Cross-view Adaptation Network for Cross-domain Relation Extraction

Bo Yan[1]✉, Dongmei Zhang[1], Huadong Wang[2], and Chunhua Wu[3]

[1] School of computer science, Beijing university of posts and telecommunications, China
{mjkbyb,zhangdm}@bupt.edu.cn
[2] Samsung Research China-Beijing (SRC-B)
huadong.wang@samsung.com
[3] School of cyberspace security, Beijing university of posts and telecommunications, China
wuchunhua@bupt.edu.cn

Abstract. In relation extraction, directly adopting a model trained in the source domain to the target domain will suffer greatly performance decrease. Existing studies extract the shared features between domains in a coarse-grained way, which inevitably introduce some domain-specific features or suffer from information loss. Inspired by human beings often using different views to find connection between domains, we argue that, there exist some fine-grained features which can be shared across different views of origin data. In this paper, we proposed a cross-view adaptation network, which use adversarial method to extract shared features and introduce cross-view training to fine-turn it. Besides, we construct some novel views of input data for cross-domain relation extraction. Through experiments we demonstrated that the different views of data we construct can effectively avoid introducing some domain-specific features into unified feature space and help the model learn a fine-grained shared features of different domain. On the three different domains of ACE 2005 dataset, Our method achieved the state-of-the-art results in F1-score.

Keywords: Relation Extraction · Domain Adaption · Cross-view Training · Adversarial Training

1 Introduction

Relation extraction refers to extracting the relation between entities within a sentence. For example, given the sentence "*His **hometown** is **Beijing***", we can extract Located relation between "***hometown***" and "***Beijing***". Relation extraction is often seen as a supervised classification task, and many methods have show great performance on it. However, for the relation extraction across domains, due to the different data distribution, the models learned in one domain directly applied to another domain often have poor performance, so recent research pay more attention to cross-domain relation extraction.

To solve the problem of domain adaptation, a simple method is training a model on the source domain and using target domain data to fine-tune it [1,10]. However, this method requires expensive labeling costs and expects a high quality data distribution on the target domain. [9,13,14,15,19] use some manually crafting features such as word clustering and dependency path to adapt models. This method effectively captures some domain invariant features, but will loss information due to the limited human knowledge. With the development of Adversarial Training in recent years, some studies have used adversarial method to automatically extract domain invariant features. [7] used CNN with multiple kernels as shared feature extractor, to extract domain invariant features through adversarial training, and jointly optimizes with relation classifier. [16] train the model on the source domain, and use another model to extract target domain features to match the source feature distribution.

Most of the above studies project the shared features and private features into one unified space, to make the model learn the shared features of different domains, but these methods inevitably introduce some domain-specific features. [17] using the domain separation network [3] to extract the domain-specific features and domain-general features separately, so as to limit domain-specific features into shared feature space. Though it introduce reconstruction loss, but still suffer from information loss due to the dimension reduction of intermediate representation and at the same time the model becomes more complicated.

To address these problems, we proposed a cross-view adaptation network which adopted cross-view training [6] on the target domain. Source labeled data and target unlabeled data is fed into a shared feature extractor to learn a common representation and produce relation prediction using these features. Besides these full-view data, we construct some restrict-view data which loss some contextual information on the target domain, such as masking the entity word. These restrict-view data also be fed into shared feature extractor to produce prediction. Then full-view datas prediction will act as a teacher to teach the different restrict-view data learn the same prediction. By matching the predict distribution, the model can learn some contextual information that dont relevant to target-specific features such as entity word.

The major contributions of this paper are as follows:

- A novel domain adaptation method for relation extraction is proposed, which uses cross-view training on the target domain to fine-tune the shared feature space. This method can make the model learn some fine-grained shared features and more effectively adapt to the target domain.
- We construct different views of target domain data for relation extraction. Experiments on ACE 2005 dataset shows our model can significant improve the cross-domain relation extraction performance.

