Automatic Features for Essay Scoring – An Empirical Study

Fei Dong and **Yue Zhang**
Singapore University of Technology and Design
fei_dong@mymail.sutd.edu.sg and yue_zhang@sutd.edu.sg

Abstract

Essay scoring is a complicated processing requiring analyzing, summarizing and judging expertise. Traditional work on essay scoring focused on automatic handcrafted features, which are expensive yet sparse. Neural models offer a way to learn syntactic and semantic features automatically, which can potentially improve upon discrete features. In this paper, we employ convolutional neural network (CNN) for the effect of automatically learning features, and compare the result with the state-of-art discrete baselines. For in-domain and domain-adaptation essay scoring tasks, our neural model empirically outperforms discrete models.

1 Introduction

Automatic essay scoring (AES) is the task of building a computer-based grading system, with the aim of reducing the involvement of human raters as far as possible. AES is challenging since it relies not only on grammars, but also on semantics, discourse and pragmatics. Traditional approaches treat AES as a classification (Larkey, 1998; Rudner and Liang, 2002), regression (Attali and Burstein, 2004; Phandi et al., 2015), or ranking classification problem (Yannakoudakis et al., 2011; Chen and He, 2013), addressing AES by supervised learning. Features are typically bag-of-words, spelling errors and lengths, such word length, sentence length and essay length, etc. Some grammatical features are considered to assess the quality of essays (Yannakoudakis et al., 2011). A drawback is feature engineering, which can be time-consuming, since features need to be carefully handcrafted and selected to fit the approriate model. A further drawback of manual feature templates is that they are sparse, instantiated by discrete pattern-matching. As a result, parsers and semantic analyzers are necessary as a preprocessing step to offer syntactic and semantic patterns for feature extraction. Given variable qualities of student essays, such analyzers can be highly unreliable.

Neural network approaches have been shown to be capable of inducing dense syntactic and semantic features automatcially, giving competitive results to manually designed features for several tasks (Kalchbrenner et al., 2014; Johnson and Zhang, 2014; dos Santos and Gatti, 2014). In this paper, we empirically investigate a neural network method to learn features automatically for AES, without the need of external pre-processing. In particular, we build a hierarchical CNN model, with one lower layer representing sentence structures and one upper layer representing essay structure based on sentence representations. We compare automatically-induced features by the model with state-of-art baseline handcrafted features. Empirical results show that neural features learned by CNN are very effective in essay scoring task, covering more high-level and abstract information compared to manual feature templates.

2 Related Work

Following the first AES system Project Essay Grade (PEG) been developed in 1966 (Page, 1994), a number of commercial systems have come out, such as IntelliMetric 2, Intelligent Essay Assessor (IEA) (Foltz et al., 1999) and e-rater system (Attali and Burstein, 2004). The e-rater system now plays a

Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 1072–1077,
Austin, Texas, November 1-5, 2016.

second human rater's role in the Test of English as a Foreign Language (TOEFL) and Graduate Record Examination (GRE). The e-rater extracts a number of complex features, such as grammatical error and lexical complexity, and uses stepwise linear regression. IEA uses Latent Semantic Analysis (LSA) (Landauer et al., 1998) to create semantic vectors for essays and measure the semantic similarity between the vectors.

In the research literature, Larkey (1998) and Rudner and Liang (2002) treat AES as classification using bag-of-words features. Other recent work formulates the task as a preference ranking problem (Yannakoudakis et al., 2011; Chen and He, 2013). Yannakoudakis et al. (2011) formulated AES as a pairwise ranking problem by ranking the order of pair essays based on their quality. Features consist of word, POS n-grams features, complex grammatical features and so on. Chen and He (2013) formulated AES into a listwise ranking problem by considering the order relation among the whole essays and features contain syntactical features, grammar and fluency features as well as content and prompt-specific features. Phandi et al. (2015) use correlated Bayesian Linear Ridge Regression (cBLRR) focusing on domain-adaptation tasks. All these previous methods use discrete handcrafted features.

Recently, Alikaniotis et al. (2016) also employ a neural model to learn features for essay scoring automatically, which leverages a score-specific word embedding (SSWE) for word representations and a two-layer bidirectional long-short term memory network (LSTM) to learn essay representations. Alikaniotis et al. (2016) show that by combining SSWE, LSTM outperforms traditional SVM model. On the other hand, using LSTM alone does not give significantly more accuracies compared to SVM. This conforms to our preliminary experiments with the LSTM structure. Here, we use CNN without any specific embeddings and show that our neural models could outperform baseline discrete models on both in-domain and cross-domain senarios.

