Pinyin-bert: A new solution to Chinese pinyin to character conversion task

Anonymous ACL submission

Abstract

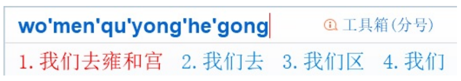
Pinyin to Character conversion (P2C) task is the key task of Input Method Engine (IME) in commercial input software for Asian languages, such as Chinese, Japanese, Thai language, and so on. The dominant technique is Ngram language model together with smoothing technique. However, Ngram model's low capacity limits its performance. Under the trend of deep learning, this paper choose the powerful bert network architecture and propose Pinyin-bert to solve the P2C task, which achieves substantial performance improvement from Ngram model. Furthermore, we combine Pinyin-bert with Ngram model under Markov model's framework and improve performance further. Lastly, we design a way to incorporate external lexicon into Pinyin-bert so as to adapt to the out of domain.

1 Introduction

Some of languages, such as Chinese, Japanese and Thai language, can not be input directly through standard keyboard. User types in these languages via commercial input software, such as Microsoft Input Method (Gao et al., 2002), Sogou Input Method[1], and so on. Pinyin is the official romanization representation for Chinese language. Each pinyin token corresponds to a pronunciation. It's natural for user to type in Chinese by pinyin sequence. For example, as the figure 1 shows, user input a sequence of 'wo men qu yong he gong' from the standard keyboard and the input software convert it into the desired Chinese sentence of "我们去雍和宫 (we are going to Yonghe Lama Temple)". Pinyin to character conversion ("P2C" for short) task is the core task of commercial input software.

Traditionally, there are three steps to solve the pinyin to character conversion task as the figure 2 shows. Firstly, the Chinese character candidates

[1]https://pinyin.sogou.com/


wo'men'qu'yong'he'gong   工具箱(分号)

1. 我们去雍和宫 2. 我们去 3. 我们区 4. 我们

Figure 1: User types in Chinese via pinyin in IME

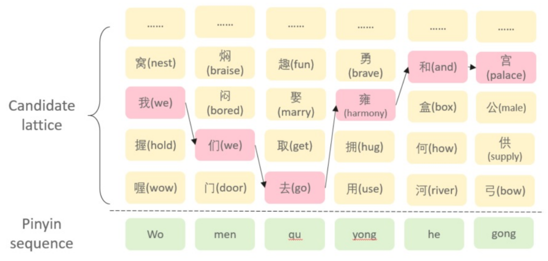


Figure 2: Pinyin to character conversion task

are generated according to the input pinyin tokens. All of the candidates constitute of a lattice. Secondly, the language model provides the probability between Chinese characters, i.e. $P(们 | 我)$. The dominant model in the commercial input method software is Ngram model, with some kind of smoothing algorithm to prevent zero-probability problem. Thirdly, an optimal path in the lattice which usually has the largest probability is found via the Viterbi algorithm. The character sequence on that path is presented to user as the desired conversion result. Ngram model plays an essential role in the above procedures. However, its simplicity and low capacity limit its performance. In the trend of deep learning technique of recent year, more powerful models, such as RNN (Wu et al., 2017), have been applied on the P2C task and achieve significant improvement. This paper chooses the bert network architecture and proposes Pinyin-bert which obtains more improvement. As far as we know, this is the first work to apply bert model to the P2C task.

In the commercial input software, sometimes it's required to combine Pinyin-bert with Ngram model. For example, during the software installation for a new user, the smaller package is preferred due to its fast download speed, which is usually implemented by the Ngram model inside. Then in the idle time, the Pinyin-bert will be downloaded to provide better services. There are two models simultaneously in one software. It's nice to combine them together to obtain more capability. In this paper, we formalize the P2C task from the Bayesian rule and design a method to combine these two models under the Markov model's framework.

