On Sampling-Based Training Criteria for Neural Language Modeling

Yingbo Gao¹,², David Thulke¹,², Alexander Gerstenberger¹, Khoa Viet Tran¹, Ralf Schlüter¹,², Hermann Ney¹,²

¹Human Language Technology and Pattern Recognition Group, Computer Science Department
²AppTek GmbH, 52062 Aachen, Germany

Abstract

As the vocabulary size of modern word-based language models becomes ever larger, many sampling-based training criteria are proposed and investigated. The essence of these sampling methods is that the softmax-related traversal over the entire vocabulary can be simplified, giving speedups compared to the baseline. A problem we notice about the current landscape of such sampling methods is the lack of a systematic comparison and some myths about preferring one over another. In this work, we consider Monte Carlo sampling, importance sampling, a novel method we call compensated partial summation, and noise contrastive estimation. Linking back to the three traditional criteria, namely mean squared error, binary cross-entropy, and cross-entropy, we derive the theoretical solutions to the training problems. Contrary to some common belief, we show that all these sampling methods can perform equally well, as long as we correct for the intended class posterior probabilities. Experimental results in language modeling and automatic speech recognition on Switchboard and LibriSpeech support our claim, with all sampling-based methods showing similar perplexities and word error rates while giving the expected speedups.

Index Terms: sampling, training criterion, NCE, LM, ASR

1. Introduction

Enjoying the benefit of large amounts of text-only training data, language models (LMs) remain an important part of the modern automatic speech recognition (ASR) pipeline [1, 2, 3]. However, the large quantity of available data is a double-edged sword, posing real challenges in training. For example, the popular BERT model [4] and the recent GPT-2 and GPT-3 models [5, 6] have millions of parameters and are trained on billions of tokens. For BERT and GPT, both systems use byte pair encoding [7] to mitigate the problem of potentially very large vocabularies. However, for ASR, it is not uncommon for LMs to operate on the word level with a vocabulary size in the order of several hundred thousands [8, 9].

In order to train the neural LMs more efficiently, many speedup methods are proposed. To name a few: hierarchical softmax is used in [10], which changes the flat classification over all words to a series of binary decisions to arrive at the correct word; negative sampling (NS) method in [11] sums over a few sampled words instead of the full vocabulary; the noise contrastive estimation (NCE) method is proposed in [12] and adapted later in [13] for efficient estimation of LMs. Note that this is in no way an exhaustive enumeration of the ideas and methods, because there exist other works that introduce interesting concepts to address the problem, e.g. the Monte Carlo (MC) sampling and importance sampling (IS) discussed in [14]. For an overview of approximations to softmax, we refer the readers to a comprehensive blog post by Sebastian Ruder [15].

Among these methods, we find an interesting appreciation for NCE. In [16], the authors discuss the self normalization properties of models trained with NCE. In another line of work that maximizes mutual information for representation learning [17, 18, 19, 20], the NCE concept is frequently used. In a preprint note by Chris Dyer [21], a short conclusion is drawn “Thus, if your goal is language modeling, you should use NCE; if your goal is word representation learning, you should consider both NCE and negative sampling.” As a result, if one overlooks the math and takes it for granted, methods like NS may not sound very attractive for language modeling.

In this paper, we start from three fundamental criteria, namely mean squared error (MSE), binary cross-entropy (BCE), and cross-entropy (CE). We explicitly write out the sampling-based versions of them. Then, we derive the theoretical optimums of the model, and show that although for some sampling-based criteria we may not directly obtain the original class posterior probabilities from the model outputs, because there is a one-to-one mapping between the optimum and the posterior, we can correct the model outputs and obtain the desired probabilities. By doing so, the model also has self-normalization properties and gives reasonable performances. Note that we are not here to argue that any of the cited work is fundamentally wrong, our goal is simply to raise awareness that models trained with sampling-based criteria other than NCE can also perform well, given enough care.

