Evaluations on Several Smoothing Methods for Chinese Language Models

1Huang Feng-Long, 2Yu Ming-Shing and 3Hwang Chien-Yo
1Department of Computer Science and Information Engineering, National United University, MaioLi, 360, Republic of China
2Department of Computer Science and Engineering, National Chung-Hsing University, Taichung 402, Republic of China

Abstract: In this study, several smoothing methods for language models on Chinese corpus with various sizes are evaluated and analyzed. Basically, there are two phases for smoothing procedures (1) Discounting and (2) Redistributing. Ten models are generated on various size of corpus from 30-300 M Chinese words. We evaluated several smoothing methods for statistical language models. In our experiments, four smoothing methods, Winter-Bell C (WB-C) and our proposed YH-A and YH-B smoothing method, are evaluated for inside testing and outside testing. Based on empirical observations, our YH-B smoothing is superior to WB-C for the TrM models with size between 30 and 90 M.

Key words: Evaluation, language model, smoothing method, cross entropy, perplexity

INTRODUCTION

In many domains of Natural Language Processing (NLP), such as machine translation (Brants et al., 2007) and speech recognition (Jelinek, 1997), the statistical Language Models (LMs) (Napitali et al., 2010) always plays an important role. Data sparseness has been an inherent issue of statistical language models and the smoothing method is usually used to resolve the zero count problems for unknown events. As shown in Fig. 1 of a speech recognition system, the P(W) is the conditional probability of a word sequence W given a speech data S, where W = w1,w2,...,wn is a possible translation of texta, m is word number of M. The sequence W can be predicted as a final target.

Language models: The statistical language models have been widely used in NLP. Supposed that W = w1, w2, w3, ..., wn, where wi and n denote the the ith Chinese character and its number in a sentence (0 ≤ i ≤ n). P(W) = P(w1, w2, ..., wn), the probability can be calculated by using chain rules:

\[ P(w_{i:n}) = P(w_i)P(w_{i+1} | w_i)P(w_{i+2} | w_{i+1}) \cdots P(w_{n} | w_{n-1}) \]

\[ = \prod_{j=i}^{n} P(w_j | w_{j-1}) \]  

where, \( w_1, w_2, w_3, ..., w_{n-1} \).

In general, unigram, bigram and trigram (3 ≤ N) are generated. N-gram model calculates P(.) of Nth events by the preceding N-1 events, rather than the string \( w_1, w_2, w_3, w_{n-1} \).

N-gram models: Basically, N-gram is so-called N-1) th-order Markov model which calculate conditional probability of successive events: calculate the probability of Nth event while preceding (N-1) event occurs.

Basically, N-gram Language Model is simplified expressed as follows:

\[ P(w_{i:n}) = \prod_{j=i}^{n} P(w_j | w_{j-1}) \]  

\[ P(w_i | w_{i-1}) = \frac{C(w_{i-1}, w_i)}{\sum_{w \in \text{vocab}} C(w_{i-1}, w)} \]

where, \( C(w) \) denotes the counts of event \( w \) occurring in dataset.

In Eq. 3 above, the obtained probability \( P(.) \) is so called Maximum Likelihood Estimation (MLE). The
category with maximum probability $P_{\text{end}}(*)$ will be the target and then the correct pronunciation with respect to the polyphonic character can be decided further.

**Unknown events-zero count issue:** As shown in Eq. 3, $C(.)$ of a novel (a unknown event) which don't occur in the training corpus, may be zero because of the limited training data, infinite language and its expansion of language. It is always a hard work for us to collect sufficient datum. The potential issue for MLE is that the probability for unseen events is exactly zero. This is so-called the zero-count problem (Witten and Bell, 1991; Katz, 1987). It is obvious that zero count always leads to the issue of zero probability of $P(.)$ in Eq. 2-3. Therefore, the smoothing methods are needed and exploited to alleviate the zero-count issue for statistical language models.

**PROCESSES OF SMOOTHING METHODS**

As described above, the zero count issue of unknown events will lead to the degradation of language models; therefore we need the smoothing methods to alleviate the situation. The idea of smoothing processes is to adjust the total probability of seen events to that of unseen events, leaving some probability mass (so-called escape probability, $P_{\text{esc}}$) for all the unseen events.