Besides the basic lexicon used by IME engine, there are usually various kinds of external lexicon used in the commercial input software so as to provide certain domain knowledge. For example, the computer lexicon provides the phrases in computer domain; the user lexicon provides the user frequently input phrases; the location lexicon, i.e. Beijing, contains the location names where the user is; the old poem lexicon contains the well-known poetry phrases. Together with these external lexicons, the IME engine can provide better service by switching quickly to certain input scenario where there is the domain lexicon. In this paper, we design the method to incorporate external lexicon into Pinyin-bert and improve the performance on the out of domain.

In summary, there are three contributions in this paper. Firstly, we propose Pinyin-bert to solve the P2C task and achieve significant performance improvement. As far as we know, it's the first work to apply the bert model on the P2C task. Secondly, we design a method to combine Pinyin-bert with Ngram model together to obtain further improvement. Lastly, we incorporate external lexicon in Pinyin-bert to improve its performance on the out of domain.

2 Related Works

2.1 language model

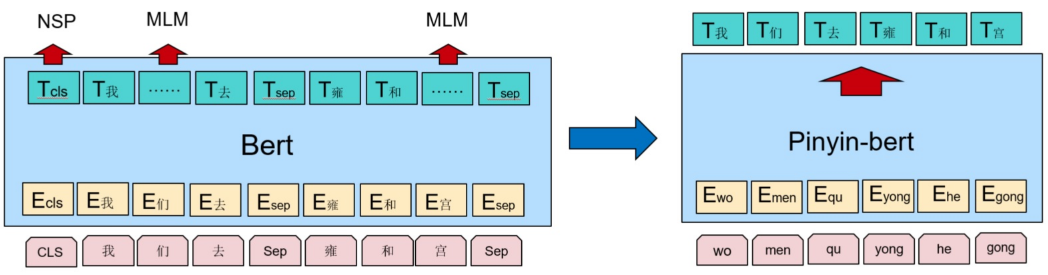
Language model predicts the current word probability by its previous words. As shown in the section 1, it plays an essential role in the IME engine. The dominant model is Ngram model (Bahl et al., 1983) which is trained by the Maximum Likelihood Estimation (MLE) criteria. In practice, it usually encounters the zero-probability problem which can be resolved by smoothing techniques, such as additive smoothing (Laplace, 1825), interpolation smoothing (RJelinek and Mercer, 1980), back-off smoothing (Katz, 1987), Kneser-Ney smoothing (Kneser and Ney, 1995), and so on.

However, the simplicity and low capacity of Ngram model limits its performance. Some more powerful language models are proposed in the trend of deep learning in recent years. For example, RNN obtain more capacity by modeling longer history information (Wu et al., 2017) (Kalchbrenner and Blunsom, 2013) (Mikolov et al., 2010). Variant network architectures are proposed to solve the vanishing gradient problem and the exploding gradient problem, such as LSTM (Malhotra et al., 2015) (Graves et al., 2013), GRU (Cho et al., 2014), and so on. (Yao et al., 2018) replaces the Ngram model with the LSTM model in the IME engine and get performance improvement both in the candidate prompt task and in the P2C task. It further proposes an incremental selective softmax method to solve the LSTM efficiency problem in the Viterbi algorithm. (Meng and Zhao, 2019) applies LSTM in a sequence-to-sequence way in the P2C task, and verify it in a smart sliding input method. (Huang et al., 2018) takes the P2C task as a language translation problem. The neural machine translation model is adopted in which the RNN model is used as encoder and a global attention model is used as decoder.