Our contribution can be summarized into two points:

• First, we examine various sampling methods systematically, linking back to traditional criteria MSE, BCE and CE, and derive optimums for them.

• Second, we show from both a theoretical and a practical perspective, that all these sampling methods under consideration work, giving similar perplexities (PPLs), word error rates (WERs) and speedups.

2. Related Work

Neural LMs are commonly used in second-pass rescoring [1, 2, 22, 23] or first-pass decoding [3] in ASR systems. While for conventional research-oriented datasets like Switchboard the word-level vocabulary size is several dozens of thousands, for larger systems, especially commercially available systems, the vocabulary size can often go up to several hundred thousand. Numerous methods to speed up the training (and potentially testing) of word-level LMs are proposed in the past decades [10, 11, 12, 13, 14, 15]. These methods either exploit the statistical structure or perform sampling in the large target vocabulary. Among the sampling methods, NCE enjoys special appreciation [17, 18, 19, 20, 21], whose self-normalization property
is also examined [16]. For self-normalization and variance regularization, it is shown that explicit additional losses can also be added to the cross-entropy training criterion [24]. Traditionally, MSE, BCE, and CE are three training criteria that give the correct class posterior probabilities [25, 26]. In this paper, we make a connection between sampling-based criteria and these traditional ones and show that as long as one corrects for the intended class posterior probabilities, these sampling methods all show similar PPLs and WERs, while giving the expected speedups.

3. Sampling-Based Training Criteria

In this section, we formally define the training criteria. To clarify the notations, in the following sections, \( n \) is a running index in the number of word positions \( N, c, c' \) and \( \tilde{c} \) are running indices in the target vocabulary \( C \), which is supposedly very large. The context or history for next word prediction is denoted with \( x \), and the Kronecker delta \( \delta \) is used to decide the identity of model prediction and ground truth. We use \( \theta \) for model parameters, \( q \) for model outputs, \( p \) for class posterior probabilities, and \( \tilde{q} \) to represent the derived optimum. When model outputs are explicitly normalized, \( q(c|x) \) is used, otherwise, \( q(x,c) \) is used. \( F \) represents the training criterion to maximize. Lastly, the distribution from which we sample is denoted with \( D \), and \( k \) is a running index in the number of samples \( K \). Due to the limited pages, we only show the criterion definition and the optimum, and comment on the important steps in derivations in this paper.

3.1. Traditional Criteria

3.1.1. Mean Squared Error (MSE)

MSE is a classic training criterion commonly used for regression problems. Intuitively, it corresponds to “counting the errors”, but in the continuous sense.

\[
F_{\text{MSE}}(\theta) := -\frac{1}{N} \sum_{n=1}^{N} \sum_{c} \left( q_n(x_n,c) - \delta(c_n,c) \right)^2 \tag{1}
\]

\[
\Rightarrow \tilde{q}_n(x,c) = p(c|x) \tag{2}
\]

To obtain the optimum, one could rewrite the summation in \( N \) into a summation in \( x \), expand the squared term, and single out the terms related to \( q_n \). By definition, \( q \) can be unbounded, but it is common to parametrize \( q \) to be positive. In our preliminary experiments, further constraining \( q \) to be between zero and one with sigmoid slightly boosts the performance. However, although we spend a great amount of effort to tune the MSE-based models\(^1\), the PPLs are still much worse than BCE and CE, therefore we only describe the MSE criterion here for the sake of completeness and not mention it later.

3.1.2. Binary Cross Entropy (BCE)

BCE is another traditional training criterion. The motivation of BCE can be summarized as to “encourage the correct predictions and discourage the wrong predictions”.

\[
F_{\text{BCE}}(\theta) := \frac{1}{N} \sum_{n=1}^{N} \left( \log q_n(x_n,c_n) + \sum_{c \neq c_n} \log(1 - q_n(x_n,c)) \right) \tag{3}
\]

\[
\Rightarrow \tilde{q}_n(x,c) = p(c|x) \tag{4}
\]

Here, divergence inequality can be used for derivation. Note that \( q \) is required to be bounded in \((0, 1)\) and it is commonly done via a sigmoid operation.

