Experimenting with Power Divergences for Language Modeling

Matthieu Labeau  Shay B. Cohen
Institute for Language, Cognition and Computation
School of Informatics, University of Edinburgh
{mlabeau,scohen}@inf.ed.ac.uk

Abstract

Neural language models are usually trained using Maximum-Likelihood Estimation (MLE). The corresponding objective function for MLE is derived from the Kullback-Leibler (KL) divergence between the empirical probability distribution representing the data and the parametric probability distribution output by the model. However, the word frequency discrepancies in natural language make performance extremely uneven: while the perplexity is usually very low for frequent words, it is especially difficult to predict rare words. To address that, we experiment with several families (\(\alpha\), \(\beta\) and \(\gamma\)) of power divergences, generalized from the KL divergence, for learning language models with an objective different than standard MLE. Intuitively, these divergences should affect the way the probability mass is spread during learning, notably by prioritizing performance on high or low-frequency words. In addition, we implement and experiment with various sampling-based objectives, where the computation of the output layer is only done on a small subset of the vocabulary. They are derived as power generalizations of a softmax approximated via Importance Sampling, and Noise Contrastive Estimation, for accelerated learning. Our experiments on the Penn Treebank and Wikitext-2 show that these power divergences can indeed be used to prioritize learning on the frequent or rare words, and lead to general performance improvements in the case of sampling-based learning.

1 Introduction

Language models are an important component in many NLP tasks, where they provide prior knowledge on the language used. They are conditional models that aim to predict the next token in a sequence: they can be applied to basic units ranging from individual characters to full words, each approach coming with its own benefits and limitations (Merity et al., 2018a). Word-level language models have traditionally been based on \(n\)-gram counts, obtaining good performance with smoothing techniques (Kneser and Ney, 1995; Goodman, 2001). Recently, neural networks have shown strong results in language modeling (Bengio et al., 2001), especially recurrent neural networks (Mikolov et al., 2011b). As previous approaches, like maximum entropy models (Berger et al., 1996), neural language models are trained via Maximum Likelihood Estimation (MLE). Thus, their training cost grows linearly with the number of words in the vocabulary, often making it prohibitively slow. This motivated a large amount of research work, bringing a variety of solutions (Chen et al., 2016).

The large vocabulary sizes encountered in training corpora arguably stem from the fact that the frequency distribution of words in a corpus of natural language follows Zipf’s law (Powers, 1998). This also implies that the discrepancy between counts of high-frequency and low-frequency words increases with the size of the corpus, as well as the number of those low-frequency words. As a consequence, distributed word representations of low-frequency words are difficult to learn. Numerous approaches use decomposition of words with various sub-word units (Sennrich et al., 2016; Kim et al., 2016), but the same phenomenon exists for low-frequency subwords. In order to deal with this issue and to accelerate training, Grave et al. (2017a) implement a dependency between embedding sizes and word frequencies in the output layer, while Baevski and Auli (2019) apply it to the input layer, comparing the possible choices of which units to model. Using a different approach, Gong et al. (2018) attempt to learn word representations that are less affected by these large discrepancies in word frequencies, with an adver-
serial training method to force the model to make frequent and rare word embeddings hard to differentiate based on word frequency alone.

These improvements have been obtained by explicitly incorporating in the model different ways of treating words according to their frequency. However, learning is always made via (or approximating) Maximum Likelihood Estimation, which finds the distribution that maximizes entropy subject to the constraints given by training examples. In this work, we explore the possibility of affecting how words are learned depending on their frequency by using alternative loss functions. We specifically explore power divergences, obtained through various generalizations of the Kullback-Leibler divergence, which is traditionally used to obtain the MLE objective function. This is motivated by the intuition that a well-suited power factor may direct learning towards prioritizing high or low-frequency words, instead of learning uniformly.

In this paper, we derive and experiment with the objective functions obtained from three power-divergences: the $\alpha$, $\beta$ and $\gamma$ divergences. We also derive objective functions for the corresponding generalizations of two sampling-based methods: an Approximated Softmax obtained with importance sampling, and Noise Contrastive Estimation. We conduct a set of experiments comparing these objectives and their effect of various parts of the word frequency distribution, by training and evaluating models on two corpora: the Penn Treebank and Wikitext-2. Our experiments show that depending on the vocabulary used and the choice of power divergence, it is indeed possible to gear learning to focus on the most frequent or infrequent words. We also observe that, while the MLE gives the best overall performance for exact objectives, derived from the KL-divergence, our generalized objectives yield perplexity improvements compared to baselines for both sampling-based methods, up to 1 point in perplexity on both corpora.

