CROSS-DOMAIN SENTIMENT CLASSIFICATION USING DEEP LEARNING APPROACH

Miao Sun[1], Qi Tan[2], Runwei Ding[*3], Hong Liu[4]

[1,2] South China Nomal University, Guangzhou 510631, China
[3] Peking University Shenzhen Graduate School, Shenzhen 518055, China
[4] Peking University, Beijing 100087, China
sunmiao502@126.com, tanqi@scnu.edu.cn, dingrunwei@pkusz.edu.cn, hongliu@pku.edu.cn

Abstract: Deep learning, as a new unsupervised leaning algorithm, has strong capabilities to learn data representations. Previous work has shown that new features learned by deep learning algorithm help to improve the accuracy of cross-domain classification. In this paper, we firstly propose a modified version of marginalized stacked denoising autoencoders (mSDA). We call it mSDA++ algorithm, which can learn excellent and low-dimensional features for training classifier. In addition, we combine mSDA with EASYADAPT algorithm to further improve the accuracy of cross-domain classification. Then we use SVM, mSDA, mSDA++, and EA+mSDA algorithms to do the cross-domain sentiment classification experiments on Amazon benchmark dataset. The results show that EA+mSDA algorithm attains the best accuracy. Besides, the mSDA++ algorithm can accelerate the subsequent calculation and reduce the data storage space.

Keywords: Text sentiment classification; Deep learning; Cross-domain; dimension reduction; feature augment

1 Introduction

Lack of lots of labeled training data is an important problem in text sentiment classification. However, there are a lot of related data in other domain. It is meaningful to train a cross-domain classifier to transfer these data to the domain which is lack of labeled samples. However, classifiers trained in one domain do not perform well in others [1]. The major obstacle in cross-domain classification is that data in the source and the target domain do not hold the independent and identically distributed assumption. In order to attain better accuracy of cross-domain classification, some methods are proposed to alleviate the difference between source and target domain distributions, such as instance reweighting [2, 3], structural correspondence learning [4, 5], etc.

Deep learning [6], as a new unsupervised leaning algorithm, has excellent ability to learn feature representations. It is of great concern in academia in recent years. Vincent et al. [7] proposed stacked denoising autoencoders (SDA) to learn robust feature representations and applied it to classification of handwritten digits. Lee et al. [8] applied convolutional deep belief net-works to audio classification. They used outputs of intermediate layers of the networks as features to train SVM classifier. Glorot et al. [9] applied SDA to text sentiment classification and attained good performance. However, SDA algorithm training with gradient descent or other optimization algorithms is slow and dependent on the initialization. For this problem, Chen et al. [10] proposed mSDA algorithm, which preserves strong feature learning capabilities without using optimization algorithm to learn the parameters. This algorithm performed well on cross-domain classification problem.

In this paper, we want to use deep learning approach to reduce the difference of data distribution and attain better accuracy of cross-domain sentiment classification. We propose a modified version of mSDA (mSDA++), which can learn excellent and low-dimensional features for training classifier. The mSDA++ algorithm reduces the time cost and space cost of subsequent calculation but also preserves strong feature learning capabilities. In addition, Chen et al. didn't consider that words in different domains may express different sentiment, but only used mSDA to learn new data representations for domain adaptation. For this problem, we combine mSDA with a domain adaptation approach (EA) [11] to further improve the accuracy of cross-domain sentiment classification.

2 Notation and background

In this paper, we apply deep learning approach to cross-domain sentiment classification problem. At first, we introduce some notation to facilitate discussion. Assume that data is taken from source domain S and target domain T. Denote by x the input space and by y the output space. We suppose the source domain dataset is $\mathrm{D_S}=\{(\mathrm{x_{s_i}},\mathrm{y_{s_i}})\}_{i=1}^{ns}$ and the target domain dataset is $\mathrm{D_T}=\{(\mathrm{x_{T_i}},\mathrm{y_{T_i}})\}_{i=1}^{n_T}$ where $\mathrm{x}\epsilon\mathrm{R}^d$ and $\mathrm{y}\in\{-1,+1\}$. Our goal is to learn classifier $\mathrm{h}:\mathrm{x}\rightarrow\mathrm{y}$ to predict the labels of data from target domain T with the help of data from source domain S. In sentiment classification problem, we hold the opinion that text expresses positive sentiment when $\mathrm{y}=+1$, and expresses negative sentiment when $\mathrm{y}=-1$.
Marginalized stacked denoising autoencoders (mSDA) is a modified version of stacked denoising autoenders (SDA). The SDA stacks several denoising autoenders (DA) into deep learning architecture to learn higher-level representations of inputs. Denoising