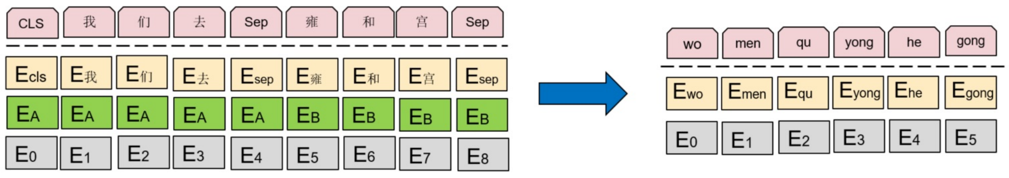
In recent years, Transformer (Vaswani et al., 2017) is proven to be an effective network architecture. From its encoder side, it brings the bert model (Devlin et al., 2018) which obtains the state-of-the-art performances on many language understanding tasks. From its decoder side, it brings the GPT model (Radford and Narasimhan, 2018) which is better in the generation task. Many variant models are proposed (Liu et al., 2019; Yang et al., 2019; Wei et al., 2019; Lan et al., 2020) and the comprehensive experiments are carried out to explore its technique details (Raffel et al., 2020). This paper applies the bert model architecture to the P2C task in the first time as far as we know, and get substantial improvement of performance.

2.2 Enhance bert by lexicon

There are a lot of works to inject lexical information into the bert model. (Cui et al., 2019) proposes to mask the whole word instead of wordpiece in the Masked Language Model ("MLM" for short) task. Spanbert (Joshi et al., 2020) designs the pretrain task especially for a span of text. Zen (Diao et al., 2020) enhances pretrain language model with Ngram representation. (Tian et al., 2020) further leverages the memory network to model wordhood information. Ambert (Zhang and Li, 2020) makes



(a) Train Pinyin-bert directly on the P2C task



(b) The input format between bert and Pinyin-bert

Figure 3: The differences between bert and Pinyin-bert

uses of multi-grained token which takes wordpiece as the fine-grained token and takes whole word or phrase as coarse-grained token. (Li et al., 2020) proposes Multi-source Word Aligned Attention ("MWA" for short) to enhance token representation by word boundary information.

Different from the above works, the intent to use external lexicon in the commercial input software is to adapt the IME engine quickly to a new domain which the lexicon is in. It makes the IME engine flexible to various input scenarios as claimed in the section 1. Therefore, we evaluate the IME engine with external lexicon on the out-of-domain dataset rather than the in-domain dataset as the above methods do. Besides, there are some limitations in the IME engine in practices. Firstly, as the user input scenario changes from time to time, it requires that the external lexicon in the IME engine could be switched dynamically. Thus we can not learn and store the word embedding of external lexicon in advance as the above methods do. Secondly, since the user input is pinyin sequence instead of character sequence, we can not utilize the word boundary information explicitly. This paper firstly designs a method to estimate dynamically the word probability in external lexicon. Then We integrates the estimated result into the Viterbi search process during the P2C task. In that way, it involves the lexical information into the IME engine.

3 Method

In this section, we first describe the implementation of Pinyin-bert in section 3.1. Then we present the way to combine Pinyin-bert with Ngram model in section 3.2. In section 3.3, we describe the method to incorporate external lexicon into Pinyin-bert.

3.1 Pinyin-bert

There are three key factors in Bert. The first one is the network architecture. The second is the pretrain-then-finetune paradigm. The last one is the specific pretrain tasks, i.e. the MLM task and the next sentence prediction ("NSP" for short) task.

The motivation of the pretrain-then-finetune paradigm is to leverage the massive unlabelled text data to provide general knowledge through the pretrain tasks. Then the model is finetuned on the labelled but small scale training data. However, that paradigm is not applicable to the P2C task. For Chinese character corpus, it is easy to convert the text into the pinyin sequence. It is called Text-to-Pinyin conversion task which can achieve more than 99.9% accuracy (Zhang and Laprie, 2003). Based on that, we can convert all of the Chinese text data into the pinyin data, and obtain the parallel corpus. We give up the pretrain procedure in Bert and train Pinyin-bert directly on the P2C task on the massive parallel corpus, as the figure 3(a) shows.

