Automatic Academic Paper Rating Based on Modularized Hierarchical Convolutional Neural Network

Pengcheng Yang[2], Xu Sun[1,2], Wei Li[1], Shuming Ma[1]
[1]MOE Key Lab of Computational Linguistics, School of EECS, Peking University
[2]Deep Learning Lab, Beijing Institute of Big Data Research, Peking University
{yang_pc, xusun, liweitj47, shumingma}@pku.edu.cn

Abstract

As more and more academic papers are being submitted to conferences and journals, evaluating all these papers by professionals is time-consuming and can cause inequality due to the personal factors of the reviewers. In this paper, in order to assist professionals in evaluating academic papers, we propose a novel task: automatic academic paper rating (AAPR), which automatically determine whether to accept academic papers. We build a new dataset for this task and propose a novel modularized hierarchical convolutional neural network to achieve automatic academic paper rating. Evaluation results show that the proposed model outperforms the baselines by a large margin. The dataset and code are available at https://github.com/lancopku/AAPR

1 Introduction

Every year there are thousands of academic papers submitted to conferences and journals. Rating all these papers can be exhausting, and sometimes rating scores can be affected by the personal factors of the reviewers, leading to inequality problem. Therefore, there is a great need for rating academic papers automatically. In this paper, we explore how to automatically rate the academic papers based on their LaTeX source file and meta information, which we call the task of automatic academic paper rating (AAPR).

A task that is similar to the AAPR is automatic essay scoring (AES). AES has been studied for a long time. Project Essay Grade (Page, 1967, 1968) is one of the earliest attempts to solve the AES task by predicting the score using linear regression over expert crafted textual features. Much of the following work applied similar methods by using various classifiers with more sophisticated features including grammar, vocabulary and style (Rudner and Liang, 2002; Attali and Burstein, 2004). These traditional methods can work almost as well as human raters. However, they all demand a large amount of feature engineering, which requires a lot of expertise.

Recent studies turn to use deep neural networks, claiming that deep learning models can relieve the system from heavy feature engineering. Alikaniotis et al. (2016) proposed to use long short term memory network (Hochreiter and Schmidhuber, 1997) with a linear regression output layer to predict the score. They added a score prediction loss to the original C&W embedding (Collobert and Weston, 2008; Collobert et al., 2011), so that the word embeddings are related to the quality of the essay. Taghipour and Ng (2016) also applied recurrent neural networks to process the essay, except that they put a convolutional layer ahead of the recurrent layer to extract local features. Dong and Zhang (2016) proposed to apply a two-layer convolutional neural network (CNN) to model the essay. The first layer is responsible for encoding the sentence and the second layer is to encode the whole essay. Dong et al. (2017) further proposed to add attention mechanism to the pooling layer to automatically decide which part is more important in determining the quality of the essay.

Although there has been a lot of work dealing with AES task, researchers have not attempted the AAPR task. Different from the essay in language capability tests, academic papers are much longer with much more information, and the overall quality is affected by a variety of factors besides the writing. Therefore, we propose a model that considers the overall information of one academic paper, including the title, authors, abstract and the main content of the LaTeX source file of the paper.

Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Short Papers), pages 496–502
Melbourne, Australia, July 15 - 20, 2018.

Our main contributions are listed as follows:

- We propose the task of automatically rating academic papers and build a new dataset for this task.
- We propose a modularized hierarchical convolutional neural network model that considers the overall information of the source paper. Experimental results show that the proposed method outperforms the baselines by a large margin.

2 Proposed Method

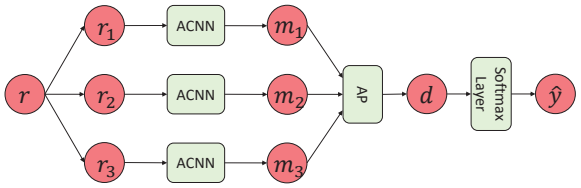
A source paper usually consists of several modules, such as *abstract*[1], *title* and so on. There is also a hierarchical structure from word-level to sentence-level in each module. The structure information is likely to be helpful to make more accurate predictions. Besides, the model can be improved by considering the difference in contributions of various parts of the source paper. Based on this observation, we propose a modularized hierarchical CNN. An overview of our model is shown in Figure 1. We assume that a source paper has l modules, with m words and the filter size is h (detailed explanations can be referred to Section 2.1 and Section 2.2). l, m and h are set to be 3, 3, 2, respectively in the Figure 1 for simplicity.