2 Cross-view Adaptation Network

In this section, we present a adaptation method for the cross-domain relation extraction task. This task can be formulated as follows: given a labeled

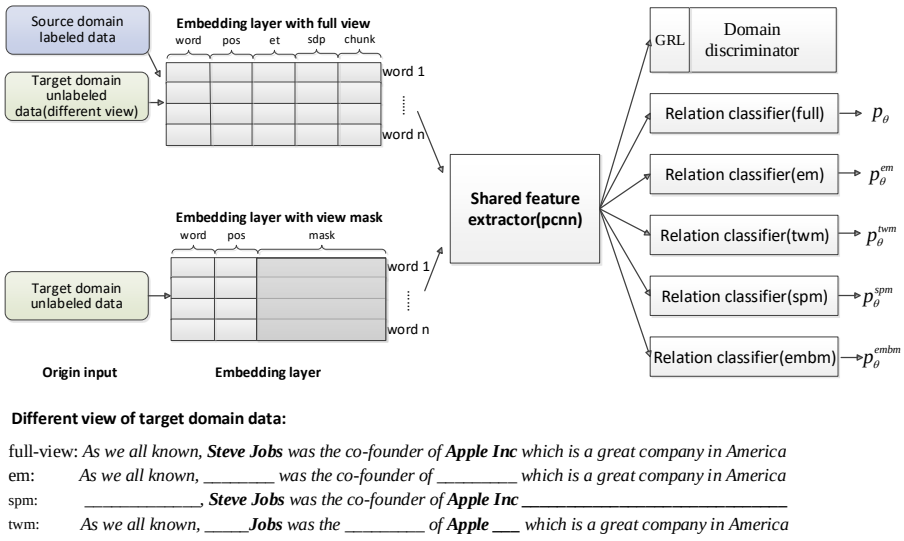

Fig. 1. The overall of cross-view domain adaptation network. We construct different views on target domain unlabeled data. In the origin input layer, we mask some words such as entity pairs. In the embedding layer, we mask some embeddings such as entity type. During training, source domain labeled data and target domain unlabeled data are fed into the network to extract shared features though domain discriminator and relation classifier (full). Different views of target domain data (origin input mask and embedding mask) will be used to fine-tune the shared feature extractor through matching the distribution of relation classifier (full)'s output. In the test stage, we use relation classifier(full) to get the predict of the model. The particular example shows the different views of target data.

source domain corpus $\mathcal{S} = \{(s_1, e_{11}, e_{12}, r_1), \ldots, (s_S, e_{S1}, e_{S2}, r_S)\}$, where $s_i = [w_1, \ldots, w_m]$ denotes a word sentence, e_{i1} and e_{i2} represent the candidate entity pairs and r_i denotes their relation type. The goal of this task is to build a relation extraction model on S and apply it to an unlabeled target domain corpus $\mathcal{T} = \{(\hat{s}_1, \hat{e}_{11}, \hat{e}_{12}), \ldots, (\hat{s}_T, \hat{e}_{T1}, \hat{e}_{T2})\}$. In other words, for the source domain, the data is labeled, but without any labels in the target domain. Using $\mathcal{S}$ and $\mathcal{T}$ as training data to train a model, so that given a sentence of the target domain and two candidate entities, the model can correctly extract the relation between the entity pairs. The overall of our model is shown in Fig. 1.

2.1 Embedding layer

From the previous work on relation extraction, we know that embedding features improve relation extraction a lot. Motivated by the work [7,9,13], the embedding layer consists of the following parts:

Word embedding We use pre-trained 300 dimensions word embedding from word2vec [12], and every word w_i is converted to the corresponding vector by looking up word embedding table $\boldsymbol{W}$.

Position embedding The position of a word refers to the relative distance between the word and two entities respectively. For example, i and j are the position index of the two entities in a sentence. For the word with index k, its position is $k-i$ and $k-j$. After getting the position of the word, we can get the corresponding position embedding $\boldsymbol{p}_{i1}$ and $\boldsymbol{p}_{i2}$ by looking up position embedding table $\boldsymbol{P}$.

Entity type embedding Each entity has a type to which it belongs. For each entity in the sentence, we get its entity type embedding $\boldsymbol{t}_i$ through entity type embedding table E. Non-entity word will have the same entity type embedding $\boldsymbol{t}_i$. For every word in the sentence, we will get two entity type embedding $\boldsymbol{t}_{i1}$ and $\boldsymbol{t}_{i2}$ because we have two candidate entities in each sentence.