CNN has been used in many NLP applications, such as sequence labeling (Collobert et al., 2011) , sentences modeling (Kalchbrenner et al., 2014), sentences classification (Kim, 2014), text categorization (Johnson and Zhang, 2014; Zhang et al., 2015) and sentimental analysis (dos Santos and Gatti, 2014), etc. In this paper, we explore CNN representation ability for AES tasks on both in-domain and domain-adaptation settings.

Feature Type	Feature Description
Length	Number of characters, words, sentences, etc.
POS	Relative and absolute number of bad POS n-grams
Prompt	Relative and absolute number of words and their synonyms in the essay appearing in the prompt
Bag-of-words	Count of useful unigrams and bigrams (unstemmed, stemmed and spell corrected)

Table 1: Feature description used by EASE.

3 Baseline

Bayesian Linear Ridge Regression (BLRR) and Support Vector Regression (SVR) (Smola and Vapnik, 1997) are chosen as state-of-art baselines. Feature templates follow (Phandi et al., 2015), extracted by EASE[1], which are briefly listed in Table 1. "Useful n-grams" are determined using the Fisher test to separate the good scoring essays and bad scoring essays. Good essays are essays with a score greater than or equal to the average score, and the remainder are considered as bad scoring essays. The top 201 n-grams with the highest Fisher values are chosen as the bag of features and these top 201 n-grams constitute useful n-grams. Correct POS tags are generated using grammatically correct texts, which is done by EASE. The POS tags that are not included in the correct POS tags are treated as bad POS tags, and these bad POS tags make up the "bad POS n-grams" features.

The features tend to be highly useful for the in-domain task since the discrete features of same prompt data share the similar statistics. However, for different prompts, features statistics vary significantly. This raises challenges for discrete feature patterns.

ML-ρ (Phandi et al., 2015) was proposed to address this issue. It is based on feature augmentation, incorporating explicit correlation into augmented feature spaces. In particular, it expands baseline feature vector $\mathbf{x}$ to be $\Phi^s(\mathbf{x}) = (\rho\mathbf{x}, (1-\rho^2)^{1/2}\mathbf{x})$ and $\Phi^t(\mathbf{x}) = (\mathbf{x}, \mathbf{0_p})$ for source and target domain data

[1] https://github.com/edx/ease

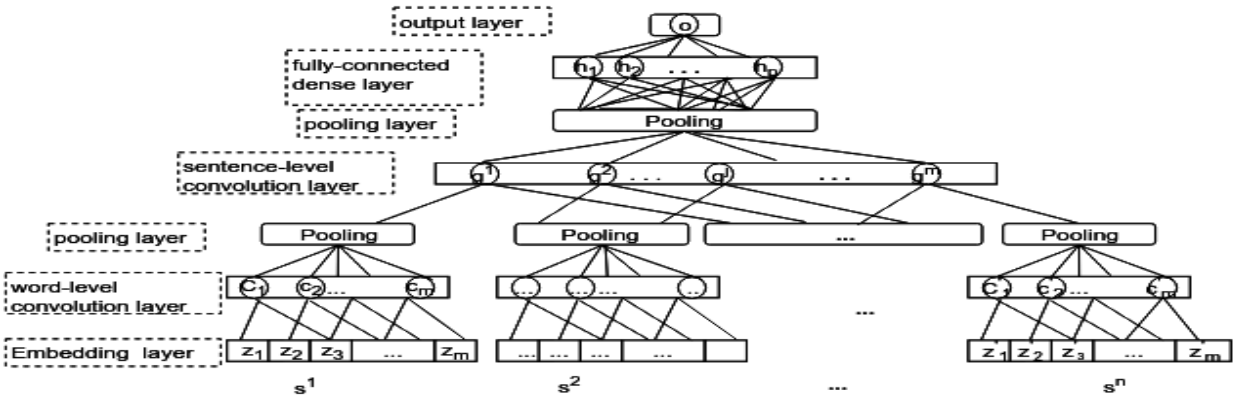

Figure 1: Hierarchical CNN structure

in R^{2p} respectively, with ρ being the correlation between source and target domain data. Then BLRR and maximum likelihood estimation are used to the optimize correlation. All the baseline models require POS-tagging as a pre-processing step, extracting syntactic features based on POS-tags.

4 Model

Word Representations We use word embedding with an embedding matrix $E_w \in R^{d_w \times V_w}$ where d_w is the embedding dimension, and V_w represents words vocabulary size. A word vector z_i is represented by $z_i = E_w w_i$ where w_i is the i-th word in a sentence. In contrast to the baseline models, our CNN model does not rely on POS-tagging or other pre-processing.