Smoothing algorithms (Jurafsky and Martin, 2000; Gale and Geoffrey, 1995) can be considered as discounting some counts of seen events in order to obtain the escape probability $P_{\text{esc}}$. And then $P_{\text{esc}}$ will be assigned into unseen events based on the smoothing algorithm. The adjustment of smoothed probability for all possibly occurred events involves discounting and redistributing processes.

**Discounting process:** Based on the statistical feature, the probability of all seen and unseen (unknown) events is summed to be unity (one). First operation of smoothing method is the discounting process which discount the probability of all seen events. It means that the probability of seen events will be decreased a bit. In the process, there are two issues:

- How to discount the probability of seen events with various count $c$, $c > 1$. Whether the discounted probability from the seen events with count $c$ is uniform or not will affect the performance of statistical language models
- The effectiveness between the size of escape probability and performance of language models

The adjustment processes can be usually divided into two categories: static and dynamic. Static smoothing methods, for the most smoothing methods, discount the probability of events based on the events occurrences in models. However, dynamic smoothing method, i.e., cache-based language, discounts the probability based on the occurrences for all seen events in both cache and models.

**Redistributing process:** In this operation of smoothing algorithm, the escape probability discounted from all seen events will be redistributed to unseen events. The escape probability is usually shared by all the unseen events. That is, the escape probability is redistributed uniformly to each unseen event, $P_{\text{esc}}/U$, where $U$ is the number of unseen events. On the other hand, each unseen event obtains same probability.

The redistribution of most well known smoothing methods, such as Add-one, Absolute discounting, Good-Turing (Gale and Geoffrey, 1995; Good, 1953), Delete interpolation, Back-off (Kneser and Ney, 1995) and Witten-Bell smoothing (Ostrogochns et al., 2013) is uniform for all unseen events. It is a possible factor that affects the performance of smoothing algorithm. There are few previous works to discuss how to redistribute the escape probability.

**SMOOTHING METHODS**

In this Section, several well-known smoothing methods will be presented. We also proposed two novel methods. All these methods will be evaluated in next section.

**Witten-Bell method:** In the study, we discussed two of five smoothing schemes: methods A and C (called WB-A and WB-C), introduced by Witten-Bell (Ney and Itzen, 1991). Previous study was in (Ostrogochns et al., 2013).

**Method A:** In this method, just one count is allocated to the probability that an unseen bigram will occur next. The probability mass $P_{\text{esc}}$ assigned to all unseen bigrams can be summed up to $1/(N+1)$. The smoothed probability $P^*$ can be expressed as:

$$P_c'(w_{n1}, w_i) = \begin{cases} 1 & \text{for } c(w_{n1}) = 0, \\ \frac{1}{U(N+1)} & \text{for } c(w_{n1}) > 1, \\ \frac{c(w_{n1})}{N+1} & \text{for } c(w_{n1}) \geq 1. \end{cases}$$

3686
Method C: It is more complex than the additive discount technique. The basic concept is recurring.

The W-BC is described as:

\[ c_i' = \begin{cases} \frac{S \cdot N}{U \cdot N + S} & \text{if } c_i = 0 \\ \frac{N}{N + S} & \text{if } c_i > 0 \end{cases} \]  

where, \( U, S \) and \( N \) denote the types of all possible unseen bigrams, seen bigrams in training corpus and the number of all the seen bigrams in training corpus, respectively.

The discounted probability will be expressed for seen bigrams as:

\[ P_{w_i}^* = \frac{c_i'}{N + S} \quad \text{if } c_i > 0 \]  

where, the probability mass \( P_{w_i}^* \) for all unseen bigrams assigned by W-B C is obtained as following:

\[ \sum_{c_i=0} P_{w_i}^* = \frac{S}{N + S} \quad \text{if } c_i = 0 \]  

where, the probability for each unseen bigram will be derived from Eq. 11 divided uniformly by \( U \):

\[ P_{w_i}^* = \frac{1}{U} \cdot \frac{S}{N + S} \]  

where, as shown in Eq. 12, it is obvious that the redistributed count \( c_i^* \) for each bigram which doesn’t appear in corpus is equal to \( S/U \). On other word, the size of \( c_i^* \) is subject to the ratio of \( S \) and \( U \). The ratio may be greater or less than 1, depending on the value of \( S \) and \( U \).