Chunks embedding Chunks is regard as a phrase that has a specific structure and a relatively stable meaning. We use the method of sequence labeling to get the chunks of the sentence, so that each word has a chunks representation, and then use the chunks embedding table $\boldsymbol{C}$ to get the chunks embedding $\boldsymbol{c}_i$.

Dependency path embedding We use a binary number to indicate whether a word is on the shortest dependency tree path between two entities, and use dependency path table $\boldsymbol{D}$ to get the dependency path embedding $\boldsymbol{d}_i$ for each word.

At last, we concatenate all above types of embedding to get the representation of one word: $\boldsymbol{v}_i = [\boldsymbol{e}_i; \boldsymbol{p}_{i1}; \boldsymbol{p}_{i2}; \boldsymbol{t}_{i1}; \boldsymbol{t}_{i2}; \boldsymbol{c}_i; \boldsymbol{d}_i]$, all the embeddings are randomly initialized and optimized during training except for the pre-trained word embedding.

2.2 Shared feature extractor

We use Piecewise Convolutional Neural Networks (PCNN) [20] as the shared feature extractor. PCNN pays attention to the distance and position of entities, and the context information near the entities, which are the most important features in the relation extraction. It divides a sentence into three parts according to the entity position, such as "*As we all known*, ***Steve Jobs*** *was the co-founder of* ***Apple Inc*** *which is a great company in America*", which will be divide into: 1) "*As we all known*, ***Steve Jobs***", 2) "***Steve Jobs*** *was the co-founder of* ***Apple Inc***", 3) "***Apple Inc*** *which is a great company in America*".

Applying the multiple convolution kernels on embedding layer, we get a convoluted context matrix $\boldsymbol{C} = [\hat{\boldsymbol{c}}_1, \hat{\boldsymbol{c}}_2, \cdots, \hat{\boldsymbol{c}}_n]$. Let i_1, i_2 denote the index of the last token of first entity e_1 and the second entity e_i respectively, the context matrix can be segmented into three parts $\boldsymbol{C}_1 = [\hat{\boldsymbol{c}}_i]_{1 \le i \le i_1}$, $\boldsymbol{C}_2 = [\hat{\boldsymbol{c}}_i]_{i_1 < i \le i_2}$ and $\boldsymbol{C}_3 = [\hat{\boldsymbol{c}}_i]_{i_2 < i \le n}$. Using the piecewise max pooling (max-over-time) procedure on them, we generate the fixed size shared feature representation

$$f(V; \theta_s) = [max(\boldsymbol{C}_1), max(\boldsymbol{C}_2), max(\boldsymbol{C}_3)] \tag{1}$$

where V is the input embeddings described in section 2.1, θ_s denotes the parameters of shared feature extractor.

2.3 Relation classify loss

The source labeled data together with target unlabeled data are fed into PCNN to get the shared features $f(V;\theta_s)$, but only the shared features $f(V_s;\theta_s)$ from source domain are used to predict the relation type, there V_s denote the source domain data embedding. We use one hidden layer with tanh activation function followed by a softmax to produce the relation distribution:

$$p_{rel} = softmax(R(f(V_s;\theta_s);\theta_y)) \tag{2}$$

where θ_y denotes the parameters of the relation classify layer,R is the relation classify layer. The relation classify loss L_{rel} is defined as:

$$L_{rel} = -\frac{1}{S}\sum_{i=1}^{S}\sum_{j=1}^{M} y_{ij}\log p_{ij} \tag{3}$$

where S is the number of source domain data, M is the total number of relation types, y_{ij} is a binary number to indicate whether the example i has the relation j and p_{ij} is the probability of example i has the relation j.

