An automatic grading system for electronic medical records with neural network

Yi Liu
Department of Computer Science and Engineering, Shanghai Jiao Tong University, 800 Dongchuan Road, Shanghai, China

*E-mail: liuyi61ly@hotmail.com

Abstract. Automatically grading electronic medical records from clinical researchers is an important task on healthcare research. In this paper, we present a convolutional neural network framework to grade medical records by grading note. We regard the scoring process as a text pattern classification task with mapped sentences from grading concept rule to medical records. This framework involves two stages. The first stage is key medical concept matching between grading note and medical records. The second stage is text pattern classification which can predict whether the key concept in grading note is correct, missing or incorrect. The result shows that our neural network model performs better than other traditional machine learning grading methods. Our system makes a great progress for text pattern classification accuracy and performs much better baselines on grading process.

1. Introduction
Automatically grading electronic medical records (EMR) for clinical researchers has attracted much research attention in healthcare community. EMRs help doctors further understandings of patients’ health conditions, identifying their best-matching patient category and determining patients’ customized treatment plan [1-2]. Unfortunately, before writing an excellent EMR, it takes a lot of time to train a recorder. The most important part in this progress is grading EMRs written by recorders.

Each EMR is collected by the conversion between doctor and patient. It usually contains various types of information, including patient personal information, past medical history, present illness and symptom etc. A good EMR should include all the important information known as key medical concepts. The grading task should judge if a recorder captured the key medical concepts. As shown in figure 1, medical concept is called grading rubric. If a grading rubric can be found correctly in the corresponding EMR, it will be labelled as C. Similarly, I will represent incorrect and M will represent missing.

A quick approach called word-count based model first scans all the words in grading rubric and EMR and scoring according to the proportion of words common to both texts. However, without considering semantic information will reduce the accuracy of scoring. Thus, the semantic distance representation between two texts has been developed. The two most common ways for representing texts is via a bag of words (BOW) [3] or by their term frequency inverse document frequency (TF-IDF) [4]. Furthermore, the Word Mover’s Distance (WMD) is developed to improve the above approaches. These methods can reflect the EMRs quality in a certain degree but is interfered with a large amount of irrelevant information in EMRs. Also, they cannot map the grading rubric exactly to...
the corresponding sentence in EMRs. In this paper, we build an automatically grading system for EMR. It first matches grading rubric to sentence in EMR with closest semantic relatedness. Then, a convolutional neural network (CNN) is established for classifying new concatenated text pattern. We score the EMR by evaluating number of C labels as a percentage of the total number of labels in grading note.

![Figure 1. Grading rubric and medical record.](image1)

**Figure 1.** Grading rubric and medical record.

### 2. The proposed methodology of grading system

#### 2.1. General framework of grading system

As shown in figure 2, automatically grading system (auto-grader) considers the grading process as a two-stage pipeline framework. The first stage is called grading rubric matching. We split each grading note into a list of grading rubrics and sentence level vectors. Meanwhile, we get sentence vectors for EMR by same method. Then, we map the sentence embedding in grading rubric to EMR by calculating relatedness score of two pairs of sentences. The second stage is regarded as text pattern classification. We present a slight variant of the CNN architecture to predict whether the key concept is correct, missing or incorrect. The score is calculated based on the prediction results.

![Figure 2. Workflow of auto-grading system with CNN model.](image2)

**Figure 2.** Workflow of auto-grading system with CNN model.

#### 2.2. Grading rubric matching

##### 2.2.1. Sentence embedding

The most important part of grading rubric matching is calculating sentence embedding through word vector. The supervised learning methods usually train a neural work and use the last hidden layer as sentence embedding [6]. However, these methods require additional annotated data. Here, we get sentence embedding $v_s$ by weighted averaging word embedding $v_w$ which is similar to Sanjeev’s work [7].

##### 2.2.2. Semantic relatedness sentences extraction and mapping

In order to judge whether grading rubric is correct, incorrect or missing, we need to find out if there’s a sentence in EMR which contains exact same meaning of grading rubric. Sentence similarity is a
basic way to compare the semantic relatedness of a pair of sentences. Here, we use the Cosine Similarity to measure the relatedness score of the grading rubric in grading rubric to corresponding sentence in EMR. Here \( v_g = [x_1,x_2,x_3,\cdots,x_n] \) and \( v_r = [y_1,y_2,y_3,\cdots,y_n] \). The cosine similarity is
\[
\text{CosSim}(v_g,v_r) = \frac{x_1y_1+x_2y_2+\cdots+x_ny_n}{\sqrt{x_1^2+x_2^2+\cdots+x_n^2}\sqrt{y_1^2+y_2^2+\cdots+y_n^2}}
\]
where \( v_g \) stands for the embedding of grading rubric and \( v_r \) stands for sentence embeddings in EMR. For each \( v_g \), we select sentence embedding \( v_r \) in EMR with the highest score and map them together.

