n in the abstracts as the phrases set. $J_V(D_1, D_2) \in [0, 1]$, and the high value means large discrepancy. The results for unigrams, bigrams and trigrams are in Table II. As the length of the phrase increase, the jaccard distance between the two sets of phrases becomes significantly

[1]http://www.ccf.org.cn/xspj/gyml/

[2]http://en.wikipedia.org/wiki/Jaccard index

larger. The high value of J_V in Table II indicates that there is an explicit *distribution mismatch* between these domains.

B. Implementation Details

We combined one paper title and abstract as one text. The text was pre-processed with word tokenization, replacing the digits with zero. We selected the words with the frequency more than 15 as vocabulary, and the word embeddings $\mathbf{W}_e$ were initialized with public 300-dimensional glove vectors [3].

The hidden layer dimensions d_h of biLSTM encoder and target bidirectional decoder were set to 300. For the CNN discriminator, we used filter windows (c) of sizes $\{3, 4, 5\}$ with $n_q = 200$ feature maps each. The weights in the network were initialized with Xavier initialization [39]. We used dropout [40] on the input embeddings and the input of the CNN discriminator with dropout rate 0.5.

Similar to [17], the adversarial strength β was increased gradually during training to prevent the domain discriminator from the noisy signal in the early training phrase. And $\beta = \frac{2}{1+\exp(-10 \cdot p)} - 1$, where $p = \frac{t}{T}$, t is the current epoch and $T = 100$ is the maximum epoch. The reconstruction loss strength α was set to 0.2 and 0.05 in the DM $\rightarrow$ SL and DM $\rightarrow$ CG respectively. The topic numbers used in topical correlation in both source and target domains were set to 50 and 100 in the DM $\rightarrow$ SL and DM $\rightarrow$ CG respectively. We further analyse the influence of the hyperparameters in the following section. In order to compute topic distribution of the text, we used LDA implementation in the topic modeling toolkit [41] to train the topic model. We trained the model using about 500,000 scientific papers collected by [12].

We trained the model parameters using Stochastic Gradient Descent plus momentum [42] with learning rate 0.1, and gradient clipping value was 1. We used mini-batch size 64. Models were selected using early stop. [4]

C. Baseline Methods

To validate the performance of TANN model, several state-of-the-art models were selected as baseline methods.

As our target domain is unlabeled, we first compared our model against four unsupervised keyphrase extraction methods directly on the target domain: **TF-IDF**, **TextRank**, [7], **ExpandRank** [8], and **Topical PageRank(TPR)** [15]. For these four methods, we used the optimal settings according to their papers. The topic distribution of words in TPR was obtained following the same procedure as our TANN model. For these unsupervised methods, we considered top 5 ranked phrases when computing the F1-score.

Supervised baseline methods are as follows:

- **CRF**: Keyphrase extraction can be formulated as a sequence tagging problem. Here we train a keyphrase tagger using CRF with a lot of hand-crafted features for keyphrase extraction in [18], including parse-tree features, stopword features and compound features. We trained CRF using source labeled data.

[3]https://nlp.stanford.edu/projects/glove/

[4]Code will be released at https://github.com/wwwyn/TANN

TABLE III
F1-SCORE OF DIFFERENT METHODS. BOLD NUMBERS ARE THE BEST SCORES.

Methods	DM $\rightarrow$ SL	DM $\rightarrow$ CG
TF-IDF	0.156	0.129
TextRank	0.170	0.175
ExpandRank	0.152	0.124
TPR	0.196	0.160
CRF	0.164	0.147
Joint-layer RNN	0.185	0.165
SourceOnly(BiLSTM-CRF)	0.179	0.167
BiLSTM-CRF+MMD	0.211	0.187
TANN	**0.296**	**0.243**