2.4 Domain discriminator loss

Following previous work[7,8], we use adversarial training to learn the shared features of the source and target domain. All the shared features $f(V;\theta_s)$ will be feed into the domain discriminant layer to predict the domain to which the sample belongs. The domain discriminant layer includes one hidden layer with tanh activation function followed by a softmax:

$$p_{dom} = softmax(D(f(V;\theta_s);\theta_d)) \tag{4}$$

where θ_d is the parameters of domain discriminator layer. We use L_{dom} to represent the loss of domain classification:

$$L_{dom} = -\frac{1}{S+T}\sum_{i}^{S+T}(1-y_i)\log(1-p_i) + y_i\log p_i \tag{5}$$

where T is the number of target domain data, y_i indicates which domain the sample belongs, p_i is the probability of the sample belonging to source domain. To make the shared feature extractor extract shared features between domains, following previous work [7], we use the gradient reversed layer (GRL) to reverse the gradient of the parameters before the domain discriminator, then the forward and back propagation are formulated as follows:

$$GRL(x) = x \tag{6}$$

$$\frac{dGRL(x)}{dx} = -I \tag{7}$$

where I is an identity matrix. Using GRL, the parameters θ_s are optimized to maximize the L_{dom}, i.e., the domain discriminator can't distinguish which domain the features come from. Meanwhile, the parameters θ_d are trained to minimize the L_{dom}, which tends to correctly distinguish the domain, thus through adversarial training, the shared feature extractor will learn some shared features of source and target domain.

2.5 Cross-view adaptation loss

The model we described above can extract shared features of source and target domain, but it inevitable introduce some domain-specific features. In addition, we use some external features, such as chunks information, dependency parsing, and so on. These external features are obtained by other model, and may exist some errors. In order to solve the above problems, we apply cross-view training on the target domain.

First, we construct four restricted views of target domain data for cross-domain relation extraction task:

Entity pair mask (em) The effect of relation extraction depends on the context information near the entity pair, and the similar context information often has the same relation. For example, in the phrase "***Basra*** *is a port city in* ***Iraq***", "***Iraq***" and "***Basra***" have a relation of PART-WHOLE, and in the sentence "***Paris*** *is the most prosperous city in* ***France***", "***Paris***" and "***France***" are also have PART-WHOLE relation. In cross-domain relation extraction task, entities in different domains are different, but contexts often similar across domains. These different entities are domain-specific features that hurt the performance of relation extraction on the target domain. Based on this idea, we construct a restricted view of input data by masking the target domain entity pairs, leaving only the context information near entities. By feeding data from this view we get the relation distribution $p_\theta^{em}(y|x_i)$:

$$p_\theta^{em}(y|x_i^{em}) = softmax(R(f(x_i^{em};\theta_s);\theta_{em})) \tag{8}$$

where x_i^{em} is the entity pair mask on the target domain data. By matching the distribution of full-view data, the model learns some fixed context patterns that do not depend on entity pairs, this avoid introduce some domain-specific features to some extent. In the test stage, the model tends to predict relation as same as context-like example on the training set.

Target specific word mask (twm) Similar to the idea of entity pair mask, a more violent way is directly masking the words that only appeared in the target domain:

$$p_\theta^{twm}(y|x_i^{twm}) = softmax(R(f(x_i^{twm};\theta_s);\theta_{twm})) \tag{9}$$

where x_i^{twm} is the target specific word mask on target domain data. Experiments show that this view of input data also improved the cross-domain relation extraction.

Shortest dependency path mask (spm) Shortest dependency path is the shortest path between two entities in the dependency tree. In a sentence, the

information needed for extracting the relation between two entities is usually determined by the shortest path between the two entities in the dependency graph [4]. So we mask the words outside the shortest path, thus not only preserving the information needed for the relation extraction, but also removing some domain-specific information. The shortest dependency path mask can be formulated as:

$$p_\theta^{spm}(y|x_i^{spm}) = softmax(R(f(x_i^{spm};\theta_s);\theta_{spm})) \tag{10}$$

where x_i^{spm} is the shortest dependency path mask on the target domain data.

Embedding mask (embm) It seems less data efficient that masking word directly. So except for constructing the word level mask, we also explored the mask of the embedding layer. Among the features of our embedding layer, chunk and dependency path features are obtained from other trained models and therefore inevitable have some errors, entity type features are often domain-specific features. So we remove these embedding, leaving only word embedding and position embedding as input to the shared features extractor:

$$p_\theta^{embm}(y|x_i^{embm}) = softmax(R(f(x_i^{embm};\theta_s);\theta_{embm})) \tag{11}$$

where x_i^{embm} is the embedding mask on target domain data. Through the construction of this input view, the model will be more domain-general and have stronger Fault-tolerance.