3.1.3. Cross Entropy (CE)

CE is arguably the most commonly used criterion nowadays and finds its roots in information theory and probabilistic theory. Intuitively, CE “encourages the model probability on the target word to be more exactly correct”.

\[
F_{\text{CE}}(\theta) := \frac{1}{N} \sum_{n=1}^{N} \log \left( \frac{\exp \tilde{q}_n(x_n,c_n)}{\sum_{c \neq c_n} \exp \tilde{q}_n(x_n,c')} \right) \tag{5}
\]

\[
\Rightarrow \frac{\exp \tilde{q}_n(x,c)}{\sum_{c \neq c_n} \exp \tilde{q}_n(x,c')} = p(c|x) \tag{6}
\]

Again, applying divergence inequality here would give us the optimum. We explicitly write out the softmax operation in this case to highlight the summation in the big vocabulary \( C \) in the denominator. In this case, \( q \) is unbounded and the softmax guarantees the normalized property.

3.2. Sampling-Based Criteria

According to the previous discussion about MSE, BCE, and CE, we see a summation in \( C \) in all three cases. This summation can be viewed as an expectation of some quantity \( Q \), under the uniform distribution \( \frac{1}{C} \): \( \sum_{c} Q_c = C \sum_{c} \frac{1}{C} Q_c = CE(Q_c) \). Approximating this expectation is the core concept of the sampling methods.

3.2.1. Monte Carlo Sampling (MCS)

MCS approximates an expectation by some sample mean. Specifically, instead of summing over \( C \), we sum over \( K \) random samples where \( \tilde{c}_k \sim D \). NS in [10] is a prominent example of MCS.

\[
F_{\text{BCE-MCS}}(\theta) := \frac{1}{N} \sum_{n=1}^{N} \left( \log q_n(x_n,c_n) + \sum_{k=1}^{K} \log(1 - q_n(x_n,\tilde{c}_k)) \right) \tag{7}
\]

\[
\Rightarrow \tilde{q}_n(x,c) \approx \left( 1 + \frac{K \mathcal{D}(c)}{p(c|x)} \right)^{-1} \tag{8}
\]

3.2.2. Importance Sampling (IS)

IS rewrites the expectation by introducing another distribution other than the uniform distribution. In our case, this new distri-
The term $KD(c)$ often shows up in our derivation due to rewriting the summation in $K$ into a summation in $C$: $\sum_n \sum_k Q_{n,k} \approx \sum_n \sum_k K D(c) Q_{n,c}$. Approximately, the number of terms that show up in the two summations should be equal, when given $n$ and $K$ is large enough. This trick is used in many of the derivations. Note that the model outputs $q$ have different constraints in different criteria, the logits we plug into the criteria are activated according, e.g. applying sigmoid if $q$ is bounded between zero and one.

From the derived optimima, if the model output $q(x, c)$ is not strictly the intended class posterior probability $p(c|x)$ (unlike in the case of three traditional criteria and BCE-NCE), we confirm the statement in [21] that such models are not directly applicable for language modeling. However, for all cases, it is clear that there is a one-to-one mapping between $q$ and $p$. Therefore, given each model output $q(x, c)$, we can calculate the desired $p(c|x)$ (optionally querying the noise distribution $D$), and use this quantity for rescoring in ASR.

## 4. Experiments

### 4.1. Experimental Setup

For the experimental validation, we train word-based LMs on LibriSpeech and SwitchBoard and evaluate the resulting models in well-tuned ASR systems with second-pass lattice rescoring. The vocabulary contains about 200k words for LibriSpeech and about 30k words for SwitchBoard, respectively.

