Abstract

We propose an unsupervised importance sampling approach to selecting training data for recurrent neural network (RNN) language models. To increase the information content of the training set, our approach preferentially samples high perplexity sentences, as determined by an easily queryable $n$-gram language model. We experimentally evaluate the heldout perplexity of models trained with our various importance sampling distributions. We show that language models trained on data sampled using our proposed approach outperform models trained over randomly sampled subsets of both the Billion Word (Chelba et al., 2014) and Wikitext-103 benchmark corpora (Merity et al., 2016).

1 Introduction

The task of statistical language modeling seeks to learn a joint probability distribution over sequences of natural language words. In recent work, recurrent neural network (RNN) language models (Mikolov et al., 2010) have produced state-of-the-art perplexities in sentence-level language modeling, far below those of traditional $n$-gram models (Melis et al., 2017). Models trained on large, diverse benchmark corpora such as the Billion Word Corpus and Wikitext-103 have seen reported perplexities as low as 23.7 and 37.2, respectively (Kuchaiev and Ginsburg, 2017; Dauphin et al., 2017).

However, building models on large corpora is limited by prohibitive computational costs, as the number of training steps scales linearly with the number of tokens in the training corpus. Sentence-level language models for these large corpora can be learned by training on a set of sentences subsampled from the original corpus. We seek to determine whether it is possible to select a set of training sentences that is significantly more informative than a randomly drawn training set. We hypothesize that by training on higher information and more difficult training sentences, RNN language models can learn the language distribution more accurately and produce lower perplexities than models trained on similar-sized randomly sampled training sets.

We propose an unsupervised importance sampling technique for selecting training data for sentence-level RNN language models. We leverage $n$-gram language models’ rapid training and query time, which often requires just a single pass over the training data. We determine a preliminary heuristic for each sentence’s importance and information content by calculating its average word perplexity. Our technique uses an offline $n$-gram model to score sentences and then samples higher perplexity sentences with increased probability. Selected sentences are then used for training with corrective weights to remove the sampling bias. As entropy and perplexity have a monotonic relationship, selecting sentences with higher average $n$-gram perplexity also increases the average entropy and information content.

We experimentally evaluate the effectiveness of multiple importance sampling distributions at selecting training data for RNN language models. We compare the heldout perplexities of models trained with randomly sampled and importance sampled training data on both the One Billion Word and Wikitext-103 corpora. We show that our importance sampling techniques yield lower perplexities than models trained on similarly sized random samples. By using an $n$-gram model to determine the sampling distribution, we limit added computational costs of our importance sampling approach. We also find that applying perplexity-based importance sampling requires maintaining a relatively high weight on low perplexity sentences. We hypothesize that this is because low
perplexity sentences frequently contain common subsequences that are useful in modeling other sentences.

2 Related Work

Standard stochastic gradient descent (SGD) iteratively selects random examples from the training set to perform gradient updates. In contrast, weighted SGD has been proven to accelerate the convergence rates of SGD by leveraging importance sampling as a means of variance reduction (Needell et al., 2014; Zhao and Zhang, 2015). Weighted SGD selects examples from an importance sampling distribution and then trains on the selected examples with corrective weights. Weights of each training example $i$ are set to be $\frac{1}{Pr(i)}$, where $Pr(i)$ is the probability of selecting example $i$. The weighting provides an unbiased estimator of overall loss by removing the bias of the importance sampling distribution. In expectation, each example’s contribution to the total loss function is the same as if the example had been drawn uniformly at random.

Alain et al. (2015) developed an importance sampling technique for training deep neural networks by sampling examples directly according to their gradient norm. To avoid the high computational costs of gradient computations, Katharopolous and Fleuret (2018) sample according to losses as approximated by a lightweight RNN model trained along side their larger primary RNN model. Both techniques observed increased convergence rates and reduced errors in image classification tasks. In comparison, we use a fixed offline $n$-gram model to compute our sampling distribution, which can be trained and queried much more efficiently than a neural network model.