2.3. Grading rubric labelling and scoring

2.3.1. Convolutional neural network for label prediction

The model architecture is similar to the CNN architecture of Yoon Kim’s work \[8\]. Let \( x_i \in R_k \) be the k-dimensional word vector corresponding to the \( i \)th word in grading rubric and \( y_i \in R_k \) be the k-dimensional word vector corresponding to the \( i \)th word in the most related sentence in EMR. Let the max of sentence length be \( n \) (padded when sentence length is less than \( n \)) and the concatenated text pattern can be represented as equation (2),
\[
T_{1:2n} = x_1 \odot x_2 \cdots \odot x_n \odot y_1 \odot y_2 \odot \cdots \odot y_n = t_1 \odot t_2 \odot \cdots \odot t_{2n}
\]
where \( \odot \) is the concatenation operator. In this paper, let \( T_{ij} \) refer to the concatenation of words \( t_i,t_{i+1},\cdots,t_{i+j} \). The input layer becomes a matrix of word vectors corresponding to the words in the text pattern. There are \( 2n \) words in the text pattern, so the matrix is \( 2n \times k \).

This type of matrix can be static or non-static according to \[9-10\]. We use pretrained embedding to make the matrix static. For the convolutional layer, a filter \( w \in R^{hk} \) is applied to a window of \( h \) to produce a new feature during convolution operation. For example, a feature \( \alpha_i \) is generated from a window of words \( t_{i:h-1} \) by
\[
\alpha_i = f(w \cdot x_{i:i+h-1} + b)
\]
where \( b \in R \) is a bias variable and \( f(\cdot) \) is a rectified linear unit (ReLU). This filter is applied to each possible window of words in the text pattern \( \{t_{i:h},t_{i+1},\cdots,t_{i+h-1} \} \) to produce a feature map \( \alpha = [\alpha_1,\alpha_2,\cdots,\alpha_{2n-h+1}] \) and \( \alpha \in R^{2n-h+1} \).

We then apply a pooling layer. A max-over-time pooling method \[8\] is used to extract the maximum value \( \bar{\alpha} = \max \{\alpha\} \) from the previous one-dimensional feature map \( \alpha \) as the feature. Just like extracting one feature from one filter, this model can also use multiple filters with different window sizes to get multiple features. The pooling layer outputs the one-dimensional features and fully connects softmax layer. Similar to traditional neural networks, the output layer is defined as
\[
y = w \cdot (z \ast r) + b
\]
Here \( z = [\tilde{\alpha}_1,\tilde{\alpha}_2,\cdots,\tilde{\alpha}_m] \) and \( m \) represent that we have \( m \) filters. The softmax layer reflects the probability distribution on the final classification with \( n \) labels and can be set with the need of the task. We employ dropout to prevent neural networks from overfitting and \( * \) is the element-wise multiplication operator and \( r \) is a vector of \( m \) Bernoulli random variables which have certain probability to be 0 or 1.

2.3.2. Grading electronic medical records

The percentage of concepts need to be captured by doctor and patient determines the score of each EMR. For each generated text pattern, we calculate the number of each label and get the percentage of label C as the final score of EMRs. As equation (8) shows,
\[
\text{Score} = \frac{\text{num}_C}{\text{num}_C + \text{num}_I + \text{num}_M} \cdot 100
\]
where Score is the final score of EMRs and \( \text{num}_i \) (\( i = C,I,M \)) represent the number of each label of grading rubric.
3. Experiments and results

3.1. Dataset details
The EMRs are collected from our partner hospital and the data only permits noncommercial or academical use. The grading rubric and its label was done by several doctors. The data contain two parts. The first is EMRs which was written by recorders. These EMRs are based on conversation between clinicians and patients. They contain many abbreviated messages and incomplete expressions from oral information. The second part is grading notes which are written by doctor. A grading note contains grading rubrics which a recorder must record when he is listening conversation between clinicians and patients. We manually go through every grading rubric and label them by \{ C, I, M \} for correct, incorrect or missing.

There are 1429 grading note and EMR pairs. Each grading note contains dozens to hundreds of rubrics. Firstly, we match every rubric with their most related sentence in EMRs. This process generates concatenated text pattern of two sentences with labels. We then use machine learning approaches to predict the labels of concatenated text pattern. Here, about 80000 concatenated text patterns will be the train data in the next stage.

3.2. Baseline model
To establish a baseline performance, we use weighted averaging the word vectors to get sentence embedding. We then concatenate both the sentence embedding pairs in grading rubric and EMR by calculating their cosine similarity. The label prediction problem can be regard as a vector classification problem. We train a nearest neighbor classifier (KNN), a logistic regression classifier (LR), a decision tree (DT) and a multi-layer neural network.

