Correcting Chinese Spelling Errors with Phonetic Pre-training

Ruiqing Zhang, Chao Pang, Chuanqiang Zhang, Shuohuan Wang,
Zhongjun He,[*] Yu Sun, Hua Wu and Haifeng Wang
Baidu Inc. No. 10, Shangdi 10th Street, Beijing, 100085, China
{zhangruiqing01, pangchao04, zhangchuanqiang, wangshuohuan}@baidu.com
{hezhongjun, sunyu02, wu_hua, wanghaifeng}@baidu.com

Abstract

Chinese spelling correction (CSC) is an important yet challenging task. Existing state-of-the-art methods either only use a pre-trained language model or incorporate phonological information as external knowledge. In this paper, we propose a novel end-to-end CSC model that integrates phonetic features into language model by leveraging the powerful pre-training and fine-tuning method. Instead of conventionally masking words with a special token in training language model, we replace words with phonetic features and their sound-alike words. We further propose an adaptive weighted objective to jointly train error detection and correction in a unified framework. Experimental results show that our model achieves significant improvements on SIGHAN datasets and outperforms the previous state-of-the-art methods.

1 Introduction

Spelling errors are common in practice and the errors will be enlarged in the downstream tasks. Therefore, Spelling correction is important to many NLP applications such as search optimization (Martins and Silva, 2004; Gao et al., 2010), machine translation (Belinkov and Bisk, 2017), part-of-speech tagging (Van Rooy and Schäfer, 2002; Sakaguchi et al., 2012), etc. Spelling correction requires a comprehensive grasp of word similarity, language modeling and reasoning, making it one of the most challenging tasks in NLP.

In this paper, we focus on Chinese spelling correction (CSC). Unlike alphabetic languages, Chinese characters cannot be typed without the help of input systems, such as Chinese Pinyin (a pronunciation-based input method) or automatic speech recognition (ASR). Thus typos of similarly pronounced characters occur quite often in Chinese text. According to Liu et al. (2010), 83% of Chinese spelling errors on the Internet results from

Wrong:	ta	**de**	yu	shuo	de	hen	hao
	他	的	语	说	得	很	好
	He	of	language	speak		very	good
Correct:	ta	**de**	yu	shuo	de	hen	hao
	他	德	语	说	得	很	好
	He	German	language	speak		very	good
Predict without phonetic features:	ta	**ying**	yu	shuo	de	hen	hao
	他	英	语	说	得	很	好
	He	English	language	speak		very	good

Figure 1: An example of CSC. Each row contains three lines, Chinese pinyin, Chinese character, and gloss. The Chinese character "德(de, German) is incorrectly typed as its homophone "的(de, of)". The CSC model produces a fluent but incorrect sentence by replacing the character with "英(ying, English)", without considering the phonetic similarity.

phonologically similar characters. As illustrated in Figure 1, the character "德(de, German)" is incorrectly typed as one of its homophone[1] "的(de, of)".

Traditional methods of CSC firstly detect misspelled characters and generate candidates via a language model, and then use a phonetic model or rules to filter wrong candidates (Chang, 1995; Chen et al., 2013; Dong et al., 2016). To improve CSC performance, studies mainly focus on two issues: 1) how to improve the language model (Wu et al., 2010; Dong et al., 2016; Zhang et al., 2020) and 2) how to utilize external knowledge of phonological similarity (Jia et al., 2013; Yu and Li, 2014; Wang et al., 2018; Cheng et al., 2020). The language model is used to generate fluent sentences and the phonetic features can prevent the model from producing predictions whose pronunciation deviates from that of the original word. As illustrated in Fig. 1, the original *Wrong* sentence contains an incorrect word "的(de, of)". The CSC model produces a fluent but incorrect sentence by replacing "的(de,

[*] Corresponding author.

[1] In this paper, we ignore the tone of Pinyin, and use homophone to refer to characters with the same pinyin spellings.

of)” with “英(ying, English)”. However, the pronunciations of these two words are totally different, because the model ignores phonetic features.

Recent studies tackle the issue using deep neural networks. Hong et al. (2019) used a pre-trained language model BERT (Devlin et al., 2019) to generate candidates and train a classifier with phonetic features to select the final correction. Wang et al. (2019) considered CSC as a sequence-to-sequence task and generated candidates from a confusion set [2] instead of the entire vocabulary. These methods take phonetic information as external knowledge but the discrete candidate selection obstructs the language model from learning directly via back-propagation. Zhang et al. (2020) proposed an end-to-end CSC model by modifying the mask mechanism of BERT. However, they did not use any phonological information, which is important for exploring words similarity.

In this paper, we propose a novel end-to-end model for Chinese spelling correction. The model incorporates the phonetic information into language model and leverages the pre-training and fine-tuning framework. Concretely, we first modify the learning task of pre-trained masked language model (Devlin et al., 2019). Rather than replacing characters with an indiscriminate symbol “[MASK]”, we mask characters with pinyin or similar pronounced characters. This enables the language model to explore the similarity between characters and pinyin. Then we fine-tune on error correction data with a model of two networks, a detection network predicts the probability of spelling error for each word, and a correction network generates correction by fusing the word embedding and pinyin embedding with the probabilities as input. We jointly optimize the detection and correction networks in a unified framework.

The contributions of this paper are summarized as follows:

- We propose a novel end-to-end CSC model that incorporates phonetic features into language representation. The model encodes the Chinese characters and Pinyin tokens in a shared space.

- The integration of phonological information greatly facilitates CSC. Experimental results on the benchmark SIGHAN datasets show that our method significantly outperforms the previous state-of-the-art methods.

[2]Confusion set is a set of similar characters.

2 Related work

Earier work on CSC follows the pipeline of error detection, candidate generation, and candidate selection (Wu et al., 2010; Jia et al., 2013; Chen et al., 2013; Chiu et al., 2013; Liu et al., 2013; Xin et al., 2014; Yu and Li, 2014; Dong et al., 2016; Wang et al., 2018). These methods mainly employ unsupervised language models and rules to select candidates.