Then we feed these restricted views of target domain data together with full views to the network. We use $p_\theta(y|x_i)$ to represent the relation distribution of full views described in (2), where x_i is the origin input sentence on the target domain. During training, we minimize the difference between $p_\theta(y|x_i)$ and $p_\theta^j(y|x_i)$. Specifically, we first get $p_\theta(y|x_i)$ by feeding full views of target domain data and fix it(i.e.,do not perform back-propagation) in every training batch, then use the relation distribution $p_\theta^j(y|x_i)$ of restricted views data to match it. We use KL divergence to measure the difference in data distribution, and let the relation distribution of restricted view data fit the distribution of the full-view data by minimizing L_{cv}:

$$L_{cv} = \frac{1}{T}\sum_{i=1}^{T}\sum_{j\in K} D(p_\theta(y|x_i), p_\theta^j(y|x_i^j)) \tag{12}$$

where D is the KL-divergence, $K = \{em, spm, twm, embm\}$ is the set of all input views.

During training, we jointly optimize all the loss function by minimize L_{loss}:

$$L_{loss} = L_{rel} + \alpha L_{dom} + \beta L_{cv} \tag{13}$$

We set $\alpha = 0.1$ and $\beta = 0.01$ through development set.

3 Experiments

3.1 Dataset

We use the English part of ACE 2005, which is a widely used dataset for cross-domain relational extraction. It covers six domains: Newswire (nw), Broadcast

Conversation (bc), Broadcast News (bn), Telephone Speech (cts), Usenet Newsgroups (un), and Weblogs (wl). Following previous work [9,17,19], we use nw and bn as the training set, half of bc as the development set, and the remaining half of bc as the test set. After processing, 43497 entity pairs were generated for training. Tabel 1 show the detailed statistics.

Table 1. ACE 2005 dataset statistics.

Domain	Total	Number of entities	No relation rate
bn+nw	43497	5442	91.6%
bc dev	7004	936	91.2%
bc test	8083	1107	91.1%
cts	15803	769	96.1%
wl	13882	2150	94.9%

3.2 Implementation Details

Following Fu and Grishman[7], we use the same settings for the embeddding layer. The pre-trained word embedding is 300 dimensions generated by word2vec. The embedding size of position/chunk/dependency path is 50. For each convolution layer of PCNN, the convolution step is set to 2, 3, 4, 5, and total filter numbers are 150. Our fixed sentence length is 155 which is the maximum of all sentence and the dropout rate is 0.5. All of the input views shared the same PCNN parameters, but the fully connected layer are different respectively, and we only use full-view fully connected layer for testing, the hidden size of fully connected layer and domain discrimination layer is 300. We use Adam for optimizing the model, the learning is 0.001 and will be halved every two epochs.

3.3 Baseline Models

We compare our proposed model to the following baseline models:

- **Log-linear** [13] This paper explored some neural network method for relation extraction, include CNN, bidirectional RNN, forward RNN, backward RNN and proposed a combined model called Log-linear. We compare with the single models rather than the combined models they proposed.
- **FCM & Hybrid FCM** [9] Feature-rich Compositional Embedding Model (FCM) is a model that combines both hand-crafted features with learned word embeddings. The Hybrid model combines the FCM with an existing log-linear model.
- **LRFCM** [19] Low-rank FCM (LRFCM) dramatically reduced the number of FCMs parameters and can scale to more features and more labels.
- **DANN** [7] This model is similar with our proposed model but only use domain adversarial training to extract shared features.

– **GSN** [17] This paper demonstrates that traditional method for cross-domain relation extraction inevitable introduced some domain-specific features. So they extract shared features and private features separately and maximize the difference in their distribution. They also use shared feature to perform relation extraction.

3.4 Evaluation and Analysis

We use macro F1-score to measure relation extraction performance. The result shows that our proposed baseline model PCNN+DANN achieved comparable performance to state-of-the-art models(Table 2). After using cross-view training with different restricted views we construct, our combine model outperform all the previous methods in the three domains of ACE2005 dataset. specifically, on bc domain, our model obtained 1% gains compared with PCNN+DANN model, and achieved comparable results to GSN model, but GSN is more complex and requires more time training. On cts domain, our model increased the F1-score by 3%, which is a significant improvement.