It's also different at the input layer between Pinyin-bert and bert. Firstly, there is not segment embedding since there is no NSP task in

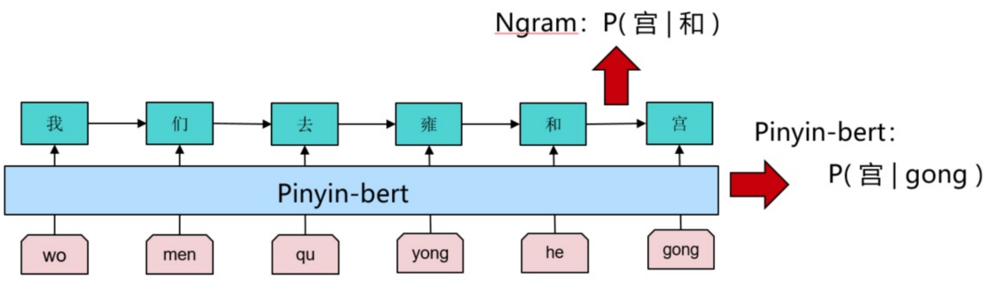


Figure 4: Unify Pinyin-bert with Ngram in the Framework of Markov Model Framework

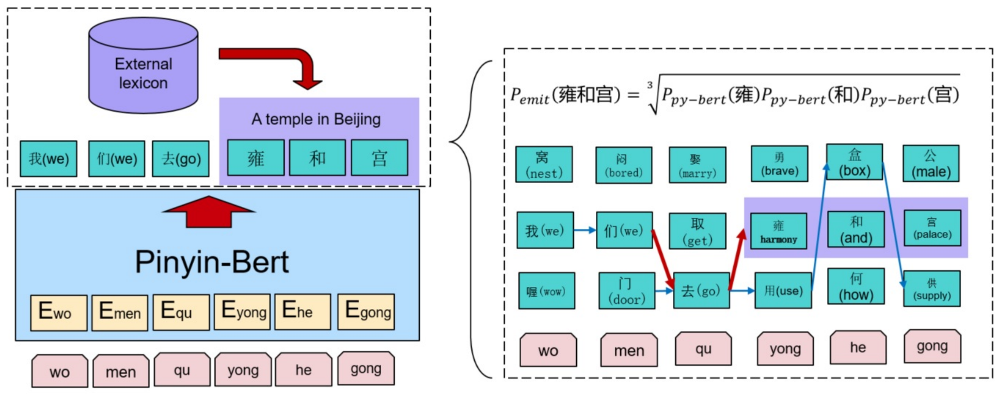


Figure 5: Add lexicon in Pinyin-bert

Pinyin-bert. Secondly, the token embedding is the pinyin embedding instead of the subword embedding since only pinyin can be input into Pinyin-bert. There is only 410 pinyin tokens (without tone) which is about 2x order of magnitude smaller than subword. The scale of embedding layer of Pinyin-bert is much smaller than the normal bert. It's illustrated in the figure of 3(b).

3.2 Combining Pinyin-bert with Ngram

As illustrated in the section 1, it's nice to combine Pinyin-bert and Ngram together in some certain situation of user client. In this section, we firstly formalize the P2C task by the conditional probability. Then we derive the formula in the Bayesian rule. After some proper simplification, it finally finds that these two models are unified together in the framework of Markov model.

Language model estimates the joint conditional probability of $P(c_1...c_i|y_1...y_i)$ in the P2C task. That is to estimate the Chinese character sequence $c_1...c_i$ given the input pinyin sequence $y_1...y_i$. By the Bayesian rule, we can decompose that probability as follows in the formula 1:

$$\begin{aligned}
&P(c_1...c_i|y_1...y_i) \\
&= P(c_1...c_{i-1}, c_i|y_1...y_i) \\
&= P(c_1...c_{i-1}|y_1...y_i) * P(c_i|c_1...c_{i-1}, y_1...y_i) \\
&\approx P(c_1...c_{i-1}|y_1...y_{i-1}) * P(c_i|c_1...c_{i-1}, y_1...y_i) \\
&\approx P(c_1...c_{i-1}|y_1...y_{i-1}) \\
&\qquad * P(c_i|c_1...c_{i-1}) * P(c_i|y_1...y_i)
\end{aligned} \tag{1}$$