2 Background

Language modeling aims to learn a probability distribution over a sequence of tokens from a finite target vocabulary $Y$. Such a distribution is decomposed into a product of conditional distributions of tokens over $Y$ given the previous tokens in the sequence. Hence, we learn a parametric model of the form $p_\theta(y|x)$, where $x \in X$ represents the sequence of previous tokens, $y$ is a target label belonging to $Y$, and $\theta$ is the set of model parameters. They are obtained via maximum likelihood estimation (MLE), which consists in minimizing the negative likelihood objective function:

$$\text{NLL}(\theta) = - \sum_{(x,y) \in D} \log p_\theta(y|x)$$

over examples $(x, y)$ corresponding to sequences of tokens drawn from the data $D$. This can be seen as minimizing the Kullback-Leibler divergence between the parametrized probability distribution $p_\theta$ that we are learning and the distribution $p_D$ described by our training data:

$$D_{KL}(p_D||p_\theta) = \sum_{(x,y) \in X \times Y} p_D(y|x) \log \frac{p_D(y|x)}{p_\theta(y|x)},$$

since $p_D(y|x) = 1$ if the sequence $(x, y)$ appears in the training data, and equals 0 otherwise. Hence, the set of parameters $\theta^*$ minimizing $D_{KL}(p_D||p_\theta)$ is the maximum likelihood distribution on the training data $D$.

In this work, we will use several classes of divergences. A measure of discrepancy $D$ between two probability densities $p$ and $q$ is a divergence if $D(p||q) \geq 0$ with equality if and only if $p = q$. In the following, we will derive an objective function from a divergence $D$ with the data distribution as first argument, and the second being the distribution of the parametric model:

$$\text{Obj}(\theta) = D(p_D||p_\theta).$$

3 Power Divergences

A large number of divergence measures has been introduced for a variety of applications (Basseville, 2013). Several families of divergences are notably obtained from generalizing the Kullback-Leibler divergence by using a generalized logarithm function, which is a power function:

$$\log_\alpha(x) = \frac{1}{1-\alpha}(x^{1-\alpha} - 1),$$

defined by a parameter $\alpha$, and that converges to the logarithm as $\alpha \to 1$. In this section, we will consider three families of divergences that can be generated from this function; see Cichocki and Amari

---

1Divergences are not necessarily metrics: they are defined only by non-negativity and positive definiteness.
Tsallis entropies showed how to build Fenchel-Young Losses from the same Cichocki and Amari (2010) instead of Basu et al. (1998). Recently, Blondel et al. (2018) using this same generalized logarithm: see Appendix A from which can be seen as a deformation of the Shannon entropy is shown in Table 1. It is a special case of $f$-divergences

The notion of $\alpha$-divergence was introduced by Csiszar (1967). The full expression of $D_\alpha(p||q)$ is shown in Table 1. It is a special case of $f$-divergence (Ali and Silvey, 1966), derived from a standardized version of the generalized logarithm\(^2\) (Eq. 1). Applying $\alpha$-divergences to parameter estimation generalizes MLE, and can be shown to have similar properties, as consistency and asymptotic normality of the estimation error (Keziou, 2003). Intuitively, the choice of $\alpha$ will impact the importance of the likelihood ratio $p/q$: while the limiting cases $\alpha \to 1$ and $\alpha \to 0$ are the Kullback-Leibler $D_{KL}(p||q)$ and the reverse Kullback-Leibler divergences $D_{KL}(q||p)$, having $\alpha \geq 1$ will make learning zero-avoiding\(^3\) for $q$, and $\alpha \leq 0$ zero-forcing\(^4\) (Minka, 2005). We should also note that $D_\alpha(p||q) = D_{1-\alpha}(q||p)$. When working with normalized densities, the $\alpha$-divergence is linked to the Rényi divergence (Rényi, 1961), which has been used to measure domain similarity (Van Asch and Daelemans, 2010). Given that we are trying to learn from conditional distributions which are zero everywhere except for one target token, we are interested in experimenting with values of $\alpha > 1$, which should push the model towards generalizing, while having $\alpha \in (0, 1)$ should force the model to concentrate probability mass on training examples. Since we are here working with normalized distribution, we obtain the following objective:

$$
Obj_\alpha(\theta) = \frac{1}{\alpha(\alpha - 1)} \sum_{(x,y) \in D} (p_\theta(y|\theta))^{1-\alpha}.
$$