Table 2. relation extraction performance on different models(Macro F1-score %). *+em*,*+spm*,*+twm*,*+embm* mean adding entity type mask, shortest dependency path mask, target word mask and embedding mask respectively to the PCNN+DANN model. *Combine* is our final model, taking all input views into account.

Model	bc	cts	wl
Forword RNN	61.44	54.93	55.10
Backward RNN	60.82	56.03	51.78
Bidirectional RNN	63.07	56.47	53.65
CNN	63.26	55.63	53.91
FCM	61.9	52.93	50.36
Hybrid FCM	63.48	56.12	55.17
LRFCM	59.4	-	-
GSN	66.38	57.92	56.84
CNN+DANN	65.16	-	-
PCNN+DANN	65.78	58.56	56.62
+em	66.78	59.28	57.45
+spm	65.53	59.11	57.12
+twm	66.71	58.42	57.21
+embm	66.65	58.77	57.34
Combine	**66.81**	**60.88**	**57.62**

We also performed ablation experiments to explore the effects of different input views(Table 2). Among them, the entity pair mask has the greatest impact on the results, which also verifies our previous analysis. Some entities are domain-specific and have a negative effect on the results. Using the entity pair mask, to some extent, the shared feature extractor is prevented from introducing some domain-specific features so that the model can better transfer from source

domain to target domain. Target specific word mask have a very different effect on different domains. We calculated the proportion of shared words in different domains and found that when the number of these words decrease, the model gets less gain. We analyze that this may be due to more context information is lost when there are fewer shared words, which affects the relation extraction. This also explains that the data utilization efficiency is lower by directly masking the origin word. The embedding mask view compensates for this shortcoming. It used all words in a sentence as input, but only masked some features in the embedding layer. We also tried to mask some intermediate representations of the model, such as masking one output feature of PCNN's piece, but the experiments show that this will make the result worse, which is obvious because it masked most of the context near entity. Like Clark and Manning [6], we also use LSTM as a relation extractor and construct forward and backward restricted views, but the results are also worse. We suppose that sequence tagging task relies more on sequential information but relation extraction task does not.

4 Related work

Most existing cross-domain relation extraction studies focus on learning a shared feature representation between source and target domains. [15] use manually constructed features such as word clustering and tree kernels to improve the ability of domain adaptation. Some neural network based methods such as CNN, RNN, or compositional models also significantly advance the performance of cross-domain relation extraction [9,13,14,19]. Recently some methods based on adversarial training have also been applied to cross-domain relation extraction. [7] and [16] used adversarial training to project the common features of source and target domains into one feature space, and then use these features to identify relations. [5,11,17] respectively extract private and shared features based on the domain separation network [3] to avoid introduce private features into shared feature space. These methods all find the connection between different domains from a full view, and intuitively, if observing the data from different part views, we can more clearly discover the shared features of different domains.

Cross-view training [6] is a semi-supervised learning algorithm. It constructs some restricted data input views and use them to improve the intermediate representation of the model. A very similar algorithm is multi-view learning, which divides features into sub-features [18], and uses co-training algorithm [2] to train two separate models. On unlabeled data, each one acts as a teacher for the other model. While in cross-view training, there is only one model and use different views of unlabeled data to improve the shared model. Cross-view training has achieved good results in some tasks such as sequence labeling and machine translation, but it has not been applied to domain adaptation. We used similar training methods and constructed some novel input views for relation extraction. Experiments show this method is suitable for cross-domain relation extraction.

5 Conclusion

We proposed a cross-view training based method for cross-domain relation extraction, and not required any labeled data in target domain. Specifically, We innovatively constructed some data input views for cross-domain relation extraction, and experiments demonstrated that by changing the data input views, such as masking some domain-specific information, the model can learn the shared features more effectively. To the best of our knowledge, this is the first study to apply cross-view training to cross-domain relation extraction. We hope that this method can be extended to other domain adaptation tasks in future research.

6 Acknowledgements

This work was supported by National Key R&D Program of China (2017YFB0802703) and National Natural Science Foundation of China (61602052).

