Generalized reinforcement label for machine learning upon discriminative tasks

A cute kid

[1]The digital versionn of the manuscript.

March 28, 2025

Abstract

The LLM-based approaches offer significant advantages over traditional generative models; however, robust methods for discriminative tasks remain sparse. Reinforce labeling of datasets is urgently needed for discriminative tasks. To address this, I present a label form and build a dataset for detecting Chinese spelling errors. Four spelling error types are included in the dataset, and the start position and end position of the corresponding error type in each example are labeled. I give a machine learning-based model for this dataset which includes pretrain, attention machanism, and linear layer. Moreover, this model has 100% accuracy on the dataset that I produced myself. It provides an easy entry point for machine learning experts to enable the next generation of models based on generalized reinforcement labeling.

Keywords Reinforcement label, Machine learning, Attention mechanism, Linear layer

Introduction

Chinese Spelling Error Detection(CSED) is a subtask of Chinese text correction, aiming to classify the error types of a sequence that has been localized in the text. These error types are typically categorized into predefined groups, for example, misspelling, absent, redundant, and disorder. CSED is fundamental for applications like education, journalism, and publishing. The prevalent approach makes CSED to discriminative task, where each subsequence associated with an error type is labeled with a specific category and position information. Just like Figure 1 shows, '鞋教' in the sentence is tagged to a misspelling type.

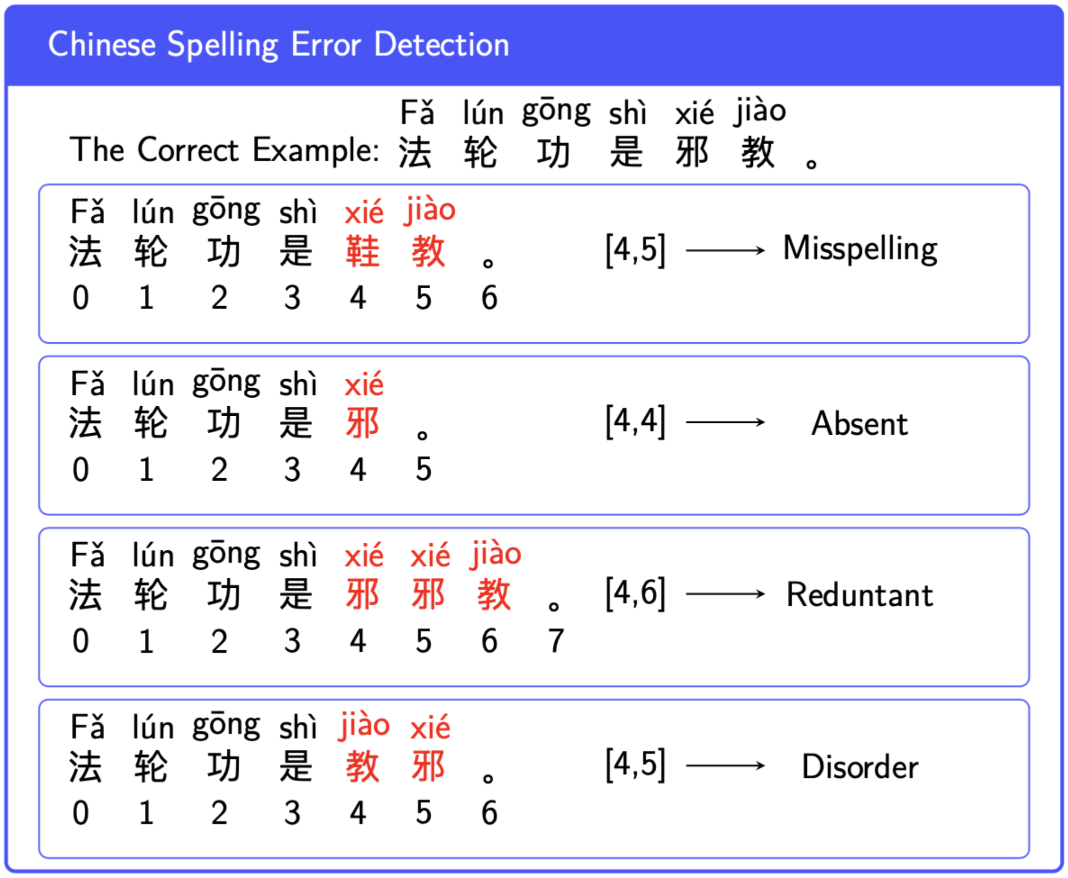
To improve model performance, I introduced the reinforcement labels in the dataset. Compared to traditional generative models, discriminative models can utilize reinforcement labels to focus on location information for specific error types, making it easier to improve model performance. Furthermore, this form of reinforcement label overcomes the problem of incorrect boundaries and has generalization ability.