CNN Model We take essay scoring as a regression task and employ a two-layer CNN model, in which one convolutional layer is used to extract sentences representations, and the other is stacked on sentence vectors to learn essays representations. The architecture is depicted in Figure 1. Given an input sentence z_1, z_2, ..., z_n, a convolution layer with a *filter* $\mathbf{w} \in R^{h \times k}$ is applied to a window of h words to produce n-grams features. For instance, a feature c_i is generated from a window of words $z_{i:i+h-1}$ by $c_i = f(\mathbf{w} \cdot z_{i:i+h-1} + b)$, $b \in R$ is the bias term and f is the non-linear activation function rectified linear unit (ReLU).

The filter is applied to the all possible windows in a sentence to produce a feature map $\mathbf{c} = [c_1, c_2, ..., c_{m-h+1}]$. For $\mathbf{c}^j$ of the j-th sentence in an essay, max-pooling and average pooling function are used to produce the sentence vector $s^j = max\{\mathbf{c}^j\} \oplus avg\{\mathbf{c}^j\}$. The second convolutional layer takes s^1, s^2,..., s^n as inputs, followed by pooling layer (max-pooling and average-pooling) and a fully-connected hidden layer. The hidden layer directly connects to output layer which generates a score.

Set	#Essays	Genre	Avg Len.	Range	Med.
1	1783	ARG	350	2-12	8
2	1800	ARG	350	1-6	3
3	1726	RES	150	0-3	1
4	1772	RES	150	0-3	1
5	1805	RES	150	0-4	2
6	1800	RES	150	0-4	2
7	1569	NAR	250	0-30	16
8	723	NAR	650	0-60	36

Table 2: Details of the ASAP data; the last two columns are score range and median scores. For genre, ARG specifies *argumentative* essays, RES means *response* essays and NAR denotes *narrative* essays.

5 Experiments

5.1 Setup

Data We use the Automated Student Assessment Prize (ASAP)[2] dataset as evaluation data for our task, which contains 8 prompts of different genres as listed in Table 2. The essay scores are scaled into the range from 0 to 1. The settings of data preparation follow (Phandi et al., 2015). We use quadratic weighted kappa (QWK) as the metric. For domain-adaptation (cross-domain) experiments, we follow (Phandi et al., 2015), picking four pairs of essay prompts, namely, 1→2, 3→4, 5→6 and 7→8, where 1→2 denotes prompt 1 as source domain and prompt

[2]https://www.kaggle.com/c/asap-aes/data

Parameter	Parameter Name	Value
d_w	Word embedding dimension	100
h_{wrd}	Word context window size	5
h_{sent}	Sentence context window size	3
k_{wrd}	Word convolution units	50
k_{sent}	Sentence convolution units	50
p	Hidden size	50
drop_rate	Dropout rate	0.5
batch_size	Batch size	4
λ	Learning rate	0.01

Table 3: Neural Model Hyper-parameters

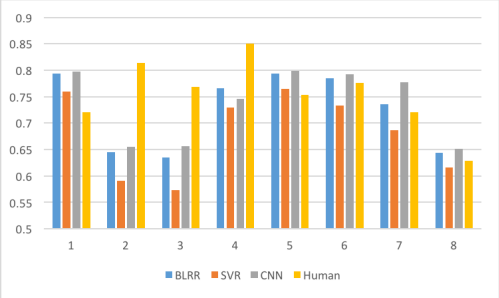

Figure 2: In-domain results

2 as target domain. All source domain essays are used as training data. Target domain data are randomly divided into 5 folds, where one fold is used as test data, and other 4 folds are collected together to sub-sample target domain train data. The sub-sampled sizes are 10, 25, 50, 100, with the larger sampled sets containing the smaller ones. And we repeated sub-sampling 5 times for each target training number to alleviate bias.

Hyper-parameters We use Adagrad for optimization. Word embeddings are randomly initialized and the hyper-parameter settings are listed in Table 3.

5.2 Results

In-domain The in-domain results are shown in Figure 2. The average values of all 8 prompt sets are listed in Table 4. For the in-domain task, CNN outperforms the baseline model SVR on all prompts of essay sets, and is competitive to BLRR. For the statistical significance, neural model is significantly better than baseline models with the p-value less than 10^{-5} at the confidence level of 95%. The average kappa value over 8 prompts is close to that of human raters.

Cross-domain The domain-adaptation results are shown in Table 5. It can be seen that our CNN

Model	BLRR	SVR	CNN	Human
Avg	0.725	0.682	0.734	0.754
Std dev	0.0025	0.0033	0.0029	—

Table 4: Indomain average kappa value and standard deviation over all 8 prompts.