With the development of end-to-end networks, some work proposed to optimize the error correction performance directly as a sequence-labeling task with conditional random fields (CRF) (Wu et al., 2018) and recurrent neural networks (RNN) (Zheng et al., 2016; Yang et al., 2017). Wang et al. (2019) used a sequence-to-sequence framework with copy mechanism to copy the correction results directly from a prepared confusion set for the erroneous words. Cheng et al. (2020) built a graph convolution network (GCN) on top of BERT (Devlin et al., 2019) and the graph was constructed from a confusion set. Zhang et al. (2020) proposed a soft-masked BERT model that first predicts the probability of spelling error for each word, and then uses the probabilities to perform a soft-masked word embedding for correction. However, they did not use any phonetic information.

Our work is most related to Zhang et al. (2020), but with some important differences. We will further discuss this in Section 3.4.

3 Methods

Formally, the Chinese spelling correction task is to map a sequence $\mathbf{x_w} = (x_{w_1}, x_{w_2}, ..., x_{w_N})$ which may contain spelling errors to another correct sequence $\hat{\mathbf{y}} = (\hat{y}_1, \hat{y}_2, ..., \hat{y}_N)$, where both x_{w_i} and $\hat{y}_i$ $(1 \leq i \leq N)$ are Chinese characters.

We propose an end-to-end CSC model which consists of two components, detection and correction. The detection module takes $\mathbf{x_w}$ as input and predicts the probability of spelling error for each character. The correction model takes the combination of the embedding of $\mathbf{x_w}$ and its corresponding pinyin sequence $\mathbf{x_p} = (x_{p_1}, x_{p_2}, ..., x_{p_N})$ as input and predicts the correct sequence $\mathbf{y}$. We propose a method to fuse $\mathbf{x_w}$ and $\mathbf{x_p}$ embeddings using the probability of spelling error as weights.

Following the pre-train and fine-tune framework,

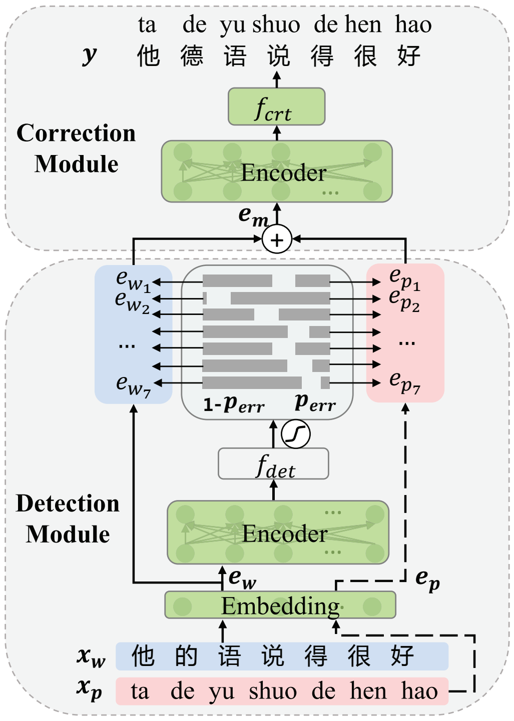

Figure 2: Illustration of our CSC model. Given the input word sequence $\mathbf{x_w}$, the detection module first predicts the probability of error $\mathbf{p_{err}}$ for all the characters. Then the correction module combines the word embedding $\mathbf{e_w}$ and pinyin embedding $\mathbf{e_p}$ to an embedding fusion $\mathbf{e_m}$ and send it to generate the final correction $\mathbf{y}$. The parameters of the Embedding, Encoder, and f_{crt} are initialized by a pre-trained language model with phonetic features. The structure and parameters of the two encoders are identical.

we first pre-train a masked language model, **MLM-phonetics**, by learning to predict characters from similarly pronounced characters and pinyin. Then in fine-tuning, we jointly optimize the the detection and correction modules.

In this section, we first introduce the model architecture (Sec. 3.1), the optimization method (Sec. 3.2) , and the pre-training of **MLM-phonetics** (Sec. 3.3), then summarize the novelty of our method (Sec. 3.4).

3.1 Model Architecture

Fig. 2 shows our model architecture, the lower is the error detection module and the upper is the correction module. Both are built upon transformer.

Detection Module Given a source sequence $\mathbf{x_w} = (x_{w_1}, x_{w_2}, ..., x_{w_N})$, the goal of the detection module is to check whether a character $x_{w_i} (1 \leq i \leq N)$ is correct or not. For this labelling problem, we use class 1 and 0 to label misspelled characters and correct characters, respectively.

We formalize the detection module as follows:

$$\mathbf{y_d} = \text{softmax}(f_{det}(E(\mathbf{e_w}))) \quad (1)$$

where $\mathbf{e_w} = (e_{w_1}, e_{w_2}, ..., e_{w_N})$ is the word embedding of $\mathbf{x_w}$, E is a pre-trained encoder and f_{det} is a fully-connected layer that maps the sentence representation to a binary sequence $\mathbf{y_d} = (y_{d_1}, y_{d_2}, ..., y_{d_N}), y_{d_i} \in \{0, 1\}$.

We use p_{err_i} to denote the probability that character x_{w_i} is erroneous:

$$p_{err_i} = p(y_{d_i} = 1 | \mathbf{x_w}; \theta_d) \quad (2)$$

where θ_d is the parameters of error detection module.

Correction Module The goal of the correction module is to generate correct characters based on the output of the detection module.

We not only use the word embeddings for input, but also use the pinyin embeddings to integrate the phonetic information. Concretely, we first generate the pinyin sequence $\mathbf{x_p}$ using the PyPinyin[3] tool, get the pinyin embedding $\mathbf{e_p}$ from the embedding layer, and fuse it with the word embedding $\mathbf{e_w}$ by linear combination:

$$\mathbf{e_m} = (1 - \mathbf{p_{err}}) \cdot \mathbf{e_w} + \mathbf{p_{err}} \cdot \mathbf{e_p} \quad (3)$$

This combination uses the spelling error probability predicted by the detection module as weights to balance the importance of the semantic feature (character embedding) and phonetic feature (pinyin embedding). We introduce two special cases: If $p_{err_i} = 0$, indicating the character x_{w_i} is detected to be correct, and the model uses only its word embedding in $\mathbf{e_m}$. If $p_{err_i} = 1$, meaning that the character is detected to be erroneous, and the model uses its pinyin embedding.