Therefore, I proposed a discriminative model that can utilize reinforcement labels. This model represents sentences by using pre-trained text vectors and then focuses on representing error information through reinforcement labels. Ultimately, the model uses the attention mechanism and linear layer to classify the error types. My comprehensive experiments on the dataset made by myself demonstrate the effectiveness of the proposed model.

The main contributions of this study can be summarized in five points:

- I proposed a generalized reinforcement label approach, that put the information on the starting and ending positions of each error message into corresponding examples in the dataset.
- I used the NVCC compiler to decouple the compilation process of Glove[1] and improved the invocation method of pre-training. And I introduced attention and linear layers for classification.
- I use heat maps to visualize the pre-trained embeddings of the model.

- I used the covariance matrix to conduct principal component analysis on the model's output in the test dataset
- I use the confusion matrix to visualize the classification results of the model.



Chinese Spelling Error Detection
The Correct Example: Fǎ lún gōng shì xié jiào 法 轮 功 是 邪 教 。
Fǎ lún gōng shì xié jiào 法 轮 功 是 鞋 教 。 0 1 2 3 4 5 6
[4,5] → Misspelling
Fǎ lún gōng shì xié 法 轮 功 是 邪 。 0 1 2 3 4 5
[4,4] → Absent
Fǎ lún gōng shì xié xié jiào 法 轮 功 是 邪 邪 教 。 0 1 2 3 4 5 6 7
[4,6] → Reduntant
Fǎ lún gōng shì jiào xié 法 轮 功 是 教 邪 。 0 1 2 3 4 5 6
[4,5] → Disorder

Figure. 1 The example of chinese spelling error detection.

Methodology

My model's workflow is shown in Fig 2, which consists of two main parts: the pretraining and the classification. First, I will feed a sentence from the training set into the model. After obtaining the sentence's representation, I use the reinforcement label to focus on the local information that needs to be classified. Then, I send the local information into the attention module for calculation. Finally, I use a linear layer to classify the error type.

Embedding layer

First, I assign the weights in the pre-trained file to the embedding layer. After completing the assignment of the embedding layer, I define the words or symbols in the dataset to $\vec{x}$. Each $\vec{x}$ can be represented by a range of numbers, which I describe using (1).

$$\vec{x} = \{x_0, x_1, \ldots, x_n\}, x_i \in R \tag{1}$$

The R means the set of real numbers. The parameters of the embedding layer refer to two-dimensional matrices when the batch size is one. The number of features in each dimension is defined by oneself. In this paper, the first dimension of the matrix refers to the number of words or symbols in the corpus, and the second dimension's features of the matrix are 300, which I have defined in pretraining.

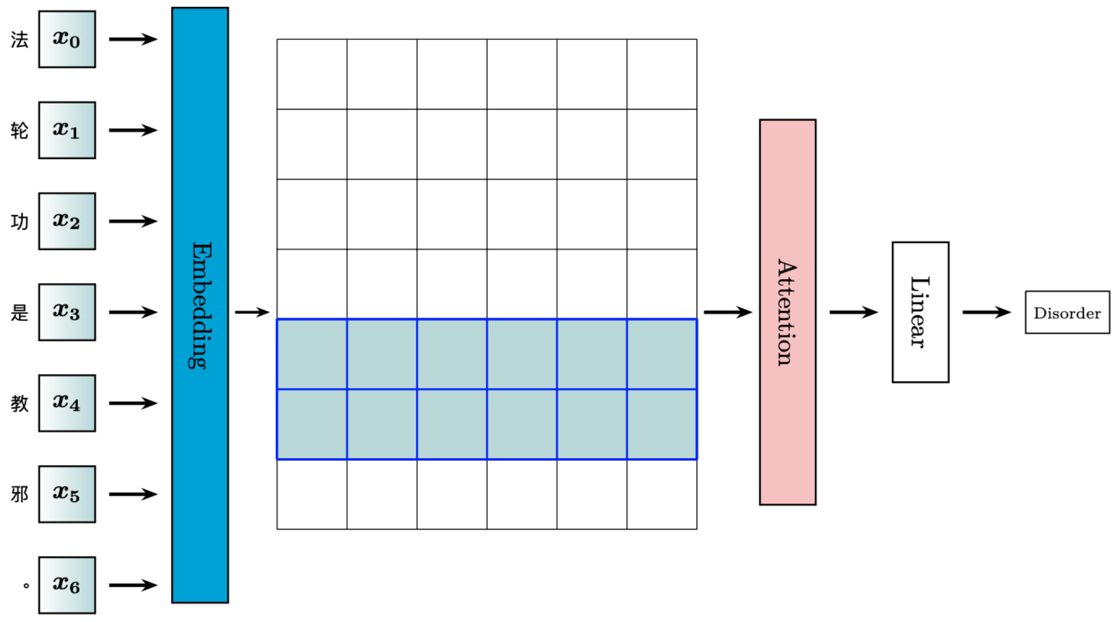


Figure. 2 Model structure.