2.1 Modularized Hierarchical CNN

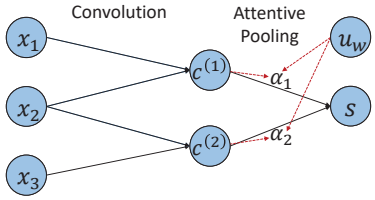
Given a complete source paper $\boldsymbol{r}$, represented by a sequence of tokens, we first divide it into several modules $(\boldsymbol{r_1}, \boldsymbol{r_2}, \cdots, \boldsymbol{r_l})$ based on the general structure of the source paper (*abstract, title, authors, introduction, related work, methods and conclusion*). For each module, the one-hot representation of the i-th word $\boldsymbol{w}_i$ is embedded to a dense vector $\boldsymbol{x}_i$ through an embedding matrix. For the following modules (*abstract, introduction, related work, methods, conclusion*), we use the attention-based CNN (illustrated in Section 2.2) in word-level to get the representation $\boldsymbol{s}_i$ of the i-th sentence. Another attention-based CNN layer is applied to encode the sentence-level representations into the representation $\boldsymbol{m}_i$ of the i-th module.

There is only one sentence in the title of the source paper, so it is reasonable to get the module-level representation of *title* only using attention-based CNN in word-level. Besides, the weighted average method is applied to obtain the module-level representation of *authors* by Equation (1) because the authors are independent of each other.

[1]Italicized words represent modules of the source paper.



(a) Modularized hierarchical convolutional neural network.



(b) Attention-based convolutional neural network.

Figure 1: The overview of our model. ACNN denotes attention-based CNN, whose basic structure is shown in (b). AP denotes attentive pooling.

$$\boldsymbol{m}_{authors} = \sum_{i=1}^{A} \gamma_i \boldsymbol{a}_i \tag{1}$$

where $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_A)^T$ is the weight parameter. $\boldsymbol{a}_i$ is the embedding vector of the i-th author in the source paper, which is randomly initialized and can be learned at the training stage. A is the maximum length of the author sequence.

Representations $\boldsymbol{m}_1, \boldsymbol{m}_2, \cdots, \boldsymbol{m}_l$ of all modules are aggregated to form the paper-level representation $\boldsymbol{d}$ of the source paper with an attentive pooling layer. A $softmax$ layer is used to take $\boldsymbol{d}$ as input and predict the probability of being accepted. At the training stage, the cross entropy loss function is optimized as objective function, which is widely used in various classification tasks.

$$\hat{\boldsymbol{y}} = softmax(\boldsymbol{W}_d \boldsymbol{d} + \boldsymbol{b}_d) \tag{2}$$

2.2 Details of Attention-Based CNN

Attention-based CNN consists of a convolution layer and an attentive pooling layer. The convolution layer is used to capture local features and attentive pooling layer can automatically decide the relative weights of words, sentences, and modules.

Convolution layer: A sequence of vectors of length m is represented as the row concatenation of m k-dimensional vectors: $\boldsymbol{X} = [\boldsymbol{x}_1; \boldsymbol{x}_2; \cdots; \boldsymbol{x}_m]$. A filter $\boldsymbol{W}_x \in \mathbb{R}^{h \times k}$ convolves

with the window vectors at each position to generate a feature map $\boldsymbol{c} \in \mathbb{R}^{m-h+1}$. Each element c_j of the feature map is calculated as follows:

$$c_j = f(\boldsymbol{W}_x \circ [\boldsymbol{x}_j : \boldsymbol{x}_{j+h-1}] + b_x) \quad (3)$$

where $\circ$ is element-wise multiplication, $b_x \in \mathbb{R}$ is a bias term, and f is a non-linear activation function. Here we choose f to be ReLU (Nair and Hinton, 2010). n different filters can be used to extract multiple feature maps $\boldsymbol{c}_1, \boldsymbol{c}_2, \cdots, \boldsymbol{c}_n$. We get new feature representations $\boldsymbol{C} \in \mathbb{R}^{(m-h+1)\times n}$ as the column concatenation of feature maps $\boldsymbol{C} = [\boldsymbol{c}_1, \boldsymbol{c}_2, \cdots, \boldsymbol{c}_n]$. The i-th row $\boldsymbol{c}^{(i)}$ of $\boldsymbol{C}$ is the new feature representation generated at position i.