- **Joint-layer RNN**: Joint-layer RNN [25] is a recent state-of-the-art neural tagging method for keyphrase extraction. Here we used the labeled source domain data to train the Joint-layer RNN model.
- **SourceOnly**: SourceOnly is part of our model, with the topic correlation layer, adversarial training and target bidirectional decoder removed from our TANN model and trained in the source domain. Note that our SourceOnly model is a BiLSTM-CRF model [43].
- **BiLSTM-CRF+MMD**: This is a transfer learning method. We add Maximum Mean Discrepancy (MMD) [44] regularization at the output layer of the BiLSTM to minimize the domain discrepancy between the source and target domain. Here we used the Gaussian kernel in MMD, the standard deviation parameters in MMD was turned on the validation set. BiLSTM-CRF + MMD used labeled data in the source domain and unlabeled data in both domains.

D. Evaluation Metrics

Similar to previous methods, we used keyphrases that appear in the text as the ground truth keyphrases [18]. When determining the match of two keyphrases, we used Porter's stemmer [45] for preprocessing. We used F1-score to evaluate the performance. The F1-score is computed as follows: $P = \frac{\#Correct}{\#Extract}$, $R = \frac{\#Correct}{\#Truth}$, $F1 = \frac{2 \times P \times R}{P+R}$, where $\#Correct, \#Extract, \#Truth$ is the number of correctly predicted, number of predicted and number of ground-truth keyphrases respectively for all the test documents.

VI. RESULTS AND ANALYSIS

A. Overall Performance

Table III shows the F1-scores of TANN model and baseline methods on the two cross-domain keyphrase extraction tasks. We observe that TANN model consistently outperforms all the baseline methods by a large margin. The TANN model outperforms the baseline models by at least 8.5% and 5.6% when the target domain is SL and CG respectively. These results show that by utilizing both labeled data in the source domain and unlabeled data in the source and target domain, our TANN model significantly improve the performance in the unlabeled target domain.

TABLE IV
ABLATION STUDY FOR TANN MODEL.

Model	F1-score	
	DM $\rightarrow$ SL	DM $\rightarrow$ CG
TANN	**0.296**	**0.243**
−adversarial	0.214	0.179
−topic	0.238	0.185
−reconstruction	0.250	0.227
−backward reconstruction	0.282	0.236

TABLE V
F1-SCORE FOR DIFFERENT TOPIC NUMBER k OF TANN MODEL.

topic number k	DM $\rightarrow$ SL	DM $\rightarrow$ CG
10	0.248	0.221
20	0.285	0.234
30	0.284	0.225
50	**0.296**	0.200
100	0.280	**0.243**
200	0.279	0.234
300	0.293	0.232

Among the baseline methods, directly using unsupervised ranking based methods on the target domain performs quite well compared with the supervised methods trained on the source domain and applied on the target domain. In the DM $\rightarrow$ SL task, TPR performs the second best among the baseline methods, perhaps because TPR also leverages the topic information and runs PageRank once for each topic. TextRank is the second best baseline method in the DM $\rightarrow$ CG task but still significantly lower than the TANN model. The poor performance of CRF method with hand-crafted features demonstrates that the designed rules may not grasp the deep semantic information of the text. For Joint-layer RNN method, it is not surprising that it performs poorly across different domains as it has a large number of parameters and may overfit in the source domain, consequently failing to adapt to the target domain.

SourceOnly model provides an empirical performance lower bound of our model. The quite poor performance of SourceOnly compared to TANN model indicates that there is a distribution mismatch between the source domain and target domain, and directly applying the model trained in the source domain to target domain may fail to extract keyphrase in the target domain. While TANN model can efficiently deal with the domain distribution mismatch by using adversarial learning. By adding MMD regularization to minimize the domain discrepancy, the performance of BiLSTM-CRF+MMD is better than SourceOnly model but is still lower than TANN model.

From Table III, we also observe that TANN's performance on DM $\rightarrow$ CG task is worse than the performance on DM $\rightarrow$ SL task. The reason may be that the average domain discrepancy between DM $\rightarrow$ CG is larger than that of DM $\rightarrow$ SL which is reported in Table II.