Pairs	Method	$n_t = 10$	25	50	100
1→2	ML-ρ	0.365	0.437	0.521	**0.559**
	CNN	**0.546**	**0.569**	**0.563**	**0.559**
3→4	ML-ρ	0.435	0.540	0.590	0.619
	CNN	**0.628**	**0.656**	**0.659**	**0.662**
5→6	ML-ρ	0.415	0.600	0.678	0.718
	CNN	**0.647**	**0.700**	**0.714**	**0.750**
7→8	ML-ρ	0.328	0.438	0.496	0.551
	CNN	**0.570**	**0.590**	**0.568**	**0.587**

Table 5: Cross-domain results.

model outperforms ML-ρ on almost all pairs of adaptation experiments. ML-ρ domain-adaptation method's performance improves as the size of target domain training data increases. However, compared to ML-ρ, target training data size has less impact on our neural model. Even if the target training size is small, the neural model still gives strong performance. This results from the fact that neural model could learn more high-level and abstract features compared to traditional models with handcrafted discrete features. We plot the confusion matrix between truth and model prediction on test data in Figure 4, which shows that prediction scores of neural model tend to be closer to true values, which is very important in our task.

5.3 Feature Analysis

To visualize the features learned by our model, we use t-distributed stochastic neighbor embedding (t-SNE) (Van der Maaten and Hinton, 2008), projecting 50-dimensional features into 2-dimensional space. We take two domain pairs 3→4 and 5→6 as examples on the cross-domain task, extracting fully-connected hidden-layer features for target domain data using model trained on source domain data. The results are showed in Figure 3. The baseline discrete features are more concentrated, which shows that patterns on source prompt are weak in differentiating target prompt essays. By using ML-ρ and leveraging 100 target prompt training examples, the discrete features patterns are more scattered, increasing the differentiating power. In contrast, CNN

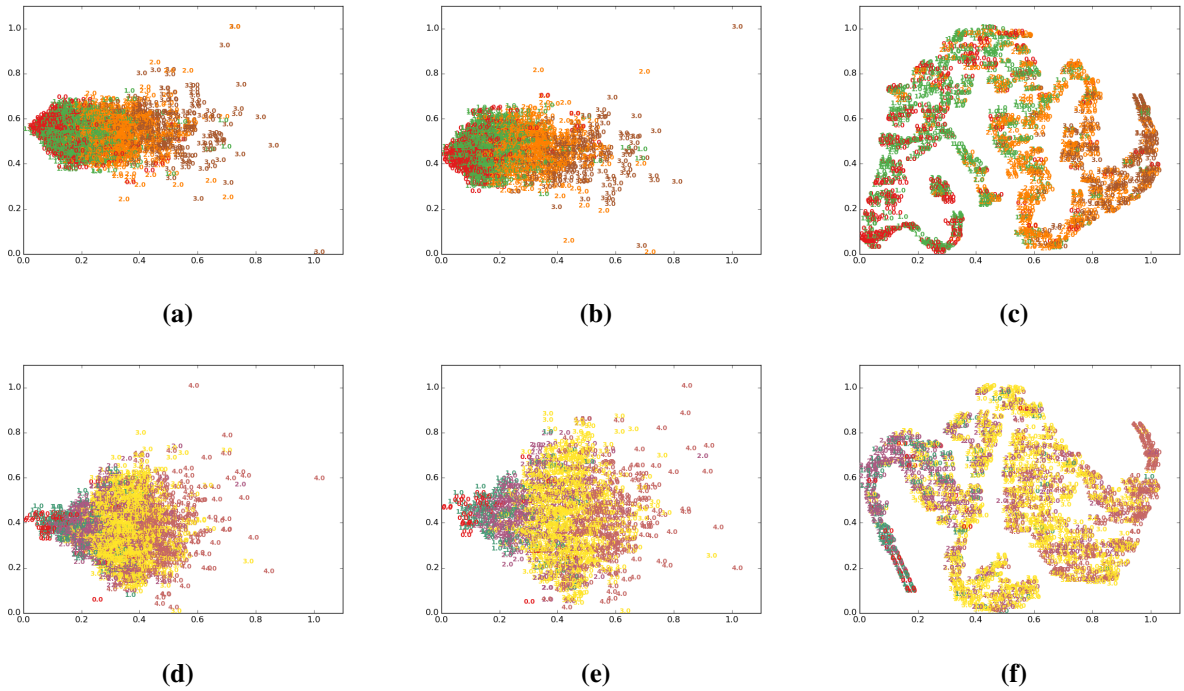

Figure 3: Visualization of discrete and neural features using t-SNE (each value represents an essay of the corresponding score). Top: Set 4 (3→4), Bottom: Set 6 (5→6). (a) discrete features; (b) ML-ρ features, $n_t = 100$; (c) neural features; (d) discrete features; (e) ML-ρ features, $n_t = 100$; (f) neural features.