In natural language processing, subsampling of large corpora has been used to speed up training for both language modeling and machine translation. For domain specific language modeling, Moore and Lewis (2010) used an $n$-gram model trained on in-domain data to score sentences and then selected the sentences with low perplexities for training. Both Cho et al. (2014) and Koehn and Haddow (2012) used similar perplexity-based sampling to select training data for domain specific machine translation systems. Importance sampling has also been used to increase rate of convergence for a class of neural network language models which use a set of binary classifiers to determine sequence likelihood, rather than calculating the probabilities jointly (Xu et al., 2011).

Because these subsampling techniques are used to learn domain specific distributions different from the distribution of the original corpus, they target lower perplexity sentences and have no need for corrective weighting. In contrast, we study how training sets generated using weighted importance sampling can be selected to maximize knowledge of the entire corpus for the standard language modeling task.

3 Methodology

First, we train an offline $n$-gram model over sentences randomly sampled from the training corpus. Using the $n$-gram model, we score perplexities for the remaining sentences in the training corpus.

We propose multiple importance sampling and likelihood weighting schemes for selecting training sequences for an RNN language model. Our proposed sampling distributions (discussed in detail below) bias the training set to select higher perplexity sentences in order to increase the training set’s information content. We then train an RNN language model on the sampled sentences with weights set to the reciprocal of the sentence’s selection probability.

3.1 Z-Score Sampling ($Z_{full}$)

This sampling distribution naively selects sentences according to their z-score, as calculated in terms of their $n$-gram perplexities. The selection probability of sequence $s$ is set as:

$$P_{Keep}(s) = k_{pr} \left( \frac{ppl(s) - \mu_{ppl}}{\sigma_{ppl}} + 1 \right),$$

where $ppl(s)$ is the $n$-gram perplexity of sentence $s$, $\mu_{ppl}$ is the average $n$-gram perplexity, $\sigma_{ppl}$ is the standard deviation of $n$-gram perplexities, and $k_{pr}$ is a normalizing constant to ensure a proper probability distribution.

For sentences with z-scores less than $-1.00$ or sequences where $ppl(s)$ was in the 99th percentile of $n$-gram perplexities, sequences are assigned $P_{keep}(s) = k_{pr}$. This ensured all sentences had positive selection probability and limited bias towards the selection of high perplexity sequences in the tail of the distribution. Upon examination, sequences with perplexities in the 99th percentile were generally esoteric or nonsensical. Selection
of these high perplexity sentences provided minimal accuracy gain in exchange for their boosted selection probability.

3.2 Limited Z-Score Sampling ($Z_\alpha$)

Training on low perplexity sentences can be helpful in learning to model higher perplexity sentences which share common sub-sequences. However, naive z-score sampling results in the selection of few low perplexity sentences. Additionally, the low perplexity sentences that are selected tend to dominate the training weight space due to their low selection probability.

To smooth the distribution in the weight space, selection probability is only determined using z-scores for sentences where their perplexities are greater than the mean. Thus, the selection probability of sentence $s$ is:

$$P_{\text{Keep}}(s) = \begin{cases} k_{pr} \left( \frac{\alpha \cdot \text{ppl}(s) - \mu_{\text{ppl}}}{\sigma_{\text{ppl}}} + 1 \right), & \text{if } \text{ppl}(s) > \mu_{\text{ppl}}, \\ k_{pr}, & \text{else}. \end{cases}$$

where $\alpha$ is a constant parameter that determines the weight of the z-score in calculating the sequence’s selection probability.

3.3 Squared Z-Score Sampling ($Z^2$)

To investigate the effects of sampling from more complex distributions, we also evaluate an importance sampling scheme where sentences are sampled according to their squared z-score.

$$P_{\text{Keep}}(s) = \begin{cases} k_{pr} \left( \alpha \frac{\text{ppl}(s) - \mu_{\text{ppl}}}{\sigma_{\text{ppl}}}^2 + 1 \right), & \text{if } \text{ppl}(s) > \mu_{\text{ppl}}, \\ k_{pr}, & \text{else}. \end{cases}$$

4 Experiments

We experimentally evaluate the effectiveness of the $Z_{\text{full}}$ and $Z^2$ sampling methods, as well as the $Z_\alpha$ method for various values of parameter $\alpha$.