TABLE VI
F1-SCORE FOR DIFFERENT RECONSTRUCTION LOSS WEIGHT α OF TANN MODEL.

α value	DM $\rightarrow$ SL	DM $\rightarrow$ CG
0.01	0.239	0.215
0.05	0.241	**0.243**
0.1	0.265	0.222
0.2	**0.296**	0.231
0.4	0.226	0.226
0.8	0.244	0.210

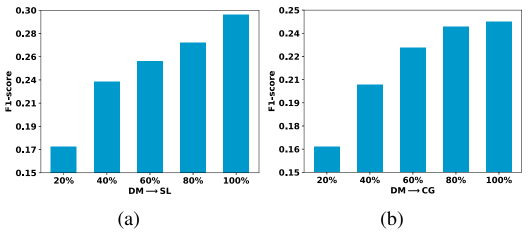


Fig. 5. Performance with varying percentage of source labeled data N_s^l.

B. Ablation Study for Model Components

Table IV shows F1-score on two target domains by removing the proposed adversarial training, topic correlation layer and bidirectional reconstruction loss from the TANN model respectively. We observe substantial drops of 8.2%, 5.8%, 4.6% by removing adversarial training, topic correlation layer and bidirectional reconstruction loss respectively in the DM $\rightarrow$ SL task. Similar significant drops are 6.4%, 5.8%, 1.6% respectively in the DM $\rightarrow$ CG task. These results indicate the importance of the proposed adversarial training, topic correlation layer and target bidirectional reconstruction loss for keyphrase extraction task. We also observe that our TANN (Full) outperforms the TANN-(Backward Reconstruction) in the two target domains, indicating adding backward reconstruction loss can further improve the performance.

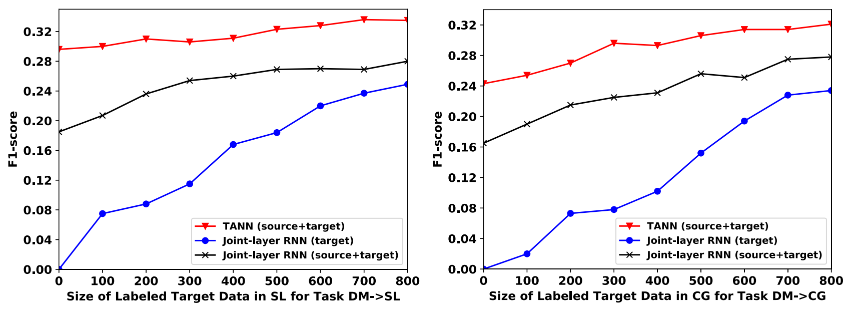
C. Influence of Hyperparameters

We conduct several experiments on the influence of model hyperparameters, including the number of topics c, reconstruction loss weight α and the source labeled data size N_s^l.

We demonstrate the influence of the topic number k used in the topic correlation layer in Table V. Table V shows different numbers of topics ranging from 10 to 300. The best F1-score is obtained when the topic number is 50 for DM $\rightarrow$ SL and 100 for DM $\rightarrow$ CG. The performance is poor when the topic number is very smaller such as $k = 10$.

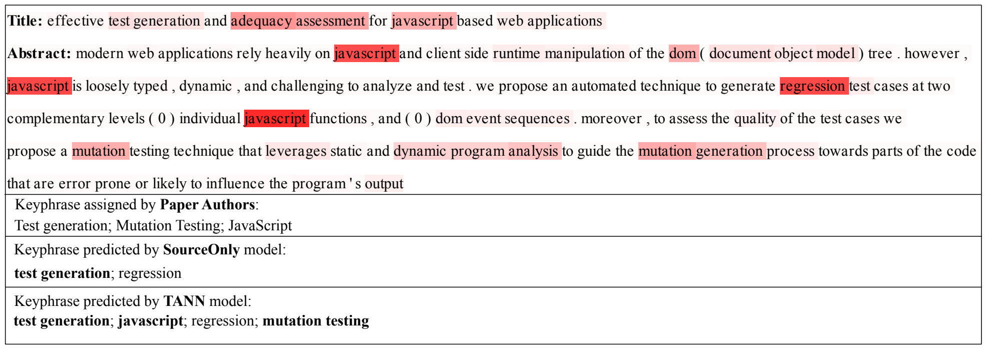
Table VI shows the F1-score of TANN model over different reconstruction loss weight α. We observe that too small reconstruction loss weight ($\alpha = 0.01$) or too large weight ($\alpha = 0.8$) are poor for the TANN performance. TANN performs best when α is 0.2 for the DM $\rightarrow$ SL task and $\alpha = 0.05$ for the DM $\rightarrow$ CG task.