Attentive pooling layer: Given a sequence $\boldsymbol{c}^{(1)}, \boldsymbol{c}^{(2)}, \cdots, \boldsymbol{c}^{(q)}$, which are q n-dimensional vectors, the attentive pooling is applied to aggregate the representations of the sequence by measuring the contribution of each vector to form the high-level representation $\boldsymbol{s}$ of the whole sequence. Formally, we have

$$\boldsymbol{z}_i = \tanh(\boldsymbol{W}_c \boldsymbol{c}^{(i)} + \boldsymbol{b}_c) \quad (4)$$

$$\alpha_i = \frac{\boldsymbol{z}_i^T \boldsymbol{u}_w}{\sum_k \exp(\boldsymbol{z}_k^T \boldsymbol{u}_w)} \quad (5)$$

$$\boldsymbol{s} = \sum_i \alpha_i \boldsymbol{z}_i \quad (6)$$

where $\boldsymbol{W}_c$ and $\boldsymbol{b}_c$ are weight matrix and bias vector, respectively. $\boldsymbol{u}_w$ is a randomly initialized vector, which can be learned at the training stage.

3 Experiments

In this section, we evaluate our model on the dataset we build for this task. We first introduce the dataset, evaluation metric, and experimental details. Then, we compare our model with baselines. Finally, we provide the analysis and the discussion of experimental results.

3.1 Dataset

Arxiv Academic Paper Dataset: As there is no existing dataset that can be used directly, we create a dataset by collecting data on academic papers in the field of artificial intelligence from the website[2]. The dataset consists of 19,218 academic papers. The information of each source paper consists of the venue which marks whether the paper is accepted, and the source LaTeX file. We divide the dataset into training, validation, and test parts. The details are shown in Table 1.

[2] https://arxiv.org/

Dataset	#Total	#Positive	#Negative
Training set	17,218	8,889	8,329
Validation set	1,000	507	493
Test set	1,000	504	496

Table 1: Statistical information of Arxiv academic paper dataset. **Positive** and **Negative** denote whether the source paper is accepted.

3.2 Experimental Details

We use accuracy as our evaluation metric instead of the F-score, precision, and recall because the positive and negative examples in our dataset are well balanced.

Since the author names are different from the common scientific words in the paper, we separately build up vocabulary for authors and text words of source papers with the size of 20,000 and 50,000, respectively.

We use the training strategies mentioned in Zhang and Wallace (2015) for CNN classifier to tune the hyper-parameters based on the accuracy on the validation set. The word or author embedding is randomly initialized and can be learned during training. The size of word embedding or author embedding is 128 and the batch size is 32. Adam optimizer (Kingma and Ba, 2014) is used to minimize cross entropy loss function. We apply dropout regularization (Srivastava et al., 2014) to avoid overfitting and clip the gradients (Pascanu et al., 2013) to the maximum norm of 5.0.

During training, we train the model for a fixed number of epochs and monitor its performance on the validation set after every 50 updates. Once training is finished, we select the model with the highest accuracy on the validation set as our final model and evaluate its performance on the testing set.

3.3 Baselines

We compare our model with the following baselines:

- **Randomly predict (RP):** We randomly decide whether the source paper can be accepted. In other words, the probability of acceptance of every source paper is always 0.5 using this strategy.

- **Traditional machine learning algorithms:** We use various machine learning classifiers

Models	Accuracy	Models	Accuracy
RP	50.0%	Logistic	60.0%
CART	58.6%	KNN	60.3%
MNB	58.3%	GNB	58.5%
SVM	61.6%	AdaBoost	58.9%
Bagging	59.4%	LSTM	60.5%
CNN	61.3%	C-LSTM	60.8%
MHCNN	**67.7%**		

Table 2: Comparison between our proposed model and the baselines on the test set. Our proposed model is denoted as **MHCNN**.

to predict the labels based on the tf-idf features of the text.

- **Neural networks models:** We apply three representative neural network models: CNN (Kim, 2014), LSTM (Hochreiter and Schmidhuber, 1997), and C-LSTM (Zhou et al., 2015). We concatenate all modules of the source paper into a long text sequence as the input to the neural network models.