Finally, the correction results $\mathbf{y}$ is predicted through a fully-conntected layer f_{crt}:

$$\mathbf{y} = softmax(f_{crt}(E(\mathbf{e_m}))) \quad (4)$$

Note that, the parameters of the embedding, the encoder E and the correction network f_{crt} are initialized by **MLM-phonetics**. In the pre-training, **MLM-phonetics** is trained to reconstruct the correct characters from their commonly confused counterparts and their pinyin, thus it can be transformed with the fused embedding.

[3]https://pypi.org/project/pypinyin/

3.2 Jointly Fine-tuning

There are two objectives for our model: to train the detection parameters f_{det} and to adjust the detection and correction modules to achieve an optimal balance. We jointly optimize the detection loss $\mathcal{L}_d$ and the correction loss $\mathcal{L}_c$ that:

$$\mathcal{L}_d = -\sum_i \log p(\hat{y}_{d_i}|\mathbf{x_w};\theta_d) \tag{5}$$

$$\mathcal{L}_c = -\sum_i p(y_{d_i}|\mathbf{x_w};\theta_d) \cdot \log p(\hat{y}_i|\mathbf{e_m};\theta_c) \tag{6}$$

where θ_d and θ_c is the parameter of the detection and correction module, respectively. $\hat{y}_{d_i}$ is the ground-truth detection result and y_{d_i} is the prediction by the detection module, both of them is a binary value of 0 or 1.

In particular, the correction loss is the negative log likelihood weighted by the probability of the detection result, $p(y_{d_i}|\mathbf{x_w};\theta_d) \in (0.5, 1]$. This is to distinguish between the responsibilities of the two tasks. When the detection module gives a low-confidence prediction, that is, $p(y_{d_i}|\mathbf{x_w};\theta_d)$ approaches 0.5, $\mathbf{e_m}$ fuses the semantic features and phonetic features with similar weights. But we hope that the detection module could provide clear judgement of right or wrong, i.e., $p(y_{d_i}|\mathbf{x_w};\theta_d)$ approaches to 1, so that $\mathbf{e_m}$ can be dominated by either semantic features or phonetic features. In such case, the correction of error words will not be interfered by the semantic features in $\mathbf{e_m}$, and vice versa. Therefore, we penalize the low-confidence prediction given by the detection module. Concretely, when the probability of the detection result is low, $\mathcal{L}_c$ decreases and the model will focus more on optimizing $\mathcal{L}_d$. And when the detection probability is high, the model optimizes $\mathcal{L}_d$ and $\mathcal{L}_c$ at balance.

The adaptive weighting objective enables us to jointly train our model with the sum of the two loss functions:

$$\mathcal{L} = \mathcal{L}_d + \mathcal{L}_c \tag{7}$$

We compare different weighting strategies with our adaptive weighting in experiments.

3.3 Pre-training MLM-phonetics

In this section, we introduce our pre-trained language model, **MLM-phonetics**, that 1) integrates phonetic features and 2) solves the problems of using standard masked language model in our CSC architecture.

The pre-train and fine-tune framework (Devlin et al., 2019) has been proven effective in facilitating downstream NLP tasks including sentence classification, question answering, etc. But the inputs of these tasks are of identical distribution with pre-training, while the input sentences in CSC are with errors, which are different from the pre-training samples. Some work thus far side-stepped the input divergence by avoiding to input error sentences directly to pre-trained models. For example, Zhang et al. (2020) use a bidirectional-GRU for error detection before a BERT-based correction network.

In order to take advantage of the pre-training technique, we modify the pre-training task. In the pre-training of a standard mask language model (**MLM-base**), the model is trained by predicting 15% randomly selected characters which are replaced with the [MASK] token, random character, and themselves at the sampling rate of 80%, 10%, and 10%, respectively.

To avoid input divergence and integrate phonetic features, we propose two pre-training replacements: *confused-Hanzi*[4] and a *noisy-pinyin*. We use Figure 3 to illustrate these replacements:

- [MASK] replacement trains the reasoning ability of the language model by restoring masked characters only according to the context.

- *Random Hanzi* replacement trains **MLM-base** to correct words from random ones (e.g, to predict "得(de)" from "不(bu)"), which is a more difficult task compared with correcting from similarly pronounced characters. However, due to the different input distribution, this strategy is of little help to CSC.

- *Same* replacement encourages **MLM-base** to copy the input characters (e.g, the replacement "好(hao)").

- *Confused-Hanzi* replacement trains **MLM-phonetics** to correct words to their commonly confused characters in the confusion set (e.g, to predict "豪(hao)" to "好(hao)"). It provides the model a way to access samples with typo.

- *Noisy-pinyin* replacement trains **MLM-phonetics** to predict the original characters from pinyin of their commonly confused characters in the confusion set (e.g, to predict

[4]*Hanzi* is a transliteration meaning Chinese characters.

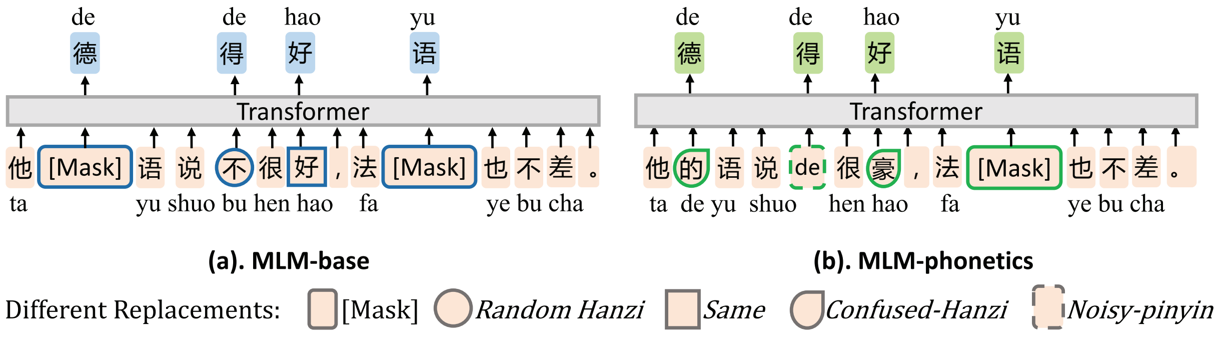
de
德
de
得
hao
好
yu
语
Transformer
他 [Mask] 语 说 不 很 好 , 法 [Mask] 也 不 差 。
ta yu shuo bu hen hao fa ye bu cha
(a). MLM-base
de
德
de
得
hao
好
yu
语
Transformer
他 的 语 说 de 很 豪 , 法 [Mask] 也 不 差 。
ta de yu shuo hen hao fa ye bu cha
(b). MLM-phonetics
Different Replacements: [Mask] Random Hanzi Same Confused-Hanzi Noisy-pinyin

Figure 3: An example of the different replacement strategy for **MLM-base** and **MLM-phonetics**.