4.1 Dataset Details

Sentence-level models were trained and evaluated on samples from Wikitext-103 and the One Billion Word Benchmark corpus. To create a dataset of independent sentences, the Wikitext-103 corpus was parsed to remove headers and to create individual sentences. The training and heldout sets were combined, shuffled, and then split to create new 250k token test and validation sets. The remaining sequences were set as a new training set of approximately 99 million tokens. In Billion Word experiments, training sequences were sampled from a 500 million subset of the released training split. Billion Word models were evaluated on 250k token test and validation sets randomly sampled from the released heldout split.

Models were trained on 500 thousand, 1 million, and 2 million token training sets sampled from each training split. Rare words were replaced with `<unk>` tokens, resulting in vocabularies of 267K and 250K for the Wikitext and Billion Word corpora, respectively.

4.2 Model Details

To calculate the sampling distribution, an $n$-gram model was trained on a heldout set with the same number tokens used to train each RNN model (Hochreiter and Schmidhuber, 1997). For example, the sampling distribution used to build a 1 million token RNN training set was determined using perplexities calculated by an $n$-gram model also trained on 1 million tokens. $N$-gram models were trained as 5-gram models with Kneser-Ney discounting (Kneser and Ney, 1995) using SRILM (Stolcke, 2002). For efficient calculation of sentence perplexities, we query our models using KenLM (Heafield, 2011).

RNN models were built using a two-layer long short-term memory (LSTM) neural network, with 200-dimensional hidden and embedding layers. Each training set was trained on for 10 epochs using the Adam gradient optimizer (Kingma and Ba, 2014) with a mini-batch size of 12.

5 Results

In Tables 1 and 2, we summarize the performances of models trained on samples from Wikitext-103 and the Billion Word Corpus, respectively. We report the Random and $n$-gram baseline perplexities for RNN and $n$-gram language models trained on randomly sampled data. We also report $\mu_{\text{ngram}}$ and $\sigma_{\text{ngram}}$ for each training set, which are the mean and standard deviation in sentence perplexity as evaluated by the offline $n$-gram model.

In all experiments, RNN language models trained using our sampling approaches yield a decrease in model perplexity as compared to RNN models trained on similar sized randomly sampled sets. As size of the training set increases, the RNNs trained on importance sampling datasets...
also yield significantly lower perplexities than the \( n \)-gram models trained on randomly sampled training sets. As expected, \( \mu_{\text{ngram}} \) and \( \sigma_{\text{ngram}} \) increase substantially for training sets generated using our proposed sampling methods.

Overall, the \( Z_{4.0} \) sampling method produced the most consistent reductions in average perplexity of 102.9 and 54.2 compared to the Random and \( n \)-gram baselines, respectively. \( Z_{\text{full}} \) and \( Z^2 \) exhibit higher variance in their heldout perplexity as compared to the \( Z_{\alpha} \) and baseline methods. We expect that this is because these methods select higher perplexity sequences with significantly higher probability than \( Z_{\alpha} \) methods. As a result, low perplexity sentences, which may contain common subsequences helpful in modeling other sentences, are ignored in training.

### 6 Conclusions and Future Work

We introduce a weighted importance sampling scheme for selecting RNN language model training data from large corpora. We demonstrate that models trained with data generated using this approach yield perplexity reductions of up to 24\% when compared to models trained over randomly sampled training sets of similar size. This technique leverages higher perplexity training sen-

\begin{table*}
\centering
\begin{tabular}{|c|c|c|c|c|c|}
\hline
Model & Tokens & \( \mu_{\text{ngram}} \) & \( \sigma_{\text{ngram}} \) & ppl & \\
\hline
\( n \)-gram & 300k & 449.0 & 346.4 & 492.3 & \\
Random & 500k & 497.1 & 398.8 & 749.1 & \\
\( Z_{0.5} \) & 500k & 544.1 & 440.1 & 643.9 & \\
\( Z_{1.0} \) & 500k & 615.6 & 481.3 & 645.2 & \\
\( Z_{2.0} \) & 500k & 729.0 & 523.6 & 593.2 & \\
\( Z_{4.0} \) & 500k & 576.5 & 499.7 & 720.0 & \\
\( Z^2 \) & 500k & 627.1 & 451.9 & 663.7 & \\
\hline
\( Z_{\text{full}} \) & 1M & — & — & 502.7 & \\
Random & 1M & 448.9 & 380.2 & 550.6 & \\
\( Z_{0.5} \) & 1M & 495.7 & 431.8 & 545.7 & \\
\( Z_{1.0} \) & 1M & 540.4 & 475.4 & 435.4 & \\
\( Z_{2.0} \) & 1M & 615.6 & 528.4 & 426.9 & \\
\( Z_{4.0} \) & 1M & 732.9 & 584.4 & 420.1 & \\
\( Z^2 \) & 1M & 571.5 & 535.7 & 435.7 & \\
\hline
\end{tabular}
\end{table*}