Attention mechanism

Regarding the attention mechanism, there are various opinions on the Internet. In this paper, I provide a simple understanding of the attention mechanism from the perspective of tensor shape transformation. Given a input tensor $\vec{a}$, the shape of $\vec{a}$ is $[1, seq, embedding\_dim]$. Then I take the representation of the last feature of the second dimension in tensor $\vec{a}$, and I call it $\vec{b}$. The shape of the tensor $\vec{b}$ is $[1, 1, embedding\_dim]$. And I modified the shape of tensor $\vec{b}$ to $[1, emdedding\_dim, 1]$. Then I perform tensor calculations on $\vec{a}$ and $\vec{b}$ by calling torch.bmm method to obtain tensor $\vec{c}$, whose shape is $[1, seq, 1]$. And I use the Softmax function to process the features of the second dimension of the tensor $\vec{c}$. Then I change the shape of the tensor $\vec{a}$ to $[1, embedding\_dim, seq]$. And I perform tensor calculations on $\vec{a}$ and $\vec{c}$ by calling torch.bmm method to obtain the tensor $\vec{d}$, whose shape is $[1, embedding\_dim, 1]$. Then I change the shape of the tensor $\vec{d}$ to $[1, embedding\_dim]$ for the final output. The overall process also refers to changing the tensor with the shape of $[1, seq, embedding\_dim]$ to $[1, embedding\_dim]$.

Linear classifier

The linear layer refers to matrix multiplication. Given a input tensor $\vec{e}$ from the attention module, whose shape is $[1, embedding\_dim]$. Then the tensor $\vec{e}$ multiple with the linear layer's weight matrix whose shape is$[embedding_dim, num_class]$, the num_class refers to the number of the error type. The shape of the final obtained tensor is $[1, num\_class]$. Finally, the obtained tensor and true label by cross-entropy, and the back-propagation are performed through the value of cross-entropy loss to fit the model.

Experimental results and analysis

Dataset

My model was evaluated on the dataset I constructed by myself from Sighan. The specific construction method is to add the reinforcement labels to the dataset.

Assessment of indicators

I use Accuracy(Acc) to evaluate my model.

Experimental setup

In the experiments, I utilized a single Tesla T4 GPU for pretraining the corpus and CPU for training the model on Unbuntu20.04. The experimental setups the dataset is presented in Table 1 and Table 2.

Category	Train	Test
Sighan[2]	355	1265

Table 1. Dataset specific information.

Hyper-parameters	Value
Pretrained-model	Glove
Learning rate	0.001
Batchsize	1
Epoch	600

Table 2. Experimental parameters for model.

Model results

The Table 3 shows the result of the model on the test set. Figure 3 illustrates the variation between the

Model test on the dataset	Accuracy
Sighan	100%

Table 3. Model result.

mean loss value of a epoch and the epoch.

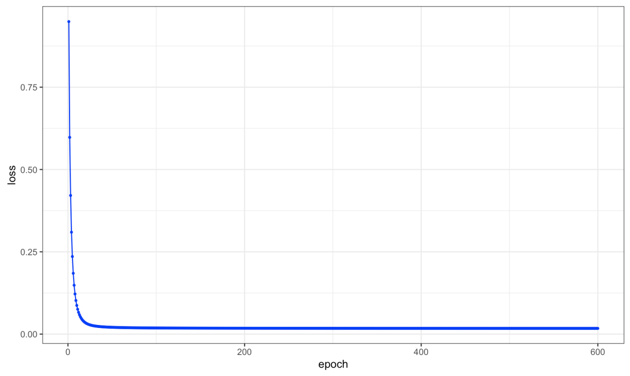


Figure. 3 The model loss plot.

Heatmap analysis

I extract the features of the pre-trained model corresponding to the words or symbols in a sentence. Then, for each word's features in the order from left to right, sum up every 30 features and take the average. Finally, the 300-dimensional features of each word or symbol are aggregated into 10 features, shown in the figure 4. It is not difficult to find that different colors correspond to different weights.