As the target domain is unlabeled and TANN model only utilizes the labeled data from the source domain, we investigate the impact of the different size of source labeled training data



(a) (b)

Fig. 6. Performance of TANN model with various amount of labeled target data in SL and CG domain.



Title: effective test generation and adequacy assessment for javascript based web applications

Abstract: modern web applications rely heavily on javascript and client side runtime manipulation of the dom (document object model) tree . however , javascript is loosely typed , dynamic , and challenging to analyze and test . we propose an automated technique to generate regression test cases at two complementary levels (0) individual javascript functions , and (0) dom event sequences . moreover , to assess the quality of the test cases we propose a mutation testing technique that leverages static and dynamic program analysis to guide the mutation generation process towards parts of the code that are error prone or likely to influence the program ' s output

Keyphrase assigned by **Paper Authors**:
Test generation; Mutation Testing; JavaScript

Keyphrase predicted by **SourceOnly** model:
test generation; regression

Keyphrase predicted by **TANN** model:
test generation; **javascript**; regression; **mutation testing**

Fig. 7. An example of the predicted keyphrases by SourceOnly model and our TANN model in the DM $\rightarrow$ SL task, bold phrases indicate correct predictions. We also visualize the topic correlation weights in the paper title and abstract, deeper color means larger correlation weights.

n_s^l. As shown in Fig. 5, as the amount of source labeled data increases, the F1-score of our model consistently improves especially when the source labeled data size changes from 20% to 40%.

D. Experiments on Target Domain with Limited Label

In the above settings, we consider the tough situation where there is no labeled data in the target domain. We also test the proposed TANN model when additional small amount of annotated data can be obtained in the resource-poor target domain. A simple way to leverage the small amount of labeled data in the target domain is to fine-tune the model after pre-training TANN model in the above settings. Fig. 6 shows the result of TANN model trained with different size of labeled target data compared with Joint-layer RNN. As we can see, the margin is significantly large when the size of labeled target data is range from 0 to 800 comparing TANN with the Joint-layer RNN trained directly on the target domain. These results demonstrate that when the target labeled data is scarce, TANN model can significantly improve the performance in the target domain by leveraging source labeled data and unlabeled data in both domains. By leveraging a large amount of source labeled data, Joint-layer RNN(source + target) outperforms Joint-layer RNN(target). However, TANN model still outperforms Joint-layer RNN(source + target) by a large margin, indicating the importance of leveraging unlabeled data.

E. Case Study

Finally, Fig. 7 shows the predicted keyphrases of an abstract in the SL domain by SourceOnly Model and our TANN model when performing the transfer task from DM to SL domain. Clearly, TANN model predicts all the three author-assigned keyphrases while SourceOnly model only predicts one correct keyphrase. Moreover, the keyphrases predicted by the SourceOnly model only grasp one small point of this abstract. However, TANN successfully predicts the word "*javascript*", which reflects the main topic of this abstract. We also visualize the topic correlation layer in TANN model. As the topic correlation weights $\mathbf{t}_i$ is a multidimensional vector, we use the first-derivative saliency [46] to visualize the topic correlation weights $\mathbf{t}_i$ of the words in Fig. 7 and we observe that the word "*assessment/javascript/dom/regression/mutation*" has larger correlation weights. The results demonstrate that TANN can not only capture the global topic information but also give more attention to the topic-related phrases.