As we can see, the joint conditional probability is firstly decomposed into two individual conditional probabilities: $P(c_i|c_1...c_{i-1}, y_1...y_i)$ estimates the current character based on the input pinyin sequence as well as the previous characters. $P(c_1...c_{i-1}|y_1...y_i)$ estimates the previous characters based on the whole input pinyin sequence. In the next, we make simplification twice. Firstly, we simplify $P(c_1...c_{i-1}|y_1...y_i)$ to $P(c_1...c_{i-1}|y_1...y_{i-1})$ since y_i is *in the future* during user inputs. Secondly, we simplify $P(c_i|c_1...c_{i-1}, y_1...y_i)$ into the product of $P(c_i|c_1...c_{i-1})$ and $P(c_i|y_1...y_i)$ as the last line of formula 1 shows. $P(c_i|c_1...c_{i-1})$ is *the state transition probability* in Markov model, and can be estimated by Ngram model. $P(c_i|y_1...y_i)$

is *the emission probability* in Markov model, and can be estimated by Pinyin-bert model. $P(c_1...c_{i-1}|y_1...y_{i-1})$ can be further decomposed into these two kinds of probabilities in a similar way. The figure 4 shows the whole process. With these two kinds of probability, we can search an optimal path as the final conversion result by the Viterbi algorithm in Markov model's framework.

3.3 Incorporating external lexicon

As presented in the section 1, there are many kinds of external lexicons used in the commercial input software, which helps to adapt to various input scenarios of user. We incorporates these external lexicons into Pinyin-bert within Markov model's framework. Specifically there are three steps. Firstly, we recognize the word item according to the external lexicon and add it into the lattice of candidates. Secondly, we estimate the word probability by its compositive characters. Lastly, we get the optimal path together with the added word in the lattice by the Viterbi algorithm.

Here is an example illustrated in the figure 5. "雍和宫(Yonghe Lama Temple)" is a location name from the dictionary of Beijing City. We recognize and add it into the lattice of character candidate, as shown in the right part of figure 5. Then we estimate its probability by the geometric mean of each compositive character as the formula 2 shows. Finally, we search an optimal path by Viterbi algorithm as indicated by the red arrow of the figure 5 's right part. The IME engine returns a sentence on that path to user.

$$P_{emit}(\text{雍和宫}) = \sqrt[3]{P_{Py-bert}(\text{雍}) * P_{Py-bert}(\text{和}) * P_{Py-bert}(\text{宫})} \tag{2}$$

4 Experiment

4.1 Description of Data set and Lexicon

As far as we know, there is no benchmark available to the P2C task. So we build our own data set. The table 1 presents the detailed information. These data are collected from some major Chinese news websites, such as Netease, Tencent News and so on. Total 2.6 million articles are taken as the training corpus, and another 1 thousand disjoint articles are taken as the test corpus. Besides, two additional test corpus are chosen to evalute the performance of out of domain. They are collected from the

corpus	#articles	#disk
news_train	2603869	9.7G
news_test	1000	3.7M
baike_qa_test	49357	30M
society_qa_test	68345	30M

Table 1: The detailed information of corpus

baike website and the society forum [2]. These corpus are firstly segmented into sentences according to a punctuation list including comma, period, and so on. Secondly, we filter out the non-pinyin characters, i.e. number, punctuation, English. Thirdly, these sequences are segmented according to a max length (16 in our experiment) since user types only a few tokens once a time. Lastly, we convert them into the pinyin sequence by PYPinyin [3].