3.4 Results

In this subsection, we present the results of evaluation by comparing our proposed method with the baselines. Table 2 reports experimental results of various models. As is shown in Table 2, the proposed MHCNN outperforms all the above mentioned baselines. The best baseline model SVM achieves the accuracy of 61.6%, while the proposed model achieves the accuracy of 67.7%. In addition, our MHCNN outperforms other representative deep-learning models by a large margin. For instance, the proposed MHCNN achieves an improvement of 6.4% accuracy over the traditional CNN. This shows that our MHCNN can learn better representation by considering modularized hierarchical structure in the source paper. Our proposed MHCNN aims to divide a long text into several modules and using attention mechanism to aggregate the representations of each module to form a final high-level representation of a complete source paper. By incorporating knowledge of the structure of the source paper and automatically selecting the most informative words, the model is capable of making more accurate predictions.

4 Analysis and Discussions

Here we perform further analysis on the model and experiment results.

Models	Accuracy	Decline
MHCNN	67.7%	−−
w/o Attention	66.8%*	↓0.9%
w/o Module	61.3%*	↓6.4%

Table 3: Ablation Study. The symbol * indicates that the difference compared to MHCNN is significant with $p \leq 0.05$ under t-test.

4.1 Exploration on Internal Structure of the Model

As is shown in Table 2, our MHCNN model outperforms all baselines by a large margin. Compared with the basic CNN model, the proposed model has a modularized hierarchical structure and uses multiple attention mechanisms. In order to explore the impact of internal structure of the model, we remove the modularized hierarchical structure and attention mechanisms in turn. The performance is shown in Table 3. "w/o Attention" means that we still use modularized hierarchical structure while do not use any attention mechanism. "w/o Module" means that we do not use both attention mechanism and modularized hierarchical structure, which is the same as the CNN model in the baselines.

As is shown in Table 3, the accuracy of the model drops by 0.9% when the attention mechanism is removed from the model. This shows that there are differences in the contribution of textual content. For instance, the *abstract* of a source paper is more important than its *title*. Attention mechanism can automatically decide the relative weights of modules, which makes model predictions more accurate. However, the accuracy of the model drops by 6.4% when we remove the modularized hierarchical structure, which is much larger than 0.9%. It shows that the modularized hierarchical structure of the model is of great help to obtain better representations by incorporating knowledge of the structure of the source paper.

4.2 The Impact of Modules of the Source Paper

One interesting issue is which part of the source paper best determines whether it can be accepted. To explore this issue, we subtract each module from complete source papers in turn and observe the change in the performance of the model. The experimental result is shown in Table 4.

As is shown in Table 4, the performance of the

Contexts	Accuracy	Decline
Full data	67.7%	−−
w/o *Title*	66.6%*	↓1.1%
w/o *Abstract*	65.5%*	↓2.2%
w/o *Authors*	64.6%*	↓3.1%
w/o *Introduction*	65.7%*	↓2.0%
w/o *Related work*	66.0%*	↓1.7%
w/o *Methods*	66.2%*	↓1.5%
w/o *Conclusion*	65.0%*	↓2.7%

Table 4: Ablation Study. The symbol * indicates that the difference compared to full data is significant with $p \leq 0.05$ under t-test.

model shows different degrees of decline when we remove different modules of the source paper. This shows that there are differences in the contribution of different modules of the source paper to its acceptance, which further illustrates the reasonableness of our use of modularized hierarchical structure and attention mechanism. All the declines are significant with $p \leq 0.05$ under the t-test.

When we remove *authors* module, the accuracy drops by 3.1%, which is the largest decline. This shows that the authors of the source paper largely determines whether it can be accepted. Obviously, a source paper written by a proficient scholar tends to be good work, which has a higher probability of being accepted. Except for *authors*, the two most significant modules affecting the probability of being accepted are *conclusions* and *abstract*. Because they are the essence of the entire source paper, which can directly reflect the quality of the source paper. However, the *methods* module of the source paper has little effect on the probability of being accepted according to Table 4. The reason may be that the *methods* of different source papers vary widely, which means that there exists high variance in this module. Therefore, our model may not do well in capturing a unified internal pattern to make prediction. The impact of the *title* is the smallest and the accuracy of the model drops by only 1.1% when *title* is removed from the source paper.