Yu-huang A(YH-A):

Basic concept: In case for a bigram, our method YH-A calculates the smoothed probabilities as:

\[ Q(w_{i-1}w_i) = \begin{cases} \frac{d_{c_i}}{U(N+1)} & \text{for } c(w_{i-1}) = 0, \\ \frac{c(w_{i-1}) N + 1 - d_{c_i}}{N} & \text{for } c(w_{i-1}) > 1, \end{cases} \]  

where, \( d_{c_i} \) denotes a constant (0<\( d_{c_i} \)<1) and independent of \( U \).

When computing the smoothed probability, our proposed method don’t employ interpolating scheme to combine the high order models and lower order models. As shown of Eq. 13, \((N+1-Ud_{c_i})(N+1)\) is the normalization factor for \( Q^* \) of seen bigrams. The probabilities for all the seen bigrams will be discounted by the normalization factor and then the accumulated probability then is re-distributed to the unseen bigrams. All the unseen bigrams will share uniformly the distribution mass \( d_{c_i}(N+1) \):

\[ \sum_{c_i=0} p_{w_i}^* = \frac{d_{c_i}}{N + 1} \quad \text{for } c_i = 0 \]  

where, Eq. 14 of Y-H Ais similar to Eq. 11 of WB-Ain (Ney and Essin, 1991). Instead of the constant 1 of numerator in Eq. 15, it is replaced with a constant \( d_{c_i} \) (0<\( d_{c_i} \)<1). It is necessary that we will evaluate \( d_{c_i} \) with respect to perplexity for language models in the next section. Hence, the better \( d_{c_i} \) for lower perplexity can be found.

Yu-huang method smoothing (YH-B): Our proposed smoothing method YH-B (YH-B) describes other smoothing scheme; in which the probability mass for unseen bigrams is assigned \( Ud_{c_i}(N+1) \). Consequently, it varied with \( N \) and \( U \); the number of training data and types of unseen bigrams.

The basic concept of our smoothing YH-B can be described in detail as follow. The smoothed probabilities will be calculated as follows:

\[ P(w_{i-1}w_i) = \begin{cases} \frac{d_{c_i}}{(N+1)} & \text{for } c(w_{i-1}) = 0, \\ \frac{c(w_{i-1}) N + 1 - Ud_{c_i}}{N} & \text{for } c(w_{i-1}) > 1, \end{cases} \]  

\[ d_{c_i} = \min\left(\frac{N}{N + 2U}, \frac{N + 2}{2U}\right) \]  

where, calculating the smoothed probability \( P^* \), our proposed method don’t employ interpolating scheme to combine the high order models with lower order models. As shown of Eq. 13, \((N+1-Ud_{c_i})(N+1)\) is the normalization factor of \( Q^* \) for seen bigrams. The probabilities \( Q \) will be discounted by the normalization factor and then remained \( Q^* \) are redistributed to unseen bigrams; which share uniformly the distributed probability mass \( Ud_{c_i}/(N+1) \):

\[ \sum_{c_i=0} p_{w_i}^* = \frac{Ud_{c_i}}{(N+1)} \quad \text{for } c_i = 0 \]  

Models evaluation cross-entropy and perplexity: Two commonly used schemes for evaluating the quality of language model LM are referred to the entropy and perplexity (Ostrogonac et al., 2013; Brown et al., 1992).
Supposed that a sample $T$ is consisted of several events $e_1, e_2, ... , e_n$ of mstrings. The probability $P$ for a given testing sample $T$ is calculated as following:

$$P(T) = \prod_{i=1}^{n} P(e_i)$$  \hspace{1cm} (18)

where, $P(e_i)$ is the probability for the event $e_i$ and $E(T)$ can be regarded as the coded length in testing datasets:

$$E(T) = -\sum_{x} P(x) \log_2 P(x)$$  \hspace{1cm} (19)

$$= -\sum_{x} P(e_i) \log_2 P(e_i)$$

$$PP(T) = 2^{-E(T)}$$  \hspace{1cm} (20)

where, $E(T)$ and $PP(T)$ denote the entropy (log model probability) and perplexity for testing dataset $T$, respectively. $E_{\text{min}}$ stands for the minimum entropy for a model.