For all training criteria, we make use of the same model architecture. For LibriSpeech, our LMs make use of the Transformer [28] architecture, which is motivated by recent state-of-the-art results outperforming LSTM-based LMs on this task by a large margin [9, 29]. We use 42 layers, 512 input embedding dimension, 2048 feed-forward dimension, 8 attention heads, and 512 residual and key/query/value dimension. We do not use a positional encoding, following the architecture presented in [29]. For SwitchBoard, we use LSTM LMs with 2 layers and 2048 hidden units, and 128 embedding dimension.

For sampling-based approaches, we sample 8192 samples from log-uniform noise distributions, which performs well in our preliminary experiments. The noise samples are shared over the whole batch for computational efficiency in all cases [30, 31]. The parameters are learned using stochastic gradient descent with a learning rate of 1. The global gradient norms are clipped to 1. We implement our models using the open-source toolkit RETURNN [32], based on TensorFlow [33].

The acoustic models are based on hybrid hidden Markov models. For detailed information on training setups, including discriminative training approaches, adaptation, and model architectures we refer the reader to [34] for the SwitchBoard and [9] for the LibriSpeech systems. For LibriSpeech the lattices are generated using a well-tuned LSTM LM in the first pass [3]. For SwitchBoard, the lattices are generated using a 4-grain Kneser-Ney LM [34]. We interpolate the LSTM LMs with count-based LMs for evaluation. We report on clean and other test sets for LibriSpeech and on SwitchBoard (SWB) and CallHome (CH) Hub5’00 test sets for SwitchBoard. During LM training we use a separate validation dataset. PPLs and WERs are always obtained with proper normalization over the full vocabulary unless otherwise stated.

---

2 Ideally, one should tune the hyperparameters for each training criterion individually. Given the computational and time constraints, we believe that this is still a reasonable approach.

3 Example config is available at https://git.io/Jm9lQ.
noted otherwise. The reported average training time per batch is obtained on NVIDIA GeForce GTX 1080 Ti GPUs, with a batch size of 64 for LibriSpeech and 32 for Switchboard respectively.

4.2. Main Results
Below we present the main results. Table 1 gives the PPLs and WERs of our baseline models [3, 34]. In our opinion, these are competitive systems and serve as reasonable baselines for our purpose of judging different sampling methods.

Table 1: Baseline PPLs and WERs.

| dataset      | model | PPL  | WER |
|--------------|-------|------|-----|
| LibriSpeech  | LSTM  | 64.3 | -   |
|              | clean | 2.6  | 5.8 |
| SwitchBoard  | 4-gram| 74.6 | 8.1 |

In Table 2, we present the PPLs and WERs of the sampling-based models on the LibriSpeech dataset. Since we sample 8k samples out of a 200k vocabulary, for all sampling methods under consideration, there is a significant relative training time speedup of over 40%, which is expected. Because of limited computational resources and our purpose not being to obtain the best trade-off between speed and quality, we do not sweep over different sampling sizes. Note that we do not include CE-NCE results due to some numerical problems. In terms of PPLs and WERs, we see a consistent improvement over the baseline, and we attribute this to using the Transformer and not the LSTM architecture [9, 29]. Comparing different sampling methods, we see a small variation in PPLs and even less in WERs. This justifies our statement that all these sampling methods work equally well when the model outputs are corrected accordingly.