VII. Conclusion and Future Work

In this paper, we proposed the TANN model for cross-domain keyphrase extraction to address the problem of limited labeled data in the target domain by means of resource-rich labeled data. The proposed TANN model can efficiently make use of data in the resource-rich domain and unlabeled data in target domain. We first utilized a topic-correlation layer to incorporate the global topic information into the document representation. Then, transferable knowledge was learned by using adversarial training given the topic-based representation. At the same time, target domain-specific information was preserved by leveraging bidirectional reconstruction loss. Extensive experiments showed that our TANN model can substantially improve the performance in the unlabeled or insufficiently labeled target domain. In the future, we would like to test our model on transfer pairs between more different domains such as biological, physical and chemistry domains. And we would like to extend our framework to other cross-domain tasks like cross-domain recommendation.

Acknowledgement

This research was supported in part by grants from the National Natural Science Foundation of China (Grant No. 91746301 and 61703386), the National Key Research and Development Program of China (No. 2016YFB1000904), the Anhui Provincial Natural Science Foundation (Grant No. 1708085QF140), and the Fundamental Research Funds for the Central Universities (Grant No. WK2150110006). The author wants to thank Wang Zhe for constructive criticism on the earlier draft.

References

[1] A. Hulth and B. B. Megyesi, "A study on automatically extracted keywords in text categorization," in *ACL*, 2006, pp. 537–544.

[2] N. Pudota, A. Dattolo, A. Baruzzo, F. Ferrara, and C. Tasso, "Automatic keyphrase extraction and ontology mining for content-based tag recommendation," *International Journal of Intelligent Systems*, vol. 25, no. 12, pp. 1158–1186, 2010.

[3] G. Berend, "Opinion expression mining by exploiting keyphrase extraction," in *IJCNLP*, 2011.

[4] E. Frank, G. W. Paynter, I. H. Witten, C. Gutwin, and C. G. Nevill-Manning, "Domain-specific keyphrase extraction," in *IJCAI*, vol. 2, 1999, pp. 668–673.

[5] A. Hulth, "Improved automatic keyword extraction given more linguistic knowledge," in *EMNLP*, 2003, pp. 216–223.

[6] Z. Liu, C. Liang, and M. Sun, "Topical word trigger model for keyphrase extraction," pp. 1715–1730, 2012.

[7] R. Mihalcea and P. Tarau, "Textrank: Bringing order into text," in *EMNLP*, 2004.

[8] X. Wan and J. Xiao, "Single document keyphrase extraction using neighborhood knowledge." in *AAAI*, vol. 8, 2008, pp. 855–860.

[9] Z. Liu, P. Li, Y. Zheng, and M. Sun, "Clustering to find exemplar terms for keyphrase extraction," in *EMNLP*, 2009, pp. 257–266.

[10] T. Tomokiyo and M. Hurst, "A language model approach to keyphrase extraction," in *Proceedings of the ACL 2003 workshop on Multiword expressions: analysis, acquisition and treatment-Volume 18*, 2003, pp. 33–40.

[11] O. Medelyan, E. Frank, and I. H. Witten, "Human-competitive tagging using automatic keyphrase extraction," in *EMNLP*, 2009, pp. 1318–1327.

[12] R. Meng, S. Zhao, S. Han, D. He, P. Brusilovsky, and Y. Chi, "Deep keyphrase generation," *CoRR*, vol. abs/1704.06879, 2017.

[13] R. Xu and Y. Yang, "Cross-lingual distillation for text classification," in *ACL*, 2017.

[14] N. Biadsy, L. Rokach, and A. Shmilovici, "Transfer learning for content-based recommender systems using tree matching," in *CD-ARES*, 2013.

[15] Z. Liu, W. Huang, Y. Zheng, and M. Sun, "Automatic keyphrase extraction via topic decomposition," in *EMNLP*, 2010, pp. 366–376.