The perplexity $PP$ is usually regarded as the average number for selected number which will be the possible candidates referred to a known sequence. When a language model is employed to predict the next appearing word in the current given context, the perplexity is adopted to compare and evaluate n-gram statistical language models.

In general, lower entropy $E$ leads to lower $PP$ for the language models. It means that the lower $PP$, the better performance of language models. Therefore, perplexity is a quality measurement for $LM$. While two language models, $LM_i$ and $LM_j$, are compared, the one with lower perplexity is the better language representation and commonly provides higher performance.

In fact, the probability distribution for testing language models is usually unknown. The model which can predict better the next occurring event always achieves lower cross entropy. In general $CE = E, E$ denotes the entropy using same language model $M$ for training and testing models. Based on the Shannon-McMillan-Breiman theorem (Algoet and Cover, 1988; Aratti, 2013), PP Evaluation can be expressed as following:

$$CE(p,M) = \lim_{n \to \infty} -\frac{1}{n} \log M(w_1, w_2, ... , w_n)$$  \hspace{1cm} (19)

**EXPERIMENTS AND EVALUATION**

**Training chinese corpus-textual gigaword:** Chinese GigaWord (CGW) is the Chinese corpus collected from several world news databases and issued by Linguistic Data Consortium (LDC). In the study, we adopted the CGW 3.0 published on September 2009. The CGW sources are Agence France-Presse, Central News Agency of Taiwan, Xinhua News Agency of Beijing and Zaobao Newspaper of Singapore.

**Models generation for evaluations:** In the study, we will create 10 Unigram language models with Chinese words for experiments. At first, we read in randomly the study of Chinese words from CGW corpus, a language model $LM_i$ will be created for the first $3 \times 10^5$ (30M) Chinese words.

In the following, the other new model $LM_i$ can be created consequently for the next $3 \times 10^7$ Chinese words. In other words, $LM_i$ is consisted of first $6 \times 10^7$ (60M) Chinese words of CGW, first half of which is also used to create $LM_i$.

**Empirical evaluation of inside testing:** In the study, the 10 language models created by different size of corpus are evaluated sequentially for inside testing on these 10 models. As presented in Table 1, the x-axis and y-axis present the training model (TrM) and testing models (TeM), respectively. For each row in Table 2, testing models are used for evaluating 10 training models $TrM$. On the other side, 10 testing models $TeM$ will be used, respectively to evaluate one of 10 training models for WB-C smoothing. Figure 3 present the results on 3 dimensions respect to Table 1.

WB-C smoothing, testing models, shown in Table 1, are used for evaluating 10 training models $TrM$. Figure 3 presents the results of perplexity $PP$ of WB-C.
Table 1: Perplexity for WB-C smoothing method