"得(de)" from "de"). It helps clustering similarly pronounced characters with their corresponding pinyin tokens.

The first three replacements are used for pre-training standard **MLM-base** and the last two are proposed in our method to model the similarity between characters and pinyin tokens. In the pre-training of **MLM-phonetics**, our data generator randomly chooses 20% of token positions in the training samples. If the i^{th} token is chosen, we empirically replace it with (1) the [MASK] token 40% of the time, (2) the *noisy-pinyin* of this token 30% of the time, and (3) a *confused-Hanzi* from its confusion set 30% of the time[5]. Then **MLM-phonetics** is trained to predict the original sentence from the sentence with replacements.

The two proposed pre-training tasks can smooth out the input divergence between pre-training and fine-tuning the CSC model. The *Confused-Hanzi* replacement simulates the input of the detection module and the two replacements together facilitates the pre-trained model to adapt to the fused embedding (Eq. 3).

3.4 Novelty of our method

Our method is most related to Zhang et al. (2020), but different in the following aspects.

First, our model combine the embedding of pinyin and character to prevent information loss, which is more like the human correction process that predicts correction with the pronunciation of the problematic words. On the contrary, Zhang et al. (2020) has to add residual connection before emitting the final correction, or it will forget the phonetic information of the error words after combining their embedding with [MASK].

[5]For multiple characters in the confusion set, select them/their pinyin randomly in the replacement.

Second, we share the pre-trained encoder in detection and correction by proposing new pre-training tasks, while Zhang et al. (2020) took an un-pre-trained bidirectional-GRU in detection to avoid the input divergence between pre-training and fine-tuning.

Third, we propose an adaptive weighting policy in jointly training the error detection and correction. This policy encourages the model to produce clear detection results, making the fused embedding dominated by either semantic features or phonetic features, which is close to the pre-training task. On the contrary, Zhang et al. (2020) proposed to linearly combine the detection and correction loss with a fixed hyper-parameter.

4 Experiments

We carry out experiments on the SIGHAN dataset, a benchmark for CSC.

4.1 Data Processing

The training set consists of two parts: 1) A pre-training corpus of 0.3 billion Chinese sentences, and 2) A CSC training corpus of 281K sentences pairs. The first corpus is used for pre-training **MLM-phonetics**, and the latter is used to fine-tune the CSC model initialized by **MLM-phonetics**.

For the pre-training corpus, we collect a variety of data, such as encyclopedia articles, news, scientific papers, and movie subtitles from a search engine. The CSC training data used in our experiments is the same as Wang et al. (2019) and Cheng et al. (2020), including three human-annotated training datasets (Wu et al., 2013; Yu et al., 2014; Tseng et al., 2015) and an automatically generated dataset with the approach proposed in Wang et al. (2018)[6].

[6]See Appendix A for training corpus detail.

	Detection			Correction		
	Prec.	**Rec.**	**F1**	**Prec.**	**Rec.**	**F1**
SIGHAN13						
FASPell (2019)	76.2	63.2	69.1	73.1	60.5	66.2
Pointer Networks (2019) (character-level)	56.8	91.4	70.1	79.7	59.4	68.1
*Soft-Masked BERT**	81.1	75.7	78.3	75.1	70.1	72.5
SpellGCN (2020)	80.1	74.4	77.2	78.3	72.7	75.4
ERNIE	76.6	71.9	74.2	73.0	68.5	70.6
MLM-phonetics(Ours)	**82.0**	**78.3**	**80.1**	**79.5**	**77.0**	**78.2**
SIGHAN14						
FASPell (2019)	61.0	53.5	57.0	59.4	52.0	55.4
Pointer Networks (2019) (character-level)	63.2	82.5	71.6	79.3	68.9	73.7
*Soft-Masked BERT**	65.2	70.4	67.7	63.7	68.7	66.1
SpellGCN (2020)	65.1	69.5	67.2	63.1	67.2	65.3
ERNIE	63.5	69.3	66.3	60.1	65.6	62.8
MLM-phonetics(Ours)	**66.2**	**73.8**	**69.8**	**64.2**	**73.8**	**68.7**
SIGHAN15						
FASPell (2019)	67.6	60.0	63.5	66.6	59.1	62.6
Pointer Networks (2019) (character-level)	66.8	73.1	69.8	71.5	59.5	64.9
Soft-Masked BERT (2020)	73.7	73.2	73.5	66.7	66.2	66.4
*Soft-Masked BERT**	67.6	78.7	72.7	63.4	73.9	68.3
SpellGCN (2020)	74.8	80.7	77.7	72.1	77.7	75.9
ERNIE	73.6	79.8	76.6	68.6	74.4	71.4
MLM-phonetics(Ours)	**77.5**	**83.1**	**80.2**	**74.9**	**80.2**	**77.5**

Table 1: The performance on SIGHAN13, SIGHAN14, and SIGHAN15 testset. *Soft-Masked BERT** is our reproduction of *Soft-Masked BERT* using the same training data as in our method, while *Soft-Masked BERT* was trained on an in-house dataset containing 5 million sentences and their counterparts with automatically generated errors, as reported in Zhang et al. (2020), where the authors only provided their results on SIGHAN15.

4.2 Model Settings

We compare our methods with previous state-of-the-art methods:

- *FASPell* (Hong et al., 2019) first generates candidates for each character in the input sentence through a pre-trained MLM, then uses a filtering model with visual and phonetic similarity features to select the best candidate.

- *Pointer Networks* (Wang et al., 2019) uses a seq2seq system based on the constraint that each correct word is contained in the confusion set of the erroneous character.

- *Soft-Masked BERT* (Zhang et al., 2020), for each token in the sentence, linearly combines its embedding with the embedding of [MASK], and predicts the error character based on a fine-tuned masked language model.