3.2 $\beta$-divergences

The $\beta$-divergence, also called density power divergence was introduced by Basu et al. (1998) as a robust estimation method, which showed it to be consistent for parameter estimation, with asymptotic normality of the estimation error (Basu et al., 1998, Theorem 2). The full expression of $D_\beta(p||q)$ is shown in Table 1.\(^5\) It can be seen as a Bregman divergence (Bregman, 1967) also derived using the generalized logarithm\(^2\) (Eq. 1). The motivation is to obtain divergences that are robust to outliers, which is the case for values of $\beta > 1$: choosing $\beta = 2$ gives us the $L_2$-loss, while $\beta \to 1$ gives the Kullback-Leibler divergence $D_{KL}(p||q)$ as a limiting case. Hence, choosing a value close to 1 while larger is supposed to provide a compromise between the efficiency of the Kullback-Leibler divergence and the robustness of the $L_2$-loss. Similarly, we can expect to give more importance to outliers (which we suppose to be the low-frequency tokens) by choosing $\beta < 1$. The objective becomes:

$$
Obj_\beta(\theta) = \frac{1}{\beta(\beta - 1)} \times \sum_{(x,y) \in D} \left[ (\beta - 1) \sum_{y' \in Y} (p_\theta(y'|\theta))^\beta - \beta(p_\theta(y|\theta))^{\beta - 1} \right].
$$

3.3 $\gamma$-divergences

Eguchi and Kato (2010) introduced the $\gamma$-divergence as a modification of the $\beta$-divergence, with the specific goal of obtaining a scale-invariant version of the robust $\beta$-divergence,\(^6\) also showing it to be consistent for parameter estimation, with asymptotic normality of the estimation error (Eguchi and Kato, 2010, Section 3). This divergence has notably been used for the estimation of parameters without normalizing the output probability distribution (Takenouchi and Kanamori, 2015).\(^7\) The general expression of the $\gamma$-divergence is shown in Table 1. While the use of the log non-linearity makes this divergence non-separable, which at first glance could be thought to complicate computation in practice, our motivation for using it is its scale-invariance, which we will discuss in Section 4.3. Applied to our prob-

\(^2\) and $\beta$-divergences are related to the Tsallis entropy, which can be seen as a deformation of the Shannon entropy using this same generalized logarithm: see Appendix A from Cichocki and Amari (2010). Recently, Blondel et al. (2018) showed how to build Fenchel-Young Losses from the same Tsallis entropies.

\(^3\)Having $p_i > 0$ will enforce $q_i > 0$.

\(^4\)Having $p_i = 0$ will enforce $q_i = 0$.

\(^5\) For readability, we keep the notation and values used in Cichocki and Amari (2010) instead of Basu et al. (1998).

\(^6\) Scale-invariance here means that the divergence should remain unchanged when one or both of the measures it is applied to are multiplied by scalars.

\(^7\) However, this particular method is not suitable in our case because of the sparsity of our data.
lem, the objective becomes:

\[ \text{Obj}_t(\theta) = \sum_{(x,y) \in D} \left[ \log p_\theta(y|x) - \frac{1}{\gamma} \log \sum_{y' \in \mathcal{Y}} (p_\theta(y'|x))^{\gamma} \right]. \]

(2)

4 Accelerating Learning

In order to use the previously defined objective functions, we need to compute the model probabilities \( p_\theta(y|x) \), which are usually obtained using a softmax function:

\[ p_\theta(y|x) = \frac{\exp (s_\theta(x, y))}{\sum_{y' \in \mathcal{Y}} \exp (s_\theta(x, y'))} \]

applied on scores (or logits) \( s_\theta(x, y') \) output by the model for every possible target token \( y' \in \mathcal{Y} \). However, as explained earlier, \( \mathcal{Y} \) can be very large, hence computing all the scores and summing them is extremely slow. We choose to follow the strategies employed by Jozefowicz et al. (2016): approximating the softmax using importance sampling (Bengio and Sébédio, 2003, 2008), and Noise Contrastive Estimation (NCE; Gutmann and Hyvärinen 2010; Mnih and Teh 2012). Besides, the divergences presented in Section 3 can be applied to positive measures. Hence, a possible third direction is to instead approximate the objectives to the un-normalized model distributions. All the objectives presented in this section are explained in Appendix A.4.

4.1 Approximated softmax

Plugging the first solution into our objectives is straightforward: using self-importance sampling to approximate directly the multinomial classification probability, we compute \( p_\theta \) via an approximated softmax:

\[ p_\theta(y|x) \approx \frac{\exp (s_\theta(x, y))}{p_n(y)} \frac{p_n(y)}{p_n(y)} + \sum_{i=1}^{k} \frac{\exp (s_\theta(x, \hat{y}_i))}{p_n(\hat{y}_i)} \]

where \( p_n \) is an auxiliary distribution chosen to reflect the training data while still being easy to sample from, and \( k \ll |\mathcal{Y}| \) is the number of samples drawn.