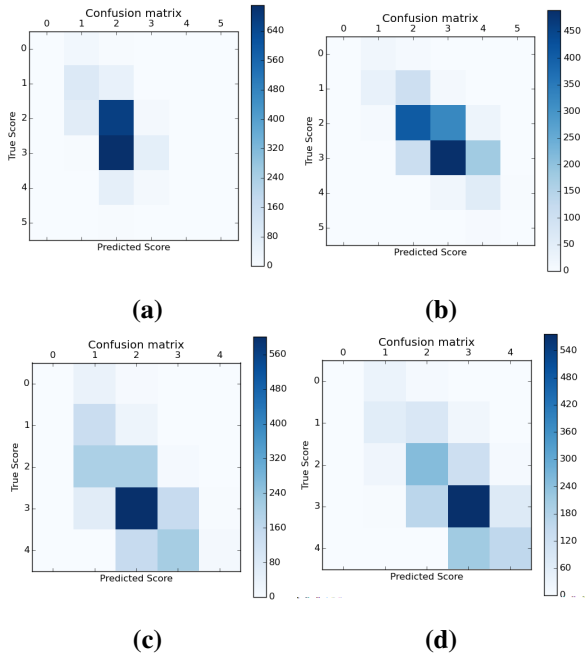

Figure 4: Confusion matrix of true and prediction scores by two different models on test data when target training size $n_t = 10$. (a) ML-ρ on 1→2; (b) CNN model on 1→2; (c) ML-ρ on 5→6; (d) CNN model on 5→6.

features trained on source prompt are sparse when used directly on the target prompt. This shows that neural features learned by the CNN model can better differentiate essays of different qualities. Without manual templates, such features automatically capture subtle and complex information that is relevant to the task.

6 Conclusion

We empirically investigated a hierarchical CNN model for automatic essay scoring, showing automatically learned features competitive to discrete handcrafted features for both in-domain and domain-adaptation tasks. The results demonstrate large potential for deep learning in AES.

Acknowledgments

We thank the anonymous reviewers for their constructive comments, which helped to improve the paper. This work is supported by NSFC61572245 and T2MOE201301 from Singapore Ministry of Education.

References

Dimitrios Alikaniotis, Helen Yannakoudakis, and Marek Rei. 2016. Automatic text scoring using neural networks. *arXiv preprint arXiv:1606.04289*.

Yigal Attali and Jill Burstein. 2004. Automated essay scoring with e-rater® v. 2.0. *ETS Research Report Series*, 2004(2):i–21.

Hongbo Chen and Ben He. 2013. Automated essay scoring by maximizing human-machine agreement. In *EMNLP*, pages 1741–1752.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.

Cícero Nogueira dos Santos and Maira Gatti. 2014. Deep convolutional neural networks for sentiment analysis of short texts. In *COLING*, pages 69–78.

Peter W Foltz, Darrell Laham, and Thomas K Landauer. 1999. Automated essay scoring: Applications to educational technology. In *proceedings of EdMedia*, volume 99, pages 40–64.

Rie Johnson and Tong Zhang. 2014. Effective use of word order for text categorization with convolutional neural networks. *arXiv preprint arXiv:1412.1058*.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Thomas K Landauer, Peter W Foltz, and Darrell Laham. 1998. An introduction to latent semantic analysis. *Discourse processes*, 25(2-3):259–284.

Leah S Larkey. 1998. Automatic essay grading using text categorization techniques. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 90–95. ACM.

Ellis Batten Page. 1994. Computer grading of student prose, using modern concepts and software. *The Journal of experimental education*, 62(2):127–142.

Peter Phandi, Kian Ming A Chai, and Hwee Tou Ng. 2015. Flexible domain adaptation for automated essay scoring using correlated linear regression.

Lawrence M Rudner and Tahung Liang. 2002. Automated essay scoring using bayes' theorem. *The Journal of Technology, Learning and Assessment*, 1(2).

Alex Smola and Vladimir Vapnik. 1997. Support vector regression machines. *Advances in neural information processing systems*, 9:155–161.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(2579-2605):85.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading esol texts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 180–189. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, pages 649–657.