autoenders are one-layer neural networks trained to reconstruct input data from partial corrupted data. The hidden outputs of tth DA are used as inputs to train $t+1$th DA, and the training is layer by layer. In SDA, nonlinear function $f(\cdot)$ and the reconstruction weight W are learned together with gradient descent algorithm, which makes the training procedure very slow. In contrast to SDA, mSDA does not use optimization algorithm to learn parameters. The mSDA stacks several mDAs by feeding the output of tth mDA(after the nonlinearity function) as the input of $t+1$th mDA. Differing from DA, mDA doesn't have hidden nodes. The mDA just uses single mapping $W: R^d \rightarrow R^d$ to reconstruct the corrupted inputs $\tilde{x}$ and uses multiple different corrupted versions of the original inputs x. It aims to minimize the squared reconstruction loss defined by eq. (1).

$$L_{sq}(W) = \frac{1}{2mn}\sum_{j=1}^{m}\sum_{i=1}^{n}\left\|x_i - W\tilde{x}_{ij}\right\|^2 \qquad (1)$$

Where $\tilde{x}_{ij}$ represents the jth corrupted version of the original input x_i.

The reconstruction weight W can be computed in closed-form [12]. After W computed, nonlinearity function is applied on the output of mDA. Figure 1 shows the process of training mDA.

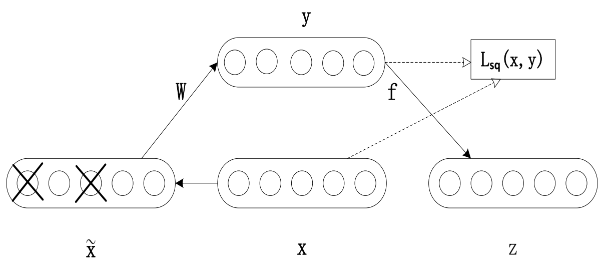

Figure 1 An example x is corrupted to $\tilde{x}$. The mDA maps it to y in order to reconstruct x. Then nonlinear function f is applied on y to generate nonlinear features

Once mSDA is trained, its outputs combined with the original inputs form the new features used for training classifier. As mSDA doesn't use optimization algorithm, it needs less training time than SDA but still preserves strong learning capabilities.

3 Proposed new methods

Adding features learned by mSDA algorithm to original features helps to improve accuracy of cross-domain classification. But new features learned by mSDA certainly have the same dimension as original features. In this section, we firstly propose a modified version of mSDA, which can learn excellent and low-dimensional features. Furthermore, considering that words in different domains may express different sentiment, we combine mSDA with a domain adaptation approach (EA) to attain better accuracy of cross-domain sentiment classification.

3.1 mSDA++: mSDA with dimension reduction

As mentioned，mDA does not have hidden nodes. In mSDA, a single mapping $W: R^d \rightarrow R^d$ is used to reconstruct the corrupted input data, and nonlinearity is injected after W are computed. Therefore, new features learned by mSDA algorithm certainly have the same dimension as original features. If we could learn new features with lower dimension, then the time cost and space cost of the subsequent training of classifier could be reduced. To achieve this goal, we add a denoising autoencoder(DA) after the last layer of mSDA. The mSDA uses greedy layer-wise training method to stack mDAs into a deep architecture. Denote the original input as $h_0 = x$ and the output of the t th mDA as $h_t = f(W_t^T h_{t-1})$. Assume that mSDA has n layers, then the outputs of the last layer of mSDA is $h_n = f(W_n^T h_{n-1})$. Then we use h_n as the input data to train the denoising autoencoder that is added after the last layer of mSDA. The denoising autoencoder tries to learn a function $h_{W,\tilde{h}_n}(\tilde{h}_n) \approx h_n$ so the output is similar to the input. As DA has hidden nodes, assume that the number of its hidden nodes is less than the number of nodes of input layer and output layer, then the outputs of its hidden layers are the new features with lower dimension (See Figure 2). Linear function is used as the activation function of this denoising autoencoder. We refer to this algorithm as mSDA++.

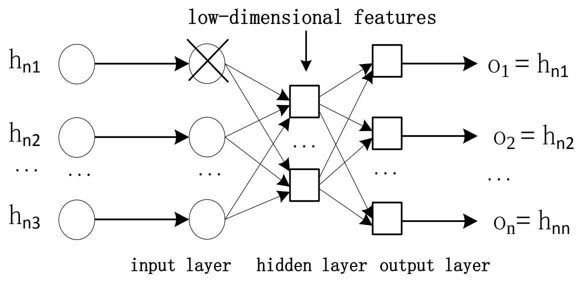

Figure 2 The DA added after the last layer of mSDA

In this paper, we use tanh(x) as nonlinear squashing-function applied on the output of each mDA. After learning the new and low-dimensional features by mSDA++, we add these new features to original features to train the classifier.