We take *the Table of General Standard Chinese Characters* [4] containing about 6 thousand characters as output in the P2C task. Besides, two external lexicons are adopted. One is *Xiandai Hanyu Changyongcibiao (Common words in Contemporary Chinese, "Common-Words" for short)* containing 0.1 million items (Li and Su, 2008). The other is the network lexicon ("Network-Words" for short) published by Tencent AI Lab [5] containing 8.8 million items.

4.2 Baselines

There are two baseline models. The first one is the bigram model which is popularly used in the commercial input software on mobile device. The second is the LSTM model used in (Yao et al., 2018), which is just a replacement of the bigram model in the IME engine. For LSTM, both the embedding size and the hidden size are 256. The learning rate is $5e^{-4}$. The batch size is 2k and the epoch number is 10. For Pinyin-bert, we follow the specifications of Google from the tiny model to the base model[6]. We also set its max length to 16 instead of 512, so as to be consistent to the training corpus.

[2] https://github.com/brightmart/nlp_chinese_corpus
[3] https://github.com/mozillazg/python-pinyin
[4] https://en.wikipedia.org/wiki/Table_of_General_Standard_Chinese_Characters
[5] https://ai.tencent.com/ailab/nlp/embedding.html
[6] https://github.com/google-research/bert

Model	#Para	Precision	Improved
Bigram	7.4M	85.10%	NV
Lstm	4.2M	89.71%	4.61%↑
Pinyin-Bert-tiny	1.3M	85.18%	0.08%↑
Pinyin-Bert-mini	5.0M	90.57%	5.47%↑
Pinyin-Bert-small	16.5M	95.41%	11.49%↑
Pinyin-Bert-medium	29.1M	95.59%	10.31%↑
Pinyin-Bert-base	91.1M	96.59%	11.49%↑

Table 2: Performances on the P2C task.

Model	Precision	+Bigram	Improved
Pinyin-bert-tiny	85.18%	91.28%	6.10%↑
Pinyin-bert-mini	90.57%	93.68%	3.11%↑
Pinyin-bert-small	95.41%	96.18%	0.77%↑
Pinyin-bert-medium	95.59%	96.63%	1.03%↑
Pinyin-bert-base	96.59%	96.99%	0.40%↑

Table 3: Performances on combining Pinyin-bert with the Bigram model.

4.3 Experiments on the P2C task

The metric to evaluate the P2C task is the character-based precision by comparing the conversion result with the standard text corpus character by character. The experimental results are presented in the table 2. Firstly, we find that the performance can be improved by simply replacing Bigram with LSTM which is more powerful and even much smaller. It proves (Yao et al., 2018) 's conclusion. Secondly, Pinyin-bert-tiny obtains a comparable performance with the Bigram model, even with its one-sixth parameter number. Pinyin-bert-mini achieves substantial improvement (up to 5.47%), still with smaller parameter number. It also outperforms the LSTM model with a comparable model size. These results prove that Pinyin-bert has a better network architecture and more capacity of model. Thirdly, as the scale increases, Pinyin-bert get better and better performance. For example, Pinyin-bert-base gets 96.95% precision and obtains 11.49% improvement from the Bigram model. They can be deployed in the resource-rich environment, i.e. on the cloud.

4.4 Experiments on combining Pinyin-bert with Ngram model

In this section, we combine Pinyin-bert with Bigram together as described in the section 3.2. The experimental results are presented in the table 3. Firstly, the combined model outperforms the single model at each scale of model size. It proves that our method can take effective use of the capability of each sub-model and get better performance. Secondly, not surprisingly, as the scale of Pinyin-bert increases and the more powerful Pinyin-bert becomes, the gain from the combined model becomes smaller. It's OK because the combined model is deployed only on the client device where the resources are very limited so that only the small scale of model is applicable. In the the resource-rich environments, it's not necessary to deploy multiple small models, and we can just deploy the single Pinyin-bert model as large as we can.