3.3. Hyperparameters and training
We initialize word embedding by using publicly available data which were trained on 100 billion words from Google News. The word embeddings are 300-dimensional word vectors trained by continuous bag-of-words architecture [11]. Words which are not present in the set of pre-trained embeddings will be initialized randomly. The class labels can be divided into two category that is 0 for C and 1 for M and I. For parameter setting, we refer to Yoon Kim’s work [10] which use filter windows (h) of 3, 4, 5 with 128 feature maps each. We set the dropout rate of 0.5 and default batch size of 64. We select 80% of the text pattern as the train set and select test data set from the rest of 20%. Training is done through Adam. Cross entropy loss measures the performance of our CNN model. We calculate a separate loss for each class label per text pattern and sum the result.

3.4. Result and discussion
We compare our method with widely used text pattern classification methods. The experimental results in table 1. show that our auto-grader with CNN outperform the traditional methods on predicting three labels in the EMRs grading task. It proves that CNN-based approach can effective compose the semantic representation of texts.

| Model                     | Accuracy |
|---------------------------|----------|
| Sentence Embedding + KNN  | 64.27%   |
| Sentence Embedding + LR   | 59.62%   |
| Sentence Embedding + DT   | 69.57%   |
| Sentence Embedding + Multi-Layer Neural Network | 67.46% |
| Auto-grader with CNN      | 70.46%   |

For grading result comparison, we introduce mean absolute error (MAE) rate and root-mean-square error (RMSE). Mean absolute error is used to measure the difference between the scores we
grade and the real scores. The mean absolute error rate MAE is defined as

$$MAE = \frac{1}{N} \sum_{k=1}^{N} |\text{real} - \text{pre}|$$

(6)

where \text{real} represent real score and \text{pre} represent predicted score by our auto-grader. The root-mean-square error is used to measure the bias of grading result.

$$RMSE = \sqrt{\frac{1}{N} \sum_{k=1}^{N} (\text{real} - \text{pre})^2}$$

(7)

Here \(N\) is the number of EMRs need to be graded.

| Model                                | MAE  | RMSE |
|--------------------------------------|------|------|
| Word-count Based Model               | 13.92| 16.89|
| Sentence Embedding +KNN              | 11.34| 13.88|
| Sentence Embedding +LR               | 13.47| 16.63|
| Sentence Embedding +DT               | 10.61| 13.38|
| Sentence Embedding +Multi-Layer Neural Network | 11.23| 14.55|
| Auto-grader with CNN                 | **10.09** | **12.96** |

From table 2, we can see auto-grader achieves better performance both in MAE and RMSE measurements.

4. Conclusion
In the present work we have described an automatic grading system for EMRs with convolutional neural networks. Our method can not only grade the EMRs through key medical concept but can also match the key concept with the corresponding sentence if it exists in EMR. We evaluated our work with other models such as word-count based model, linear regression, k-neighbors classifier and so on. The result shows that our automatically grading system performs better than machine learning baselines. We believe our CNN-based auto-grader will bring more inspiration to real word machine learning applications.

Acknowledgments
The author would like to thank Sabbir, Nadim, Kevin and Sarah’s suggestions. Augmedix Inc. provides the environment and conditions for this study.

References
[1] Z. Huang, M.-L. Shyu, J. M. Tien, M. Vigoda, D. J. Birnbach. Knowledge-assisted sequential pattern analysis with heuristic parameter tuning for labor contraction prediction. IEEE J. Biomed. Health Informat., 18, 492 (2014)
[2] F. Liu, L. Zhou, C. Shen, J. Yin. Multiple kernel learning in the primal for multimodal Alzheimers disease classification. IEEE J. Biomed. Health Informat., 18, 984 (2014)
[3] Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781. (2013a)
[4] S. Robertson. Understanding inverse document frequency: on theoretical arguments for IDF. Journal of documentation, 60(5), 503 (2004)
[5] M Kusner, et al. From word embeddings to document distances. International Conference on Machine Learning. (2015)
[6] J. mMueller, and Aditya Thyagarajan. Siamese Recurrent Architectures for Learning Sentence Similarity. In AAAI, 16 (2016)
[7] A. Sanjeev, Y. Liang, T. Ma. A simple but tough-to-beat baseline for sentence embeddings. (2016)
[8] Kim, Yoon. Convolutional neural networks for sentence classification. arXiv preprint
[9] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, P. Kuksa. Natural Language Processing (Almost) from Scratch. Journal of Machine Learning Research 12 (2011)

[10] R. Socher, J. Pennington, E. Huang, A. Ng, C. Manning. Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions. In Proceedings of EMNLP (2011)

[11] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, J. Dean. Distributed Representations of Words and Phrases and their Compositionality. In Proceedings of NIPS (2013)