On the Effectiveness of Statistical Models on the One-row Keyword problem

Luu Huu Phuc

NLP class 2020, Kyoto University

Abstract. We investigate the effectiveness of simple statistical models in solving the problem of one-row keyboard. With a training corpus of only 20,000 sentences extracted from Jack London's novels, we show that simple unigram and bigram models can achieve 74% and 90% conversion accuracy, respectively.

1 Introduction[1]

Since the birth of modern computer, computers have become more and more convenient, and have been used widely by almost everybody. The popularity of computers leads to an increasing demand for keyboards. Keyboards of every size and type can be seen anywhere and can be easily obtainable with reasonable prices. However, cheaper prices also mean poorly-built keyboard. It is not surprised that a keyboard has many broken keys just a few days after being bought. Most of the time, only the number keys and the space key survive and work correctly. Instead of throwing away the broken keyboard and buying a new one, utilizing keyboards with remaining keys can bring huge financial benefits. This issue motivates us to investigate a new input method that requires only one row of keys (i.e. the number keys) and a space key instead of the whole keyboard. We call the problem with this constraint on the keys as the "One-row keyboard" problem.

2 Problem Setting and Method

In the problem setting of the "One-row keyboard" problem, each number key corresponds for the "broken" keys below it. Specifically, characters in 1qazQAZ, 2wsxWSX, 3edcEDC, 4rfvRFV, 5tgbTGB, 6yhnYHN, 7ujmUJM', 8ikIK, 9olOL, 0pP- are represented by key 1, 2, 3, 4, 5, 6, 7, 8, 9, and 0[2], respectively. The goal is to correctly convert a sequence of numbers and space character back to a original sentences. For example, given the sequence "1 83659143 2856 9696 563 677534 492 163 1 20133 514", we want to convert it to a sentence like "A

[1] For anyone who reads this "paper" by accident, this is just a funny "paper" I wrote as an assignment for my NLP class. Please do not take it seriously! The code and result analysis are real though. :)

[2] " ' " is represented by 7, and '-' is represented by 0

keyboard with only the number row and a space bar". In the following, we call a sequence of numbers such as "492" a "pronunciation" and a word such as "row" a "word".

To tackle the problem, we propose two simple statistical models using unigram and bigram models as follows.

2.1 A unigram model

The unigram model is simple and use 1-gram based on pairs of word and pronunciation $u =< w, y >$. Mathematically, we believe that this model is similar to the SIMPLE model proposed in Mori et. al (2010)[3] . For a given pronunciation y, we predict the corresponding word w such that $w = \arg\max_w P(w|y)$. Using Bayesian rule, we have $P(w|y) \propto P(y|w)P(w)$. Here, $P(w)$ is calculated by a unigram model trained on a corpus, and $P(y|w)$ is a word-pronunciation model estimated on the corresponding annotated corpus.

2.2 A bigram model

In the bigram model, given a previous word w_1 and a pronunciation y, we want to predict a word w such that $w = \arg\max_w P(w|y, w_1)$. We approximate $P(w|y, w_1)$ with $P(y|w) * P(w|w_1)$, so the goal is to find $w = \arg\max_w P(y|w) * P(w|w_1)$. Here, $P(w|w_1)$ is a bigram model, and $P(y|w)$ is a word-pronunciation model. Both are estimated on a corpus and a corresponding annotated corpus.

Beside the vocabulary of the training corpus, we add a special starting symbol BTS at the beginning of each sentence in the corpus.

3 Evaluation

3.1 Training Corpus and Experiment Settings

For training the models, we prepare a corpus by combining sentences in four novels of the American novelist Jack Lodon, which are "The Call of the Wild", "White Fang", "The Iron Heel", and "The Sea-Wolf". The resulting corpus contains around $20,000$ sentences and $285,000$ words with a vocabulary of almost $18,000$ words. The corpus is then annotated in the form "pronunciation/word" such as "26853/White 4165/Fang 39793/could 233/see 6956865/nothing 316534972/dangerous 86/in 5615/that".