4.5 Experiments on incorporating external lexicon

In this section, we incorporate two external lexicons into Pinyin-bert as described in the section 3.3, and evaluate Pinyin-bert on two out of domain corpus. The lexicons and the corpus are described in the section 4.1. The experimental results are presented in the table 4 and 5.

Firstly, comparing the results in the table 2 and 4, we find that the performance of Pinyin-bert decreases obviously on the out of domain corpus. For example, Pinyin-bert-tiny gets 85.18% precision in the news corpus shown in the table 2, while it degrades to 83.40% in the Baike corpus shown in the table 4. Similar conclusions can be drawn on the Society Forum corpus shown in the table 5, and at other scales of Pinyin-bert. It indicates that the problem of out of domain is very severe to the commercial input software. Secondly, the performance of Pinyin-bert is improved by incorporating the external lexicon. For example, the precision of Pinyin-bert-tiny is improved 3.77% with the Common-words lexicon and 5.35% with the Network-words lexicon in the Baike corpus as the table 4 shows. Similar results can be seen in the Society Forum corpus as the table 5 shows. It proves that our method to incorporate external lexicon into Pinyin-bert can help it adapt to out of domain effectively. Thirdly, the improvement from the Network-words lexicon is greater than that from the Common-words lexicon. The former has one order magnitude larger scale and contains much more word items. It indicates that higher quality of lexicon can bring more improvement. Lastly, the gain becomes smaller as the scale of Pinyin-bert increase. However, the external lexicon is deployed only on client device. In the resource-rich environment, we just increase the model scale as much as we can so as to improve the performance, as claimed in the section 4.4.

Model	Baseline	+Common-Words	Improved	+Network-Words	Improved
Pinyin-bert-tiny	83.40%	87.17%	3.77%↑	88.75%	5.35%↑
Pinyin-bert-mini	88.80%	90.04%	1.24%↑	91.24%	2.44%↑
Pinyin-bert-small	91.28%	91.86%	0.58%↑	92.75%	1.47%↑
Pinyin-bert-medium	92.41%	92.88%	0.47%↑	93.57%	1.16%↑
Pinyin-bert-base	93.68%	94.12%	0.44%↑	94.53%	0.85%↑

Table 4: Evaluating Pinyin-bert with external lexicon on the Baike domain.

Model	Baseline	+Common-Words	Improved	+Network-Words	Improved
Pinyin-bert-tiny	80.26%	84.04%	3.75%↑	86.21%	5.95%↑
Pinyin-bert-mini	87.13%	87.89%	0.76%↑	89.52%	2.39%↑
Pinyin-bert-small	90.16%	90.17%	0.01%↑	91.37%	1.21%↑
Pinyin-bert-medium	91.52%	91.35%	0.17%↓	92.38%	0.86%↑
Pinyin-bert-base	93.16%	92.97%	0.19%↓	93.61%	0.45%↑

Table 5: Evaluating Pinyin-bert with external lexicon on the Society domain.

5 Conclusions

This paper proposes Pinyin-bert to solve the P2C task which is the core task to the commercial input software of Asian languages. As far as we know, it's the first work to apply the bert architecture on the P2C task. Pinyin-bert achieves significant performance improvements from the base models, such as Ngram and LSTM. Moreover, we combine Pinyin-bert with Ngram in a Markov model's framework and obtain further improvement. Lastly, we design the method to incorporate external lexicon into Pinyin-bert so as to adapt it to out of domain.

References

Lalit R. Bahl, Frederick Jelinek, and Robert L. Mercer. 1983. A maximum likelihood approach to continuous speech recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 5(2):179–190.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. 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 1724–1734. ACL.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Ziqing Yang, Shijin Wang, and Guoping Hu. 2019. Pre-training with whole word masking for chinese BERT. *CoRR*, abs/1906.08101.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Shizhe Diao, Jiaxin Bai, Yan Song, Tong Zhang, and Yonggang Wang. 2020. ZEN: pre-training chinese text encoder enhanced by n-gram representations. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 4729–4740. Association for Computational Linguistics.