3.2 EA+mSDA

As mentioned, Chen et al. have applied mSDA to cross-domain classification problem. In fact, many words in different texts express different sentiment. However, Chen et al. didn't consider this problem when doing the classification experiments. They only used mSDA algorithm to learn new data representations for training the classifier. For the above problem, Hal et al. have proposed a domain adaption approach termed EASYADAPT (EA). By simply augmenting the feature space, EA forces the learning algorithm to do the domain adaptation. We aim to combine mSDA with EA to attain better accuracy of cross-domain classification.

Firstly, let's review the EA algorithm. Denote by $x \in R^d$ the input space and by $y \in \{-1, +1\}$ the output space.

The EA algorithm augments feature space by mappings $\Phi^S, \Phi^T: x \rightarrow \check{x}$. Φ^S and Φ^T work on source domain and target domain respectively. The mappings are defined by eq. (2).

$$\Phi^s = \langle x, x, \mathbf{0} \rangle$$
$$\Phi^T = \langle x, \mathbf{0}, x \rangle$$
$$\text{where } \mathbf{0} = \langle 0,0,\cdots,0 \rangle \in R^d \quad (2)$$

After mapping, the feature space turns into $\check{x} \subset R^{3d}$. The augmented space consists of three parts. The first d-dimensional part indicates commonality between the source and target domains. The second and third d-dimensional parts correspond to source and target domains respectively. For k domains, the augmented space is expanded to $\check{x} \subset R^{(k+1)d}$. We could equally use $\Phi^s = \langle x, x \rangle$ and $\Phi^T = \langle x, \mathbf{0} \rangle$ to replace eq. (2).

In this paper, we use two steps to combine mSDA with EA:

1. Augment original feature space x by EA, $x \rightarrow \check{x}$.

2. Transform $\check{x}$ into new feature space by mSDA, $\check{x} \rightarrow \check{x}'$.

After these two steps, we combines $\check{x}'$ with original input x to form the new features that used for training classifier.

4 Experiments

In this section, we apply mSDA++ and EA+mSDA algorithms to cross-domain sentiment classification problem and evaluate the effectiveness.

4.1 Tasks

Amazon reviews benchmark dataset [4] is used in our experiments. The dataset includes product reviews of four domains: electronics (E), toys (T), health products (H) and kitchen appliances (K). Each domain contains 2000 labeled data and 2000 unlabeled data used for our experiments. There are twelve transfer tasks: K→E，T→E，H→E，K→H，T→H，E→H，H→K，T→K，E→K，K→T，H→T，E→T. We apply SVM(baseline), mSDA, mSDA++, and EA+mSDA algorithms to do the cross-domain sentiment classification experiments to the transfer tasks.

We use transfer loss [10] to evaluate the result of transfer tasks. Denote by $e(T,T)$ in-domain error and by $e(S,T)$ transfer error. The notation $e_b(S,T)$ represents in-domain error of baseline. The in-domain error $e(T,T)$ means the classification error of the classifier that is trained on the same domain. The transfer error $e(S,T)$ means the error of the classifier that is trained on different domain. Transfer loss is defined by eq. (3)

$$\text{transfer loss} = e(S,T) - e_b(S,T) \qquad (3)$$

In cross-domain classification problem, the lower transfer loss the better. If transfer loss was negative, it would mean that classifier trained on data from other domain attains higher accuracy than one trained on the data from original domain. In other words, cross-domain classification even attains higher accuracy than in-domain classification.

4.2 Cross-domain sentiment classification experiment based on mSDA++ algorithm

In this experiment, we combine new features learned by mSDA++ algorithm with original features and then use these features to train a linear SVM classifier. As baseline, we use labeled data from source domain to train linear SVM classifier and then test on unlabeled data from target domain. In order to show the effectiveness of mSDA++ and EA+mSDA algorithms, we also train classifier with the features that consist of original features and new features learned by mSDA.
Two of the important parameters in mSDA and mSDA++ algorithms are noise p and layer l. Noise p is used for corrupting input data. In the experiment, we set each feature to 0 with possibility $p \geq 0$ to get corrupted version of original input data. In addition, we use 3-fold cross-validation on different parameters to choose optimal parameters. The training of linear SVM classifier is performed by using MATLAB LIBSVM toolbox [13]. The implement of this experiment is on the computer with 2.50GHz CPU. The following notations are used to describe the experiment result.