<table>
<thead>
<tr>
<th>TrM/TeM</th>
<th>30M</th>
<th>60M</th>
<th>90M</th>
<th>120M</th>
<th>150M</th>
<th>180M</th>
<th>210M</th>
<th>240M</th>
<th>270M</th>
<th>300M</th>
</tr>
</thead>
<tbody>
<tr>
<td>test 30M</td>
<td>40.46</td>
<td>41.27</td>
<td>41.99</td>
<td>42.56</td>
<td>43.08</td>
<td>43.57</td>
<td>44.01</td>
<td>44.42</td>
<td>44.88</td>
<td>45.29</td>
</tr>
<tr>
<td>test 60M</td>
<td>42.27</td>
<td>40.88</td>
<td>41.31</td>
<td>41.76</td>
<td>42.20</td>
<td>42.64</td>
<td>43.03</td>
<td>43.42</td>
<td>43.84</td>
<td>44.22</td>
</tr>
<tr>
<td>test 90M</td>
<td>44.94</td>
<td>43.07</td>
<td>42.36</td>
<td>42.63</td>
<td>42.99</td>
<td>43.35</td>
<td>43.69</td>
<td>44.04</td>
<td>44.42</td>
<td>44.78</td>
</tr>
<tr>
<td>test 120M</td>
<td>47.07</td>
<td>44.99</td>
<td>43.95</td>
<td>43.57</td>
<td>43.78</td>
<td>44.07</td>
<td>44.35</td>
<td>44.66</td>
<td>45.01</td>
<td>45.34</td>
</tr>
<tr>
<td>test 150M</td>
<td>49.12</td>
<td>46.93</td>
<td>45.68</td>
<td>45.06</td>
<td>44.78</td>
<td>44.95</td>
<td>45.19</td>
<td>45.46</td>
<td>45.77</td>
<td>46.08</td>
</tr>
<tr>
<td>test 180M</td>
<td>51.07</td>
<td>48.78</td>
<td>47.38</td>
<td>46.62</td>
<td>46.15</td>
<td>45.94</td>
<td>46.09</td>
<td>46.31</td>
<td>46.59</td>
<td>46.87</td>
</tr>
<tr>
<td>test 210M</td>
<td>52.70</td>
<td>50.34</td>
<td>48.84</td>
<td>47.98</td>
<td>47.42</td>
<td>47.05</td>
<td>46.88</td>
<td>47.02</td>
<td>47.25</td>
<td>47.49</td>
</tr>
<tr>
<td>test 240M</td>
<td>54.28</td>
<td>51.87</td>
<td>50.27</td>
<td>49.33</td>
<td>48.69</td>
<td>48.22</td>
<td>47.88</td>
<td>47.72</td>
<td>47.87</td>
<td>48.07</td>
</tr>
<tr>
<td>test 270M</td>
<td>56.10</td>
<td>53.63</td>
<td>51.92</td>
<td>50.89</td>
<td>50.17</td>
<td>49.69</td>
<td>49.14</td>
<td>48.83</td>
<td>48.63</td>
<td>48.76</td>
</tr>
<tr>
<td>test 300M</td>
<td>57.85</td>
<td>55.32</td>
<td>53.52</td>
<td>52.42</td>
<td>51.63</td>
<td>50.99</td>
<td>50.42</td>
<td>50.00</td>
<td>49.66</td>
<td>49.51</td>
</tr>
<tr>
<td>Avg.</td>
<td>49.59</td>
<td>47.71</td>
<td>46.72</td>
<td>46.28</td>
<td>46.05</td>
<td>46.04</td>
<td>46.07</td>
<td>46.19</td>
<td>46.39</td>
<td>46.64</td>
</tr>
</tbody>
</table>

Table 2: Perplexity for YH-B smoothing method

<table>
<thead>
<tr>
<th>TrM/TeM</th>
<th>30M</th>
<th>60M</th>
<th>90M</th>
<th>120M</th>
<th>150M</th>
<th>180M</th>
<th>210M</th>
<th>240M</th>
<th>270M</th>
<th>300M</th>
</tr>
</thead>
<tbody>
<tr>
<td>test 30M</td>
<td>41.02</td>
<td>41.42</td>
<td>41.99</td>
<td>42.50</td>
<td>42.59</td>
<td>43.44</td>
<td>43.85</td>
<td>44.25</td>
<td>44.70</td>
<td>45.69</td>
</tr>
<tr>
<td>test 60M</td>
<td>42.64</td>
<td>41.02</td>
<td>41.32</td>
<td>41.69</td>
<td>42.10</td>
<td>42.51</td>
<td>42.89</td>
<td>43.25</td>
<td>43.66</td>
<td>44.03</td>
</tr>
<tr>
<td>test 90M</td>
<td>45.20</td>
<td>43.17</td>
<td>42.86</td>
<td>42.57</td>
<td>42.98</td>
<td>43.22</td>
<td>43.54</td>
<td>43.87</td>
<td>44.24</td>
<td>44.59</td>
</tr>
<tr>
<td>test 120M</td>
<td>47.25</td>
<td>45.05</td>
<td>43.95</td>
<td>43.50</td>
<td>43.68</td>
<td>43.93</td>
<td>44.20</td>
<td>44.49</td>
<td>44.83</td>
<td>45.15</td>
</tr>
<tr>
<td>test 150M</td>
<td>49.23</td>
<td>46.96</td>
<td>45.68</td>
<td>45.92</td>
<td>46.47</td>
<td>46.82</td>
<td>46.50</td>
<td>45.28</td>
<td>45.58</td>
<td>45.88</td>
</tr>
<tr>
<td>test 180M</td>
<td>51.12</td>
<td>48.78</td>
<td>47.38</td>
<td>46.59</td>
<td>46.08</td>
<td>45.90</td>
<td>45.93</td>
<td>46.03</td>
<td>46.40</td>
<td>46.67</td>
</tr>
<tr>
<td>test 210M</td>
<td>52.70</td>
<td>50.33</td>
<td>48.84</td>
<td>47.96</td>
<td>47.37</td>
<td>46.95</td>
<td>46.72</td>
<td>46.84</td>
<td>47.05</td>
<td>47.29</td>
</tr>
<tr>
<td>test 240M</td>
<td>54.21</td>
<td>51.83</td>
<td>50.27</td>
<td>49.33</td>
<td>48.68</td>
<td>48.17</td>
<td>47.77</td>
<td>47.54</td>
<td>47.68</td>
<td>47.86</td>
</tr>
<tr>
<td>test 270M</td>
<td>55.96</td>
<td>53.55</td>
<td>51.92</td>
<td>50.91</td>
<td>50.20</td>
<td>49.61</td>
<td>49.10</td>
<td>48.72</td>
<td>48.43</td>
<td>48.55</td>
</tr>
<tr>
<td>test 300M</td>
<td>57.62</td>
<td>55.21</td>
<td>53.52</td>
<td>52.47</td>
<td>51.70</td>
<td>51.04</td>
<td>50.45</td>
<td>49.98</td>
<td>49.54</td>
<td>49.30</td>
</tr>
<tr>
<td>Avg.</td>
<td>49.70</td>
<td>47.73</td>
<td>46.72</td>
<td>46.25</td>
<td>46.05</td>
<td>45.95</td>
<td>45.95</td>
<td>46.04</td>
<td>46.21</td>
<td>46.44</td>
</tr>
</tbody>
</table>