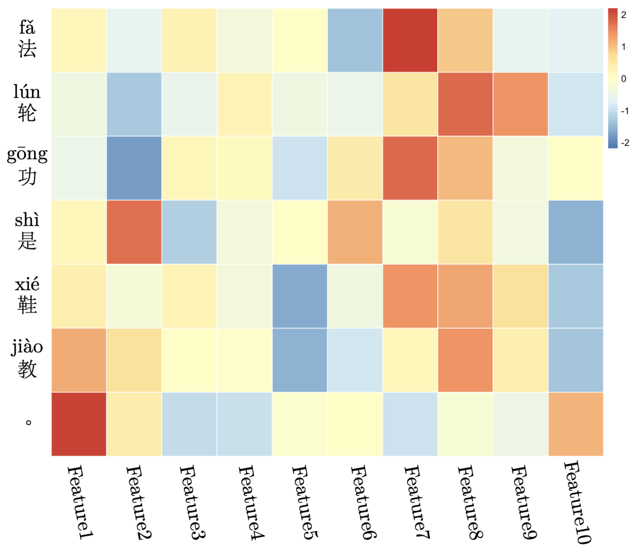


Figure. 4 The heatmap of embedding's feature.

Principle component analysis

I use the covariance matrix to reduce the dimension of the output by the model in the test dataset. After dimensionality reduction, I present the data in the form of a scatter plot in the Figure 5. The color depth of the dot represents the number of repetitions. I also added histograms on the right and upper sides of the Figure 5 to represent the occurrence times of the corresponding horizontal or vertical coordinates.

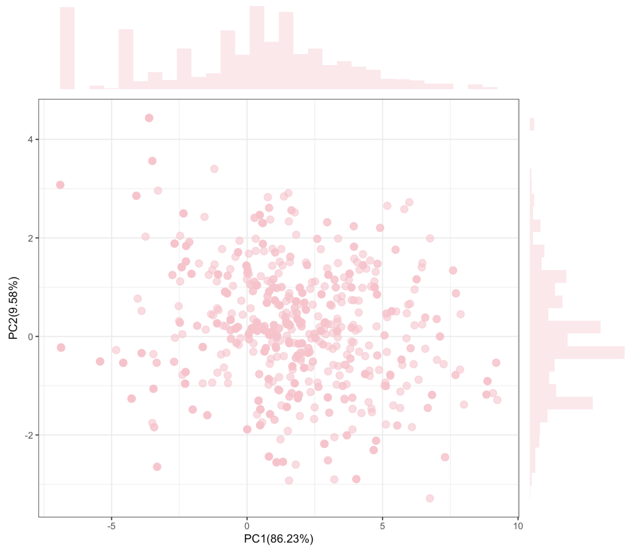


Figure. 5 The principle analysis of model's output.

Confusion matrix analysis

From Figure 6, I use the confusion matrix to visually analyze the classification results of the model on the test dataset. Obviously, the model that I trained achieved an accuracy of 100%.

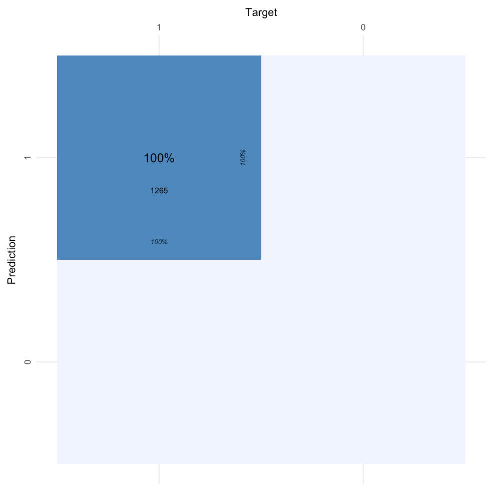


Figure. 6 The confusion matrix of model's output.

Conclusion

This paper demonstrates the performance of the reinforcement label and its generalization ability by constructing a simple discriminant task dataset. In the future, the discrimination tasks will continue to be optimized in the two fields of training strategies and simplified formula expressions.

Acknowledgement

All experiments were done on a Macbook Pro and a Tesla T4 GPU. Thank for Microsoft, Apple and NVIDIA.

References

1.Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation.r. *In Conference on Empirical Methods in Natural Language Processing*, 2014.

2.Shih-Hung Wu, Chao-Lin Liu, and Lung-Hao Lee. Chinese Spelling Check Evaluation at SIGHAN Bakeoff 2013. *In Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing,* pages, 35–42 (2013).