References

1. Blitzer, J., McDonald, R., Pereira, F.: Domain adaptation with structural correspondence learning. In: Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing. pp. 120–128. EMNLP '06, Association for Computational Linguistics, Stroudsburg, PA, USA (2006)
2. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Proceedings of the Eleventh Annual Conference on Computational Learning Theory. pp. 92–100. COLT' 98, ACM, New York, NY, USA (1998). https://doi.org/10.1145/279943.279962
3. Bousmalis, K., Trigeorgis, G., Silberman, N., Krishnan, D., Erhan, D.: Domain separation networks. In: Lee, D.D., Sugiyama, M., Luxburg, U.V., Guyon, I., Garnett, R. (eds.) Advances in Neural Information Processing Systems 29, pp. 343–351. Curran Associates, Inc. (2016)
4. C. Bunescu, R., J. Mooney, R.: A shortest path dependency kernel for relation extraction. (01 2005). https://doi.org/10.3115/1220575.1220666
5. Chen, X., Shi, Z., Qiu, X., Huang, X.: Adversarial multi-criteria learning for Chinese word segmentation. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 1193–1203. Association for Computational Linguistics, Vancouver, Canada (Jul 2017). https://doi.org/10.18653/v1/P17-1110
6. Clark, K., Luong, T., Manning, C.D., Le, Q.V.: Semi-supervised sequence modeling with cross-view training (2018)
7. Fu, L., Nguyen, T.H., Min, B., Grishman, R.: Domain adaptation for relation extraction with domain adversarial neural network. In: Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers). pp. 425–429. Asian Federation of Natural Language Processing, Taipei, Taiwan (Nov 2017)
8. Ganin, Y., Lempitsky, V.: Unsupervised domain adaptation by backpropagation. In: Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37. pp. 1180–1189. ICML'15, JMLR.org (2015)

9. Gormley, M.R., Yu, M., Dredze, M.: Improved relation extraction with feature-rich compositional embedding models. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp. 1774–1784. Association for Computational Linguistics, Lisbon, Portugal (Sep 2015). https://doi.org/10.18653/v1/D15-1205
10. Jiang, J., Zhai, C.: Instance weighting for domain adaptation in nlp (01 2007)
11. Liu, P., Qiu, X., Huang, X.: Adversarial multi-task learning for text classification. arXiv preprint arXiv:1704.05742 (2017)
12. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. Proceedings of Workshop at ICLR **2013** (01 2013)
13. Nguyen, T.H., Grishman, R.: Combining Neural Networks and Log-linear Models to Improve Relation Extraction. arXiv e-prints (Nov 2015)
14. Nguyen, T.H., Grishman, R.: Employing word representations and regularization for domain adaptation of relation extraction. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). pp. 68–74. Association for Computational Linguistics, Baltimore, Maryland (Jun 2014). https://doi.org/10.3115/v1/P14-2012
15. Plank, B., Moschitti, A.: Embedding semantic similarity in tree kernels for domain adaptation of relation extraction. vol. 1, pp. 1498–1507 (08 2013)
16. Rios, A., Kavuluru, R., Lu, Z.: Generalizing biomedical relation classification with neural adversarial domain adaptation. Bioinformatics **34 17**, 2973–2981 (2018)
17. Shi, G., Feng, C., Huang, L., Zhang, B., Ji, H., Liao, L., Huang, H.: Genre separation network with adversarial training for cross-genre relation extraction. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. pp. 1018–1023. Association for Computational Linguistics, Brussels, Belgium (Oct-Nov 2018)
18. Xu, C., Tao, D., Xu, C.: A survey on multi-view learning. CoRR **abs/1304.5634** (2013)
19. Yu, M., R. Gormley, M., Dredze, M.: Combining word embeddings and feature embeddings for fine-grained relation extraction. pp. 1374–1379 (01 2015). https://doi.org/10.3115/v1/N15-1155
20. Zeng, D., Liu, K., Chen, Y., Zhao, J.: Distant supervision for relation extraction via piecewise convolutional neural networks. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp. 1753–1762. Association for Computational Linguistics, Lisbon, Portugal (Sep 2015). https://doi.org/10.18653/v1/D15-1203