- *SpellGCN* (Cheng et al., 2020) incorporates two similarity graphs into a pre-trained sequence-labeling model via graph convolutional network. The two graphs are derived from a confusion set and correspond to pronunciation and shape similarities.

- *ERNIE* (Sun et al., 2020) directly finetunes the standard masked language model on the CSC training data.

- **MLM-phonetics**, our proposed method uses an end-to-end system based on a pre-trained language model with phonetic features.

Pointer Network uses LSTM in both encoder and decoder. All the other methods take CSC as a sequence tagging problem with a pre-trained 12-layer Transformer as encoder. *FASPell* and *Soft-Masked BERT* use the pre-trained BERT, while *ERNIE* and **MLM-phonetics** use the pre-trained ERNIE for initialization[7]. We use the sentence-level and character-level f1-score to evaluate different systems. At the sentence-level, a prediction is considered correct only if all the errors in the sentence are detected or corrected. Therefore, sentence-level evaluation is stricter and results in lower scores. Following Cheng et al. (2020), we

[7]The difference between BERT and ERNIE in fine-tuning is trivial, see Appendix B for detail.

use the scripts in (Hong et al., 2019) to calculate the sentence-level results.

4.3 Overall Results

Table 1 shows the detection and correction performance on three SIGHAN test sets. All the methods provide sentence-level results except *Pointer Network*, which provides the results at the character-level.

It shows that our method, **MLM-phonetics** significantly outperforms the other systems. For example on SIGHAN15, the detection F1-score has 2.5 point improvement (77.7→80.2) and the correction F1-score has 1.6 point improvement (75.9→77.5) compared with the previous best method *SpellGCN*. Our method also achieves over 6 points improvement over *ERNIE* in correction f1-score, verifying the effectiveness of our pre-training strategy.

All the listed methods except *Pointer Networks* use pre-trained model for initialization, but only *FASPell*, *SpellGCN* and our method take phonetic information into consideration. *FASPell* uses isolated phonetic features and language model, which inevitably lead to performance decline. *SpellGCN* incorporates phonetic knowledge into the language model by building a graph convolutional network on the top of BERT. It is proven to be effective, but the graph is derived from a prepared confusion set. Therefore, the performance of the model depends on the completeness of this set. As shown in Table 1, the precision of *SpellGCN* is close to **MLM-phonetics**, but there is a significant gap in the recall. Our method, with the help of additional Pinyin tokens, integrates phonetic features in word embedding, thus increasing the generalization of the model. *Soft-Masked BERT* corrects sentences without phonetic features. Its detection and correction performance is inferior to ours. This may partly due to the lack of phonological similarity, as well as the difference in model architecture. It is also notable that the training data of our method *MLM-phonetics* is consistent with that of *Pointer Networks*, *Soft-Masked BERT**, *SpellGCN* and *ERNIE*, but *Soft-Masked BERT* and *FASPell* use different training data. See Appendix C for some case studies.

4.4 Pre-training Tasks

In order to analyze the effect of the three replacement tasks ([MASK], *Confused-Hanzi*, *Noisy-pinyin*) in the pre-training of **MLM-phonetics**, we take three models for comparison, each trained by only two of the three tasks with equal probability.

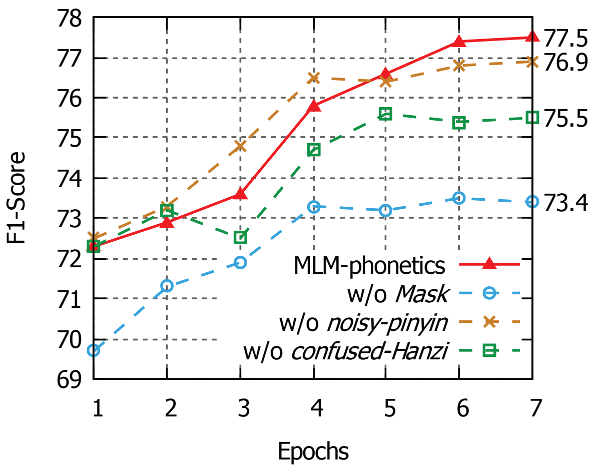

Figure 4: The Sentence-level Correction F1-score of 4 pre-trained models evaluated on SIGHAN15 testset.

During fine-tuning, the testing curves on SIGHAN15 are plotted in Figure 4. **MLM-phonetics** shows the best performance, achieving the correction f1-score of 77.5 in the 7^{th} epoch. However, it's interesting that its performance is inferior to the pre-trained model without (*w/o*) *noisy-pinyin* at the beginning. This is caused by the pre-training & fine-tuning discrepancy.

The model *w/o noisy-pinyin* only learns to predict the original characters from [MASK] and confused-Hanzi in pre-training. So the pinyin embedding has not been initialized until fine-tuning. Therefore, the pinyin embedding can be viewed as noise in the embedding fusion of the fine-tuning stage. Such embedding is close to its pre-training input distribution, thus the pre-trained model *w/o noisy-pinyin* performs good at the beginning. **MLM-phonetics**, on the contrary, is trained to reconstruct words based on either Hanzi embedding or pinyin embedding in the pre-training. But it needs to predict from a fusion of them in fine-tuning, thus it requires longer training time for adaption. As the training continues, the model benefits from the embedding fusion and finally achieves 0.6 points improvements (76.9→77.5) over the pre-trained model *w/o noisy-pinyin*.

Besides, the other two pre-trained models perform relatively low. The pre-trained model *w/o confused-Hanzi* suffers from input divergence in pre-training & fine-tuning. The model is not trained to correct words from spelling errors until the fine-tuning stage. The pre-trained model *w/o* [MASK] performs the worst, which shows the importance of

Objective		Detection			Correction		
		Prec.	**Rec.**	**F1**	**Prec.**	**Rec.**	**F1**
fixed hyperparameter:	$\lambda = 0.2$	75.6	79.1	77.3	71.7	77.5	74.5
$\lambda\mathcal{L}_{uc} + (1-\lambda)\mathcal{L}_d$	$\lambda = 0.5$	77.1	82.0	79.5	74.1	78.4	76.2
(Zhang et al., 2020)	$\lambda = 0.8$	76.1	81.2	78.6	**75.3**	77.9	76.6
Ours: $\mathcal{L}_c + \mathcal{L}_d$		**77.5**	**83.1**	**80.2**	74.9	**80.2**	**77.5**

Table 2: Sentence-level performance of using different objectives on SIGHAN15.

using [MASK] prediction to enhance the semantic comprehension.