\begin{table*}
\centering
\begin{tabular}{|c|c|c|c|c|}
\hline
Model & Tokens & \( \mu_{\text{ngram}} \) & \( \sigma_{\text{ngram}} \) & ppl & \\
\hline
\( n \)-gram & 1M & 433.2 & 515.4 & 484.0 & \\
Random & 1M & 476.8 & 410.9 & 436.6 & \\
\( Z_{0.5} \) & 1M & 543.8 & 529.0 & 421.5 & \\
\( Z_{1.0} \) & 1M & 726.4 & 517.3 & 427.3 & \\
\( Z_{4.0} \) & 1M & 635.19 & 458.69 & 495.75 & \\
\( Z_{\text{full}} \) & 1M & 639.2 & 593.7 & 435.3 & \\
\hline
\end{tabular}
\end{table*}

The table above shows perplexities for Wikitext models. All proposed models outperform the random and \( n \)-gram baselines as number of training tokens increases.

\begin{table*}
\centering
\begin{tabular}{|c|c|c|c|c|}
\hline
Model & Tokens & \( \mu_{\text{ngram}} \) & \( \sigma_{\text{ngram}} \) & ppl & \\
\hline
\( n \)-gram & 2M & — & — & 502.6 & \\
Random & 2M & 400.45 & 392.1 & 341.3 & \\
\( Z_{0.5} \) & 2M & 471.8 & 445.2 & 292.7 & \\
\( Z_{1.0} \) & 2M & 514.6 & 493.9 & 289.8 & \\
\( Z_{2.0} \) & 2M & 582.8 & 544.6 & 346.9 & \\
\( Z_{4.0} \) & 2M & 684.6 & 604.7 & 294.6 & \\
\( Z^2 \) & 2M & 518.4 & 522.9 & 287.9 & \\
\hline
\end{tabular}
\end{table*}

\begin{table*}
\centering
\begin{tabular}{|c|c|c|c|}
\hline
Model & Tokens & \( \mu_{\text{ngram}} \) & \( \sigma_{\text{ngram}} \) & ppl & \\
\hline
\( n \)-gram & 2M & — & — & 312.5 & \\
Random & 2M & 568.4 & 506.5 & 312.5 & \\
\( Z_{\text{full}} \) & 2M & 568.4 & 506.5 & 312.5 & \\
\hline
\end{tabular}
\end{table*}

Table 1: Perplexities for Wikitext models. All proposed models outperform the random and \( n \)-gram baselines as number of training tokens increases.

Table 2: Perplexities for Billion Word models. \( Z_{\alpha} \) and \( Z^2 \) both outperform the random baseline and are comparable to the \( n \)-gram baseline.

sentences to learn more accurate language models, while limiting added computational cost of importance calculations.

In future work, we will examine the performance of our proposed selection techniques in additional parameter settings, with various values of \( \alpha \) and thresholds in the limited z-score methods \( Z_{\alpha} \). We will evaluate the performance of sampling distributions based on perplexities calculated using small, lightweight RNN language models rather than \( n \)-gram language models. Additionally, we will also be evaluating the performance of sampling distributions calculated based on a sentence’s subsequences and unique \( n \)-gram content. Furthermore, we plan on adapting this importance sampling approach to use online \( n \)-gram models trained alongside the RNN language models for determining the importance sampling distribution.

### Acknowledgements

This work was supported in part by NSF Grant IIS-1351029. Support for travel provided in part by the ACL Student Travel Grant (NSF Grant IIS-1827830) and the Conference Travel Grant from Northwestern University’s Office of the Provost. We thank Dave Demeter, Thanapon Noraset, Yiben Yang, and Zheng Yuan for helpful comments.

### References