$\mathbf{d_1}$ Represents dimension of original features.

$\mathbf{d_2}$ Represents dimension of new features learned by mSDA++

$\mathbf{t_1}$ Represents the time of training classifier with original features

$\mathbf{t_2}$ Represents the time of training classifier with new features learned by mSDA++

$\mathbf{t_3}$ Represents the time of dimension reduction in mSDA++ algorithm

$\mathbf{t_4}$ represents the sum of $\mathbf{t_2}$ and $\mathbf{t_3}$.

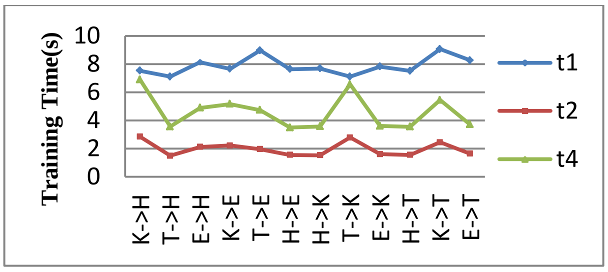

Figure 3 training time of two methods

Figure 3 describes the average training time of the two algorithms. We can find that the time of training classifier with new features learned by mSDA++ is less than the time of training classifier with original features. As Table I shows, the averaged percentage of the time of training classifier with new features learned by mSDA++ and the time of training classifier with original features is about 25%. If we considered the time of dimension reduction in mSDA++ algorithm, then the percentage would turn into 57.63%. Therefore, efficiency of training classifier is improved about 42.3% on average.

Table I The objective evaluation of two methods

Transfer tasks	d_1	d_2	t_2/t_1	t_4/t_1
K→H	1000	300	37.9%	91.6%
T→H	1000	100	21.0%	50.1%
E→H	1000	150	26.0%	60.0%
K→E	1000	200	29.0%	57.9%
T→E	1000	150	21.9%	52.7%
H→E	1000	100	20.3%	45.7%
H→K	1000	100	19.9%	46.5%
T→K	1000	300	39.2%	92.4%
E→K	1000	100	20.6%	46.1%
H→T	1000	100	20.6%	47.3%
K→T	1000	150	27.0%	60.1%
E→T	1000	100	19.9%	44.9%
average	**1000**	**154**	**25.1%**	**57.6%**

In addition, the result reported in Figure 4 shows the averaged transfer loss of (baseline) 5.39%, (five-layers mSDA) -1.30%, (six-layers mSDA++) -0.85%, which implies that mSDA++ and mSDA both attain much better accuracy of cross-domain classification than baseline does. Also we find that mSDA++ performs a little worse than SDA but reduces nearly half the training time of SVM classifier.

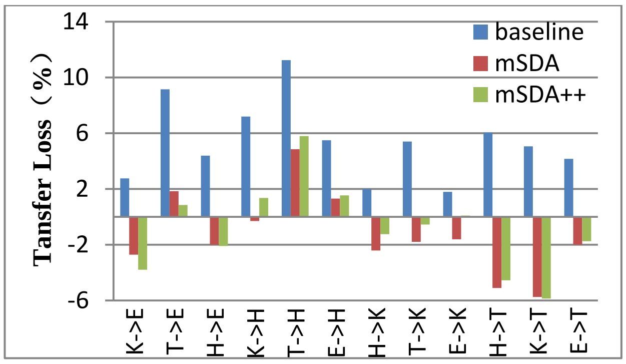

Figure 4 Transfer loss of different methods

What's more, the new feature matrix is ${d_1}/{d_2}(d_2 < d_1)$ times smaller than original feature matrix. According to the experimental results, we think mSDA++ can accelerate the speed of the subsequent calculation and reduce the data storage space but still preserves strong feature learning capabilities.

4.3 Cross-domain sentiment classification experiment based on EA+mSDA algorithm

In this experiment, we combine new features learned by EA+mSDA algorithm with original features and then use these features to train a SVM classifier for cross-domain classification. To show the effectiveness of EA+mSDA, we compare it with the other two methods, baseline and mSDA.