4.5 Balance the objective of detection and correction

Next, we explore the impact of the weighting strategy that balances the two objectives in fine-tuning. In our CSC model, both the detection and correction are sequence labeling tasks. We use the detection probability to balance the two tasks, as depicted in Eq.(6). On the contrary, Zhang et al. (2020) balances the two tasks with a fixed hyper-parameter λ: $\lambda\mathcal{L}_{uc}+(1-\lambda)\mathcal{L}_d$, in which $\mathcal{L}_{uc}$ is the un-weighted negative log-likelihood of correction:

$$\mathcal{L}_{uc} = -\sum_i \log p(y_i|\mathbf{e_m};\theta_c) \tag{8}$$

The results of the two strategies are shown in Table 2. Our method is generally better than the results of using a fixed hyper-parameter for combination. Among the three systems with fixed hyper-parameters, the system with $\lambda = 0.8$ achieves the highest correction f1-score and the one with $\lambda = 0.5$ achieves the best detection f1-score. Note that the detection F1-score is evaluated based on the correction result (i.e., only the corrected characters are regarded as detected), rather than based on the prediction of the detection module. Therefore, it's not weird to find that the setting $\lambda = 0.2$ costs much on detection but its detection f1-score is the worst. This also provides us a hint that the detection and correction need to be coordinated. Setting λ to 0.2 may improve the performance of the detection module, but a poor correction module will bring down the final detection performance.

Our method, on the contrary, balances the $\mathcal{L}_c$ and $\mathcal{L}_d$ according to the confidence given by the detection module dynamically and achieves the best performance. Compared with the fixed hyper-parameter strategy with $\lambda = 0.8$, our F1 scores have 1.6 points (78.6→80.2) improvement in detection and 0.9 points (76.6→77.5) improvement in correction, indicating the effectiveness of our dynamic balance strategy in alleviating the unbalanced problem between the two tasks.

4.6 Error Analysis

To analyze the prediction errors, we collect the incorrectly predicted samples and classify them into two classes:

- *Detection Error*: the detection module produces an error prediction, i.e. $y_{d_i} \neq \hat{y}_{d_i}$.
- *Correction Error*: the detection module generates a correct prediction, but the correction module fails to generate the right character, i.e., $y_{d_i} = \hat{y}_{d_i}$ and $y_i \neq \hat{y}_i$.

We summarize the two classes on the SIGHAN15 testset and the proportion of the *Detection Error* and *Correction Error* is 83.6% and 16.4%, respectively. This reveals that most of the false predictions are *Detection Errors*.

We further explore the reason behinds the poor detection performance. Is it mainly because many errors cannot be detected (*false negative* errors), or does the detection module make incorrect predictions of errors (*false positive* errors)? We decompose the 83.6% detection errors into the two types and find that *false negative* errors and *false positive* errors account for 41.1% and 42.5% respectively. The proportions of the two error types are almost equal. A possible reason is that some homonyms are indistinguishable, such as “的”, “地”, and “得”. All of the three characters have the pronunciation of “de” and it makes sense to use any of these candidates in many sentences, phonetically or semantically. This problem has also been proposed in Cheng et al. (2020), which takes further fine-tuning to reduce the indistinguishability. In this case, the detection module produces many predictions that are different from the ground-truth results, affecting the detection performance.

5 Conclusion

In this paper, we propose a novel end-to-end framework for CSC with phonetic pre-training. Inspired by traditional pipeline systems, the model incorporates phonetic information of characters into pre-training. We first pre-train a masked language model with phonetic features to improve the model's ability to understand sentences with misspelling and model the similarity between characters and pinyin tokens. Further, we propose an end-to-end framework to integrate detection and correction in one model. Experiments on a benchmark dataset show that our model significantly outperforms the previous state-of-the-art. The CSC model with phonetic features can be used to reduce errors for speech recognition and translation systems. In the future, we are going to apply CSC to more challenging scenarios, such as streaming ASR error correction for automatic simultaneous translation, as well as variable-length correction.

Acknowledgements

We would like to thank the anonymous reviewers for their insightful comments, and our colleague, Chen Ying, for her advice on writing.

References

Yonatan Belinkov and Yonatan Bisk. 2017. Synthetic and natural noise both break neural machine translation. *arXiv preprint arXiv:1711.02173*.

Chao-Huang Chang. 1995. A new approach for automatic chinese spelling correction. In *Proceedings of Natural Language Processing Pacific Rim Symposium*, volume 95, pages 278–283. Citeseer.

Kuan-Yu Chen, Hung-Shin Lee, Chung-Han Lee, Hsin-Min Wang, and Hsin-Hsi Chen. 2013. A study of language modeling for Chinese spelling check. In *Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing*, pages 79–83, Nagoya, Japan. Asian Federation of Natural Language Processing.

Xingyi Cheng, Weidi Xu, Kunlong Chen, Shaohua Jiang, Feng Wang, Taifeng Wang, Wei Chu, and Yuan Qi. 2020. SpellGCN: Incorporating phonological and visual similarities into language models for Chinese spelling check. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 871–881, Online. Association for Computational Linguistics.

Hsun-wen Chiu, Jian-cheng Wu, and Jason S. Chang. 2013. Chinese spelling checker based on statistical machine translation. In *Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing*, pages 49–53, Nagoya, Japan. Asian Federation of Natural Language Processing.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Shichao Dong, Gabriel Pui Cheong Fung, Binyang Li, Baolin Peng, Ming Liao, Jia Zhu, and Kam-fai Wong. 2016. ACE: Automatic colloquialism, typographical and orthographic errors detection for Chinese language. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*, pages 194–197, Osaka, Japan. The COLING 2016 Organizing Committee.

Jianfeng Gao, Xiaolong Li, Daniel Micol, Chris Quirk, and Xu Sun. 2010. A large scale ranker-based system for search query spelling correction. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 358–366, Beijing, China. Coling 2010 Organizing Committee.

Yuzhong Hong, Xianguo Yu, Neng He, Nan Liu, and Junhui Liu. 2019. FASPell: A fast, adaptable, simple, powerful Chinese spell checker based on DAE-decoder paradigm. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 160–169, Hong Kong, China. Association for Computational Linguistics.