Fig. 3: Results of perplexity PP of WB-C smoothing

The smaller size of testing models, the lower of perplexity and the larger size of training models, the higher of perplexity. The lowest PP for each row is on the diagonal line in the table above.

In Fig. 3, the results of perplexity PP of WB-C smoothing are presented. Two observations are as following:

- The lowest PP was achieved on TrM_{i10M} for average PP of lower triangle of each training models.
- The lowest PP was also achieved on the model TrM_{i10M}, for average PP of each training models. The perplexity for TrM with size larger than 180M words will be also gradually increased.

In the study, the third one is our proposed smoothing method YH-B smoothing. As displayed in Table 2 and Fig. 4, the lowest PP for YH-B experiments on model TrM_{i10M} and TrM_{i10M} are same, 41.03. It is obvious the trend is also same as that of two other smoothing methods above.

Based on the evaluation results of PP for two smoothing methods, we could conclude that, in general case for average perplexity, the lowest PP can be achieved on model TrM_{i10M}. The experiment results could prove that the model which was created on larger than 180M corpus can't achieve better performance. On the other hand, our experiments supported that the model with middle size of corpus of 180M Chinese words can always achieve the best performance of language model.

We furthermore consider the PP differences for two smoothing methods. As shown in Fig. 5, the trend of PP for these methods is almost same. Totally, the YH-B smoothing perform well a bit than others WB-C methods for all training models. Note that the perplexities of YH-A and WB-A are all higher than WB-C and YH-B and results therefore do not be displayed in the study.
A larger size of corpus will alleviate the issue of data sparseness. In general, the conclusion matches the statistical features:

- YH-B smoothing is superior to WB-C smoothing methods for the TrM models with size between 30M and 90M only and degrades on larger models. We conclude that YH-B will perform well for smaller size of models in which the unknown events will occur frequently.

**CONCLUSION**

In the study, we evaluated several smoothing methods for statistical language models. These models are created on various size of corpus, between 30M and 300M Chinese words of CGW. Several smoothing methods, Winter-Bell A and C and two our proposed YH-A and YH-B smoothing, are all evaluated. Our YH-B smoothing is superior to other smoothing methods for the TrM models with size between 30M and 90M. Based on several observations, we analyzed furthermore the empirical results which is helpful for employing the effective smoothing methods to alleviate the issue of data sparseness of various size of training corpus.

**REFERENCES**