5 Related Work

The most relevant task for our work is automatic essay scoring (AES). There are two main types of methods for the AES task: traditional machine learning algorithms and neural network models.

Most traditional methods for the AES task use supervised learning algorithms, including classification (Larkey, 1998; Rudner and Liang, 2002; Yannakoudakis et al., 2011; Chen and He, 2013), regression (Attali and Burstein, 2004; Phandi et al., 2015; Zesch et al., 2015) and so on. However, they all require lots of manual features, for instance, bag of words, spelling errors, or lengths, which can be time-consuming and requires a large amount of expertise.

In recent years, some neural network models have also been used for the AES task, which have achieved great success. Alikaniotis et al. (2016) proposed to use the LSTM model with a linear regression output layer to predict the score. Taghipour and Ng (2016) applied the CNN model followed by a recurrent layer to extract local features and model sequence dependencies. A two-layer CNN model was proposed by Dong and Zhang (2016) to cover more high-level and abstract information. Dong et al. (2017) further proposed to add attention mechanism to the pooling layer to automatically decide which part is more important in determining the quality of the essay. Song et al. (2017) proposed a multi-label neural sequence labeling approach for discourse mode identification and showed that features extracted by this method can further improve the AES task.

6 Conclusions

In this paper, we propose the task of automatic academic paper rating (AAPR), which aims to automatically determine whether to accept academic papers. We propose a novel modularized hierarchical CNN for this task to make use of the structure of a source paper. Experimental results show that the proposed model outperforms various baselines by a large margin. In addition, we find that the conclusion and abstract parts have the most influence on whether the source paper can be accepted when setting aside the factor of authors.

7 Acknowledgements

This work is supported in part by National Natural Science Foundation of China (No. 61673028), National High Technology Research and Development Program of China (863 Program, No. 2015AA015404), and the National Thousand Young Talents Program. Xu Sun is the corresponding author of this paper.

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).

Chengyao Chen, Zhitao Wang, Wenjie Li, and Xu Sun. 2018. Modeling scientific influence for research trending topic prediction. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, February 2-7, 2018*.

Hongbo Chen and Ben He. 2013. Automated essay scoring by maximizing human-machine agreement. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1741–1752.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.

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

Fei Dong and Yue Zhang. 2016. Automatic features for essay scoring – an empirical study. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1072–1077.

Fei Dong, Yue Zhang, and Jie Yang. 2017. Attention-based recurrent convolutional neural network for automatic essay scoring. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 153–162.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

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.

Dehong Ma, Sujian Li, Xiaodong Zhang, Houfeng Wang, and Xu Sun. 2017. Cascading multiway attentions for document-level sentiment classification. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 634–643.

Fanqi Meng, Dehong Gao, Wenjie Li, Xu Sun, and Yuexian Hou. 2013. A unified graph model for personalized query-oriented reference paper recommendation. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 1509–1512. ACM.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814.

Ellis B Page. 1967. Grading essays by computer: Progress report. In *Proceedings of the invitational Conference on Testing Problems*.

Ellis B Page. 1968. The use of the computer in analyzing student essays. *International review of education*, 14(2):210–225.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318.

Peter Phandi, Kian Ming A Chai, and Hwee Tou Ng. 2015. Flexible domain adaptation for automated essay scoring using correlated linear regression. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 431–439.

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

Wei Song, Dong Wang, Ruiji Fu, Lizhen Liu, Ting Liu, and Guoping Hu. 2017. Discourse mode identification in essays. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 112–122.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

Kaveh Taghipour and Hwee Tou Ng. 2016. A neural approach to automated essay scoring. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1882–1891.

Bingzhen Wei, Xu Sun, Xuancheng Ren, and Jingjing Xu. 2017. Minimal effort back propagation for convolutional neural networks. *arXiv preprint arXiv:1709.05804*.

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.

Torsten Zesch, Michael Wojatzki, and Dirk Scholten-Akoun. 2015. Task-independent features for automated essay grading. In *Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 224–232.

Ye Zhang and Byron C. Wallace. 2015. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *CoRR*, abs/1510.03820.

Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis C. M. Lau. 2015. A C-LSTM neural network for text classification. *CoRR*, abs/1511.08630.