Zhongye Jia, Peilu Wang, and Hai Zhao. 2013. Graph model for Chinese spell checking. In *Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing*, pages 88–92, Nagoya, Japan. Asian Federation of Natural Language Processing.

Chao-Lin Liu, Min-Hua Lai, Yi-Hsuan Chuang, and Chia-Ying Lee. 2010. Visually and phonologically similar characters in incorrect simplified Chinese words. In *Coling 2010: Posters*, pages 739–747, Beijing, China. Coling 2010 Organizing Committee.

Xiaodong Liu, Kevin Cheng, Yanyan Luo, Kevin Duh, and Yuji Matsumoto. 2013. A hybrid Chinese spelling correction using language model and statistical machine translation with reranking. In *Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing*, pages 54–58, Nagoya, Japan. Asian Federation of Natural Language Processing.

Bruno Martins and Mário J Silva. 2004. Spelling correction for search engine queries. In *International Conference on Natural Language Processing (in Spain)*, pages 372–383. Springer.

Keisuke Sakaguchi, Tomoya Mizumoto, Mamoru Komachi, and Yuji Matsumoto. 2012. Joint English spelling error correction and POS tagging for language learners writing. In *Proceedings of COLING 2012*, pages 2357–2374, Mumbai, India. The COLING 2012 Organizing Committee.

Yu Sun, Shuohuan Wang, Yu-Kun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie 2.0: A continual pre-training framework for language understanding. In *AAAI*, pages 8968–8975.

Yuen-Hsien Tseng, Lung-Hao Lee, Li-Ping Chang, and Hsin-Hsi Chen. 2015. Introduction to SIGHAN 2015 bake-off for Chinese spelling check. In *Proceedings of the Eighth SIGHAN Workshop on Chinese Language Processing*, pages 32–37, Beijing, China. Association for Computational Linguistics.

Bertus Van Rooy and Lande Schäfer. 2002. The effect of learner errors on pos tag errors during automatic pos tagging. *Southern African Linguistics and Applied Language Studies*, 20(4):325–335.

Dingmin Wang, Yan Song, Jing Li, Jialong Han, and Haisong Zhang. 2018. A hybrid approach to automatic corpus generation for Chinese spelling check. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2517–2527, Brussels, Belgium. Association for Computational Linguistics.

Dingmin Wang, Yi Tay, and Li Zhong. 2019. Confusionset-guided pointer networks for Chinese spelling check. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5780–5785, Florence, Italy. Association for Computational Linguistics.

Shih-Hung Wu, Yong-Zhi Chen, Ping-che Yang, Tsun Ku, and Chao-Lin Liu. 2010. Reducing the false alarm rate of Chinese character error detection and correction. In *CIPS-SIGHAN Joint Conference on Chinese Language Processing*.

Shih-Hung Wu, Chao-Lin Liu, and Lung-Hao Lee. 2013. Chinese spelling check evaluation at SIGHAN bake-off 2013. In *Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing*, pages 35–42, Nagoya, Japan. Asian Federation of Natural Language Processing.

Shih-Hung Wu, Jun-Wei Wang, Liang-Pu Chen, and Ping-Che Yang. 2018. CYUT-III team Chinese grammatical error diagnosis system report in NLPTEA-2018 CGED shared task. In *Proceedings of the 5th Workshop on Natural Language Processing Techniques for Educational Applications*, pages 199–202, Melbourne, Australia. Association for Computational Linguistics.

Yang Xin, Hai Zhao, Yuzhu Wang, and Zhongye Jia. 2014. An improved graph model for Chinese spell checking. In *Proceedings of The Third CIPS-SIGHAN Joint Conference on Chinese Language Processing*, pages 157–166, Wuhan, China. Association for Computational Linguistics.

Yi Yang, Pengjun Xie, Jun Tao, Guangwei Xu, Linlin Li, and Luo Si. 2017. Alibaba at IJCNLP-2017 task 1: Embedding grammatical features into LSTMs for Chinese grammatical error diagnosis task. In *Proceedings of the IJCNLP 2017, Shared Tasks*, pages 41–46, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Junjie Yu and Zhenghua Li. 2014. Chinese spelling error detection and correction based on language model, pronunciation, and shape. In *Proceedings of The Third CIPS-SIGHAN Joint Conference on Chinese Language Processing*, pages 220–223, Wuhan, China. Association for Computational Linguistics.

Liang-Chih Yu, Lung-Hao Lee, Yuen-Hsien Tseng, and Hsin-Hsi Chen. 2014. Overview of SIGHAN 2014 bake-off for Chinese spelling check. In *Proceedings of The Third CIPS-SIGHAN Joint Conference on Chinese Language Processing*, pages 126–132, Wuhan, China. Association for Computational Linguistics.

Shaohua Zhang, Haoran Huang, Jicong Liu, and Hang Li. 2020. Spelling error correction with soft-masked BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 882–890, Online. Association for Computational Linguistics.

Bo Zheng, Wanxiang Che, Jiang Guo, and Ting Liu. 2016. Chinese grammatical error diagnosis with long short-term memory networks. In *Proceedings of the 3rd Workshop on Natural Language Processing Techniques for Educational Applications (NLPTEA2016)*, pages 49–56, Osaka, Japan. The COLING 2016 Organizing Committee.

A Datasets

All of our used datasets are listed in Table 3. The Pre-training corpus includes 0.3 billion sentences and the remaining four corpora contain 281K <error, correct> sentence pairs in sum[8]. The three SIGHAN datasets are human-annotated and the (Wang et al., 2018) is generated automatically with ASR and OCR technique.

Datasets		#Lines	Avg. Length	#Errors
Training Corpus				
	Pre-training corpus	0.3B	29.37	
	Wang et al. (2018)	271,329	44.4	382,704
	SIGHAN 2013	350	49.2	350
	SIGHAN 2014	6,526	49.7	10,087
	SIGHAN 2015	3,174	30	4,237
	Total	281,379	44.4	397,378
Test sets				
	SIGHAN 2013	1000	74.1	1227
	SIGHAN 2014	1062	50.1	782
	SIGHAN 2015	1100	30.5	715

Table 3: Experimental Data Statistics Information.