Jianfeng Gao, Joshua Goodman, Mingjing Li, and Kai-Fu Lee. 2002. Toward a unified approach to statistical language modeling for chinese. *ACM Trans. Asian Lang. Inf. Process.*, 1(1):3–33.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey E. Hinton. 2013. Speech recognition with deep recurrent neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pages 6645–6649. IEEE.

Yafang Huang, Zuchao Li, Zhuosheng Zhang, and Hai Zhao. 2018. Moon IME: neural-based chinese pinyin aided input method with customizable association. In *Proceedings of ACL 2018, Melbourne, Australia, July 15-20, 2018, System Demonstrations*, pages 140–145. Association for Computational Linguistics.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Trans. Assoc. Comput. Linguistics*, 8:64–77.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October*

2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL, pages 1700–1709. ACL.

S. M Katz. 1987. Esitmation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech and Signal Processin*, 35:400–401.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *1995 International Conference on Acoustics, Speech, and Signal Processing, ICASSP '95, Detroit, Michigan, USA, May 08-12, 1995*, pages 181–184. IEEE Computer Society.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

P. S. Laplace. 1825. *Philosophical Essay on Probabilities*, 5 edition. Springer Verlag.

Xingjian Li and Xinchun Su. 2008. *Xiandai Hanyu changyongcibiao*. The Commercial Press, Beijing.

Yanzeng Li, Bowen Yu, Mengge Xue, and Tingwen Liu. 2020. Enhancing pre-trained chinese character representation with word-aligned attention. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 3442–3448. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. 2015. Long short term memory networks for anomaly detection in time series. In *23rd European Symposium on Artificial Neural Networks, ESANN 2015, Bruges, Belgium, April 22-24, 2015*.

Zhen Meng and Hai Zhao. 2019. A smart sliding chinese pinyin input method editor for touchscreen devices.

Tomás Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September 26-30, 2010*, pages 1045–1048. ISCA.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pretraining.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

F. RJelinek and R. L Mercer. 1980. Interpolated estimation of markov source parameters from sparse data. In *Proceedings of the Workshop on Pattern Recognition in Practice, North-Holland, Amsterdam,*, pages 381–397. The Netherland.

Yuanhe Tian, Yan Song, Fei Xia, Tong Zhang, and Yonggang Wang. 2020. Improving chinese word segmentation with wordhood memory networks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8274–8285. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Junqiu Wei, Xiaozhe Ren, Xiaoguang Li, Wenyong Huang, Yi Liao, Yasheng Wang, Jiashu Lin, Xin Jiang, Xiao Chen, and Qun Liu. 2019. NEZHA: neural contextualized representation for chinese language understanding. *CoRR*, abs/1909.00204.

Lin Wu, Michele Haynes, Andrew Smith, Tong Chen, and Xue Li. 2017. Generating life course trajectory sequences with recurrent neural networks and application to early detection of social disadvantage. In *Advanced Data Mining and Applications - 13th International Conference, ADMA 2017, Singapore, November 5-6, 2017, Proceedings*, volume 10604 of *Lecture Notes in Computer Science*, pages 225–242. Springer.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5754–5764.

Jiali Yao, Raphael Shu, Xinjian Li, Katsutoshi Ohtsuki, and Hideki Nakayama. 2018. Real-time neural-based input method. *CoRR*, abs/1810.09309.

Sen Zhang and Yves Laprie. 2003. Text-to-pinyin conversion based on contextual knowledge and d-tree for mandarin. In *IEEE International Conference on Natural Language Processing and Knowledge Engineering, NLP-KE 2003, Beijing, China, 2003*.

Xinsong Zhang and Hang Li. 2020. AMBERT: A pre-trained language model with multi-grained tokenization. *CoRR*, abs/2008.11869.