We divide the corpus into a training corpus and a testing corpus with the ratio of $9 : 1$. When training the bigram model, we add a special starting symbol BTS at the beginning of each sentence. In such case, the annotated corpus will look like "BTS/BTS 26853/White 4165/Fang ... 86/in 5615/that".

When a pronunciation y of an unknown word w is given, the model will return "T.T" (crying face) symbol denoting the unknown word.

[3] http://plata.ar.media.kyoto-u.ac.jp/mori/research/topics/KKC/

Table 1. The conversion accuracy (%) of both models

Model	Train corpus	Test corpus
Unigram	76.99±(5.03)	74.41±(4.97)
Bigram	**98.55±(0.04)**	**90.10±(1.36)**

3.2 Experiment Results

We repeat the process of dividing the corpus, training and testing 50 times for both order and report the conversion accuracy in Table 1. Here, the conversion accuracy is calculated as the percentage of pronunciation y being correctly converted to its corresponding words w. The code and dataset can be found in the author github[4].

As demonstrated in Table 1, a simple unigram model can achieve the accuracy of 74% on the test corpus, while a slightly more complicated bigram model achieves a very high accuracy of 90%. The bigram model outperforms the unigram on both training and testing accuracy, and is very robust as the standard deviation is significantly smaller than that of the unigram (0.04 vs 5.03, and 1.36 vs 4.97).

This result is not surprised because the bigram model can consider the context (a previously appeared word w_1) to better predict a word w from a pronunciation y. This advantage is shown clearly when we investigate the differences of sentences converted by both models. The unigram model cannot understand that a word in the beginning of a sentence ("**our**") should have the first letter capitalized, and a word in the middle of a sentence ("**Everywhere**", "**Go**") should not be in capital. More over, the unigram could not learn from the context, i.e. the previous word. It does not "know" that the word after "**white**" should be "**fang**" as in "**White Fang**" - the name of the main character. And it does not "know" that the word after a verb ("**continued**") should be a preposition ("**to**").

BTS Our comrades sought him everywhere (**source**)
BTS **our** comrades sought him **Everywhere** (**unigram**)
BTS Our comrades sought him everywhere (**bigram**)

BTS White Fang hanging limply continued to cry (**source**)
BTS **white rang** hanging limply continued **Go** cry (**unigram**)
BTS White Fang hanging limply continued to cry (**bigram**)

Interestingly, sometimes when there are "errors" in the source corpus, the bigram model "corrects" them. For examples, it re-capitalizes the first letter of the starting word of a sentence, e.g. "**she**" → "**She**", and "**and**" → "**And**".

[4] code and dataset at: https://github.com/phucdoitoan/Unigram-and-Bigram-models

BTS **she** interrupted with that softness ... (**source**)
BTS She interrupted with that softness ... (**bigram**)

BTS **and** instead of this being joyful for us it will be ... (**source**)
BTS And instead of this being joyful for us it will be ... (**bigram**)

We wished to compare our simple models with KyTea; however, due to some incompatibility between our OS system (Ubuntu 18.04.4) and the package[5], we could not be able to install and test it.

4 Conclusion

We have proposed a two simple statistical models using unigram and bigram to tackle the "One-row keyboard" problem. Through extensive expriments, we show that with a considerably small training corpus ($20,000$ sentences - $285,000$ words), the two models can achieve high conversion accuracy of 74% and 90%. For future works, we would like to extend the models to 3-gram and more. We also want to optimize our code implementation (the present one is in python and jupyter notebook) in order to train the models on larger corpus.

5 Acknowledgement

We would like to thank our instructor, professor Shinsuke Mori, for the profound instructions. All of the models in this paper are based on the materials from his Natural Language Processing course in Kyoto University, Japan.

[5] https://github.com/neubig/kytea/issues/27