Table II The transfer loss of different methods

Transfer tasks	Transfer loss		
	baseline	mSDA	EA+mSDA
K→H	2.75%	-2.70%	-4.25%
T→H	9.15%	1.85%	**-7.00%**
E→H	4.40%	-2.00%	-2.60%
K→E	7.20%	-0.30%	-0.20%
T→E	11.25%	4.85%	3.95%
H→E	5.50%	1.30%	1.55%
H→K	2.00%	-2.40%	-3.55%
T→K	5.40%	-1.80%	0.05%
E→K	1.80%	-1.60%	-1.70%
H→T	6.05%	-5.10%	-6.05%
K→T	5.05%	-5.75%	-6.30%
E→T	4.15%	-2.00%	-4.60%
average	**5.39%**	**-1.30%**	**-2.56%**

Table II shows transfer loss of cross-domain sentiment classification by different methods. The transfer loss reported in Table II shows us that the averaged transfer loss of mSDA is -1.30% and EA+mSDA is -2.56%. As mentioned, the lower transfer loss the better. Therefore, EA+mSDA achieves higher accuracy than mSDA does on average. Table III shows us the accuracy of cross-domain classification of different methods. From Table III , we can find that some transfer tasks, for example, T→H, have attain much higher accuracy when using EA+mSDA. Above all, we think EA+mSDA further improve the accuracy of cross-domain sentiment classification.

Table III The accuracy of different methods

Transfer tasks	Accuracy of classification		
	baseline	mSDA	EA+mSDA
K→H	74.75%	80.20%	81.75%
T→H	68.35%	75.65%	**84.50%**
E→H	73.10%	79.50%	80.10%
K→E	73.80%	81.30%	81.20%
T→E	69.75%	76.15%	77.05%
H→E	75.50%	79.70%	79.45%
H→K	76.30%	80.70%	81.85%
T→K	72.90%	80.10%	78.25%
E→K	76.50%	79.90%	80.00%
H→T	71.25%	82.40%	83.35%
K→T	72.25%	83.05%	83.60%
E→T	73.15%	79.30%	81.90%
average	**73.13%**	**79.83%**	**81.08%**

According to experimental results, we can find that deep learning algorithms have strong learning capabilities to learn new data representations. New features learned by deep learning algorithm help to improve the accuracy of cross-domain sentiment classification.

5 Conclusions

In this paper, we firstly propose a modified version of mSDA algorithm (mSDA++). In addition, considering that words in different domains may express different sentiment, we combine mSDA with EASYADAPT algorithm (EA+mSDA). Then we apply SVM (baseline), mSDA, mSDA++ and EA+mSDA algorithms on cross-domain sentiment classification of product reviews.

The experimental results show us that EA+mSDA algorithm attains the best accuracy of classification. Besides, mSDA++ algorithm can accelerate the subsequent calculation and reduce the data storage space but also preserves strong feature learning capabilities. From the results, we feel that deep learning has very strong capabilities to learn data representations. We would conduct more research on deep learning in future.

References

[1] Aue, A. and Gamon, M. Customizing sentiment classifiers to new domains: A case study. In Proceedings of recent advances in natural language processing, pp. 2-1, 2005.

[2] Huang, J., Smola, A.J., Gretton, A., Borgwardt, K. M., and Scholkopf, B. Correcting Sample Selection Bias by Unlabeled Data. In NIPS 19, pp. 601–608, 2007.

[3] Kavukcuoglu K, Ranzato, M.A., Fergus, R., and Le-Cun, Y. Kavukcuoglu K, Ranzato M, Fergus R, et al. Learning invariant features through topographic filter maps. In CVPR 2009. IEEE Conferenceon, pp.1605–1612, 2009.

[4] Blitzer, J., McDonald, R., and Pereira, F. Domain adaptation with structural correspondence learning. In Proceedings of the 2006 Conference on EMNLP, pp. 120–128, 2006.

[5] Mansour, Y., Mohri, M., & Rostamizadeh, A. Domain adaptation with multiple sources. In NIPS, 2009.

[6] Hinton GE and Salakhutdinov R R. Reducing the dimensionality of data with neural networks. Science, pp. 504-507, 2006.

[7] Vincent P, Larochelle H, Lajoie I, et al. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. The Journal of Machine Learning Research, 2010.

[8] Lee H, Pham P, Largman Y, et al. Unsupervised feature learning for audio classification using convolutional deep belief networks. In NIPS, 2009.

[9] Glorot X, Bordes A, Bengio Y. Domain adaptation for large-scale sentiment classification: A deep learning approach. In ICML, pp:513-520, 2011.

[10] Chen M, Xu Z, Weinberger K, et al. Marginalized denoising autoencoders for domain adaptation[J]. arXiv preprint arXiv, 2012.

[11] Daumé III H. Frustratingly easy domain adaptation. In ACL, pp: 256-263, 2007.

[12] Bishop, Christopher. Pattern Recognition and Machine Learning. Springer, 2006.

[13] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2001.