B Difference between BERT and ERNIE

We evaluate the performance of two pre-trained models, BERT (Devlin et al., 2019) and ERNIE (Sun et al., 2020), on the SIGHAN testset. For both of them, we use the released model[9] of the base version (12 layers with the hidden size of 768).

The zero-shot performance is listed in Table 4. In this setup, we directly use the released model for error correction without fine-tuning. It shows that ERNIE has a prominent advantage over BERT in both detection and correction. This is caused by a prediction problem of BERT that for most of the time, BERT corrects the first character to be the period symbol ("。"). We guess that this is because of a Chinese data pre-processing bug of BERT, that is, when a paragraph is divided into multiple sentences, it always divides the ending period of a sentence into the beginning of the next one. Therefore, a lot of sentences incorrectly predicts the beginning character to be the period symbol.

Then we finetune the 281K CSC training data on the two pretrained models. Table 5 shows the performance of the two models is basically the same. The difference between BERT and ERNIE is +0.4, -2.0, and +1.6 on SIGHAN13, SIGHAN14, and SIGHAN15, respectively. Therefore the difference between BERT and ERNIE after fine-tuning is trivial.

	Detection			Correction		
	Prec.	**Rec.**	**F1**	**Prec.**	**Rec.**	**F1**
SIGHAN13						
BERT	4.73	4.74	4.74	3.2	3.2	3.19
ERNIE	56.0	40.9	47.3	33.3	24.3	28.1
SIGHAN14						
BERT	4.7	8.8	6.1	2.6	4.9	3.4
ERNIE	63.5	27.2	31.1	18.8	14.1	16.1
SIGHAN15						
BERT	9.4	15.6	11.7	6.3	10.4	7.8
ERNIE	55.8	33.9	42.2	30.8	18.7	23.3

Table 4: The performance of Zero-Shot BERT and ERNIE.

	Detection			Correction		
	Prec.	**Rec.**	**F1**	**Prec.**	**Rec.**	**F1**
SIGHAN13						
BERT	73.3	70.4	71.8	71.7	68.9	70.2
ERNIE	76.6	71.9	74.2	73.0	68.5	70.6
SIGHAN14						
BERT	63.3	71.0	67.0	61.3	68.7	64.8
ERNIE	63.5	69.3	66.3	60.1	65.6	62.8
SIGHAN15						
BERT	68.3	77.8	72.7	65.5	74.6	69.8
ERNIE	73.6	79.8	76.6	68.6	74.4	71.4

Table 5: The performance of pre-trained models with fine-tuning.

C Ablation Study

We compare our method with *ERNIE* and *Soft-Masked BERT* trained on identical datasets shown in Table 3. Table 6 and Table 7 show that *MLM-phonetics* performs better at generating semantically coherent and similar sounding corrections.

[8]The 281K sentence pairs can be downloaded at https://github.com/ACL2020SpellGCN/SpellGCN/tree/master/data/merged.

[9]The released model of ERNIE: https://github.com/PaddlePaddle/ERNIE. The released model of BERT: https://github.com/google-research/bert

Input	ta	men	de	chao	fan	hen	bu	cuo	zai	shuo	ta	men	zuo	de	ga	li	ji	ye	hao	chi
	他	们	的	吵	翻	很	不	错，	再	说	他	们	做	的	咖	喱	鸡	也	好	吃！
	Their			quarrel		is very good,			and the		curry chicken they make							is also delicious!		
ERNIE	ta	men	de	kao	can	hen	bu	cuo	zai	shuo	ta	men	zuo	de	ga	li	ji	ye	hao	chi
	他	们	的	烤	餐	很	不	错，	再	说	他	们	做	的	咖	喱	鸡	也	好	吃！
	Their			roast meal		is very good,			and the		curry chicken they make							is also delicious!		
Soft-Masked BERT*	ta	men	de	chao	fa	hen	bu	cuo	zai	shuo	ta	men	zuo	de	ga	li	ji	ye	hao	chi
	他	们	的	炒	法	很	不	错，	再	说	他	们	做	的	咖	喱	鸡	也	好	吃！
	Their			fried method		is very good,			and the		curry chicken they make							is also delicious!		
MLM-phonetics **(ours)**	ta	men	de	chao	fan	hen	bu	cuo	zai	shuo	ta	men	zuo	de	ga	li	ji	ye	hao	chi
	他	们	的	炒	饭	很	不	错，	再	说	他	们	做	的	咖	喱	鸡	也	好	吃！
	Their			fried rice		is very good,			and the		curry chicken they make							is also delicious!		

Table 6: An example from SIGHAN15 test set. Errors are marked in red and right corrections are in blue. All of the three methods accurately detect the misspelling words, but only *MLM-phonetics* yields the correct result. *ERNIE* changes "吵翻" ("chao fan") to differently pronounced "烤餐" ("kao can") and *Soft-Masked BERT** changes "吵翻" to "炒法" ("chao fa", fried method), which sounds similar but is not so good as "炒饭" (fried rice) in terms of semantic coherence.

Input	ying	xiang	xiao	hai	zi	xiang	de	kuai		xue	xi	guan	li	de	ban	fa
	影	像	小	孩	子	想	得	快	,	学	习	管	理	的	斑	法
	Image		children			think fast,				learn		management			spot	method
ERNIE	ying	xiang	xiao	hai	zi	xiang	de	kuai		xue	xi	guan	li	de	fang	fa
	影	响	小	孩	子	想	得	快	,	学	习	管	理	的	方	法
	Influence		children			think fast,				learn		management			method	
Soft-Masked BERT*	ying	xiang	xiao	hai	zi	xiang	de	kuai		xue	xi	guan	li	de	fang	fa
	影	像	小	孩	子	想	得	快	,	学	习	管	理	的	方	法
	Image		children			think fast,				learn		management			method	
MLM-phonetics **(ours)**	ying	xiang	xiao	hai	zi	xiang	de	kuai		xue	xi	guan	li	de	ban	fa
	影	响	小	孩	子	想	得	快	,	学	习	管	理	的	办	法
	Influence		children			think fast,				learn		management			method	

Table 7: Another example from the SIGHAN15 test set. Again, *MLM-phonetics* predicts phonetically similar and semantically coherent correction. But both *ERNIE* and *Soft-Masked BERT** replace "斑"(ban) with "方"(fang), which is semantically coherent but sounds greatly different.