Table 2: Sampling-Based Transformer LMs on LibriSpeech.

| criterion | sampling | train time (ms/batch) | PPL  | WER |
|-----------|----------|-----------------------|------|-----|
| BCE       | -        | 0.138                 | 58.5 | 2.5 |
|           | MCS      | 0.213                 | 58.0 | 2.6 |
|           | IS       | 0.206                 | 58.4 | 2.6 |
|           | CPS      | 0.205                 | 58.4 | 2.5 |
|           | NCE      | 0.206                 | 57.9 | 2.5 |
| CE        | -        | 0.302                 | 57.7 | 2.5 |
|           | MCS      | 0.206                 | 57.9 | 2.5 |
|           | IS       | 0.201                 | 57.8 | 2.5 |
|           | CPS      | 0.203                 | 62.2 | 2.5 |

In Table 3, similar results are presented for the Switchboard dataset. In this case, all sampling-based methods give consistent relative training time speedup of over 20%. Considering that here 8k samples are drawn out of a 30k vocabulary, instead of 8k being drawn from a 200k vocabulary like that for LibriSpeech, the relative speedup being smaller is thus expected. PPL-wise, significant improvements over count-based baseline LM are achieved, which is also confirmed numerous times in different works on different datasets. The CE baseline is slightly better than the sampling-based LMs because our training recipe is well-tuned for the CE baseline. Comparing different sampling-based criteria, NCE and IS are slightly better, but the advantage is not large enough to justify one method being strictly superior than another. In terms of WERs, all sampling methods perform similarly.

Table 3: Sampling-Based LSTM LMs on Switchboard.

| criterion | sampling | train time (ms/batch) | PPL  | WER |
|-----------|----------|-----------------------|------|-----|
| BCE       | -        | 0.107                 | 52.3 | 10.3 |
|           | MCS      | 0.077                 | 52.6 | 10.3 |
|           | IS       | 0.079                 | 51.5 | 10.3 |
|           | CPS      | 0.079                 | 52.4 | 10.3 |
|           | NCE      | 0.078                 | 51.4 | 10.3 |
| CE        | -        | 0.100                 | 49.9 | 10.1 |
|           | MCS      | 0.078                 | 52.4 | 10.4 |
|           | IS       | 0.076                 | 52.3 | 10.2 |
|           | CPS      | 0.077                 | 52.3 | 10.3 |

While PPLs and WERs reported above are properly normalized and give expected training time speedups, what makes these sampling methods especially attractive is the use case without explicit normalization. To this end, we look at the BCE sampling variants and directly rescore with the model outputs of these LMs, and report the WERs on Switchboard. As seen in Table 4, BCE-IS performs on par with BCE-NCE, as well as the CE baseline shown in Table 3, which shows its competitiveness and self-normalization properties. We notice that when plugging in the log-uniform distribution for D, the BCE-MCS and BCE-CPS performance without normalization can get much worse to around 11.8% WER, which is similar to the count-based model. We attribute this to having to query the unreliable noise distribution D during search (which is not the case for BCE-IS and BCE-NCE) and therefore use a smoothed empirical unigram distribution for D instead.

Table 4: WERs without Explicit Normalization on SwitchBoard.

| criterion | sampling | WER |
|-----------|----------|-----|
| BCE-MCS   | -        | 11.0 |
| BCE-IS    | -        | 10.2 |
| BCE-CPS   | -        | 11.3 |
| BCE-NCE   | -        | 10.2 |

5. Conclusion
For language modeling with large vocabularies, we consider different sampling-based training criteria. We start from three traditional criteria and formulate sampling-based versions of them. We derive optimus and argue that when model outputs are corrected for the intended class posteriors, these methods perform equally well compared to the popular noise contrastive estimation. Experimental evidence of perplexity and word error rate results on LibriSpeech and SwitchBoard support our claim. For direct rescoring without explicit normalization, we show the self-normalization properties of such sampling-based models.

6. Acknowledgements
This project has received funding from the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme (grant agreement n° 694537, project “SEQCLAS”). The work reflects only the authors’ views and the European Research Council Executive Agency (ERC EA) is not responsible for any use that may be made of the information it contains. Simulations were performed with computing resources granted by RWTH Aachen University under project rwth0582. We thank Christoph Lüscher for providing the LibriSpeech acoustic model and Eugen Beck for the lattices, and Markus Kitza for the SwitchBoard lattices.