[16] K. S. Hasan and V. Ng, "Conundrums in unsupervised keyphrase extraction: making sense of the state-of-the-art," in *COLING*, 2010, pp. 365–373.

[17] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *Journal of Machine Learning Research*, vol. 17, no. 59, pp. 1–35, 2016.

[18] S. D. Gollapalli, X.-L. Li, and P. Yang, "Incorporating expert knowledge into keyphrase extraction." in *AAAI*, 2017, pp. 3180–3187.

[19] K. S. Hasan and V. Ng, "Automatic keyphrase extraction: A survey of the state of the art," in *ACL*, vol. 1, 2014, pp. 1262–1273.

[20] F. Xie, X. Wu, and X. Zhu, "Document-specific keyphrase extraction using sequential patterns with wildcards," in *ICDM*. IEEE, 2014, pp. 1055–1060.

[21] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information processing & management*, vol. 24, no. 5, pp. 513–523, 1988.

[22] I. H. Witten, G. W. Paynter, E. Frank, C. Gutwin, and C. G. Nevill-Manning, "Kea: Practical automatic keyphrase extraction," in *Proceedings of the fourth ACM conference on Digital libraries*. ACM, 1999, pp. 254–255.

[23] H. Chen, K. Xiao, J. Sun, and S. Wu, "A double-layer neural network framework for high-frequency forecasting," *ACM Trans. Management Inf. Syst.*, vol. 7, pp. 11:1–11:17, 2017.

[24] Q. Liu, Z. Huang, Z. Huang, C. Liu, E. Chen, Y. Su, and G. Hu, "Finding similar exercises in online education systems," in *KDD*, 2018.

[25] Q. Zhang, Y. Wang, Y. Gong, and X. Huang, "Keyphrase extraction using deep recurrent neural networks on twitter." in *EMNLP*, 2016, pp. 836–845.

[26] Z. Wang, W. He, H. Wu, H. Wu, W. Li, H. Wang, and E. Chen, "Chinese poetry generation with planning based neural network," in *COLING*, 2016.

[27] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *NIPS*, 2014, pp. 2672–2680.

[28] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial autoencoders," *CoRR*, vol. abs/1511.05644, 2015.

[29] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep domain confusion: Maximizing for domain invariance," *CoRR*, vol. abs/1412.3474, 2014.

[30] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, "Analysis of representations for domain adaptation," in *NIPS*, 2007, pp. 137–144.

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

[32] A. Graves, "Supervised sequence labelling," in *Supervised sequence labelling with recurrent neural networks*. Springer, 2012, pp. 5–13.

[33] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.

[34] Y. Kim, "Convolutional neural networks for sentence classification," *CoRR*, vol. abs/1408.5882, 2014.

[35] X. Zhang and Y. LeCun, "Text understanding from scratch," *CoRR*, vol. abs/1502.01710, 2015.

[36] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *CoRR*, vol. abs/1511.07289, 2015.

[37] J. D. Lafferty, A. D. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *ICML*, 2001.

[38] M. S. Ryan and G. R. Nudd, "The viterbi algorithm," 2009.

[39] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *AISTATS*, 2010, pp. 249–256.

[40] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[41] R. Rehurek and P. Sojka, "Software framework for topic modelling with large corpora," in *In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Citeseer, 2010.

[42] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *ICML*, 2013, pp. 1139–1147.

[43] Z. Huang, W. Xu, and K. Yu, "Bidirectional lstm-crf models for sequence tagging," *CoRR*, vol. abs/1508.01991, 2015.

[44] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *Journal of Machine Learning Research*, vol. 13, no. Mar, pp. 723–773, 2012.

[45] M. F. Porter, "An algorithm for suffix stripping," *Program*, vol. 14, no. 3, pp. 130–137, 1980.

[46] J. Li, X. Chen, E. Hovy, and D. Jurafsky, "Visualizing and understanding neural models in nlp," *CoRR*, vol. abs/1506.01066, 2015.