4.2 Adapting Noise-Contrastive Estimation

Noise-Contrastive Estimation consists of training the model for a surrogate binary classification task where, given examples from the mixture:

\[ \frac{1}{k + 1} p_D + \frac{k}{k + 1} p_n \]

we learn to discriminate between examples coming from data and samples coming from the auxiliary noise distribution \( p_n \). Minimizing the NCE loss function has been shown to be equivalent to minimizing a Bregman divergence (Gutmann and Hirayama, 2011); however, the transformation that we need to apply to the associated generating function (Pihlaja et al., 2012) in order to obtain a power divergence is not straightforward. Instead, with the posterior classification probability:

\[ p(C = \text{True}|y, x) = \frac{p_\theta(y|x)}{p_\theta(y|x) + kp_n(y)} \]

(3)

Noting \( p_\theta^C \) the positive measure on \( \mathcal{X} \times \mathcal{Y} \):

\[ p_\theta^C : (x, y) \rightarrow p_\theta(C = \text{True}|y, x) \]

we can show that the NCE objective function can be derived from the following expression:

\[ D_{KL}(p_D^C||p_\theta^C) + D_{KL}(1 - p_D^C||1 - p_\theta^C) \]

Knowing this, we simply need to replace \( D_{KL} \) by the other divergences to derive the \( \alpha, \beta \) and \( \gamma \) objective functions associated to this surrogate classification task. It is interesting to note that the power transformations will here be applied on the posterior classification probabilities \( p_\theta^C \) instead of categorical probabilities \( p_\theta \).

4.3 Working With Positive Measures

The three divergences presented in Section 3 are defined on positive measures: in theory, we can simply use the \( \exp \) function on the scores \( s_\theta \) and do not need to normalize them:

\[ \text{Obj}(\theta) = D(p_D||\exp(s_\theta)) \]

However, neither the \( \alpha \) and \( \beta \) divergences are scale invariant (see right column of Table 1 and Cichocki and Amari 2010). We can show that working with an un-normalized model distribution will, in both those cases, give an objective proportional to the scale of the model, allowing the divergence

---

8 The same objective is called ‘Ranking NCE’ by Ma and Collins (2018).

9 The full derivation is given in Appendix A.2.
Table 1: Complete expressions of $\alpha$, $\beta$, and $\gamma$ divergences between two positive measures $p = (p_i)_{i=1}^{n}$ and $q = (q_i)_{i=1}^{n}$ on $\mathbb{R}^{+}$, as well as their scaling properties. Note that the $\alpha$-divergence simplifies if we restrict them to the set of normalized probability densities (if $\sum_{i=1}^{n} p_i = \sum_{i=1}^{n} q_i = 1$).

5 Experiments

Our goal is to compare the effect of the various objective functions we proposed in Sections 3 and 4, and especially study how the values of $\alpha$, $\beta$ and $\gamma$ affect learning and performance, overall as well as on low-frequency words. Since each model is trained with a different objective function, the training scores are not comparable. Hence, we use the validation cross-entropy and perplexity at each epoch as a way to track progress during training.

5.1 Datasets

We perform our experiments on two widely used, reasonably sized datasets: the Penn Treebank (PTB; Mikolov et al. 2011a) and WikiText-2 (WT2; Merity et al. 2017). The PTB, heavily preprocessed, retains the 10k most frequent words in its vocabulary, while the others tokens are replaced by a common $<unk>$ token. The WT2,
about two times larger, retains words that appear at least three times in the training data in its vocabulary, which makes it 33,278 words. To study the impact of our various objective functions on a vocabulary containing very rare words, we also experiment with a version on the PTB to which we only apply limited preprocessing, allowing us to keep its full training vocabulary, which contains 39030 words.

5.2 Experimental setup

We based our setup on the ASGD weight-dropped LSTM (AWD-LSTM) models of Merity et al. (2018b), since to the best of our knowledge they give state-of-the-art results on the PTB and WT2 for models that are build with a softmax output layer and do not use any adaptive method (as the pointer sentinel LSTM (Merity et al., 2017), the neural cache (Grave et al., 2017b) and the dynamic evaluation (Krause et al., 2018)). Our models\(^{11}\), replications of their 3-layer LSTMs with tied input and output representations, were built using their implementation.\(^{12}\) For each dataset, we follow their choice of hyperparameters, only modifying the objective functions.

For our sampling-based objectives, we use \(k = 1024\) samples, and the unigram distribution as \(p_n\). In the case of objectives derived from binary NCE, to avoid issues with the phenomenon of self-normalization and consistency issues (Ma and Collins, 2018) we chose to use blackout (Ji et al., 2015). Indeed, this method amounts to using a noise distribution which depends on the model probabilities, making the normalization term disappear from the posterior classification probabilities of Eq. 3. Hence, we have an objective function that is fast to compute without any supplementary assumption, and the negative samples are still drawn from the unigram distribution. For both AS and NCE, in our setting, the computation time is reduced to about 40% of the time taken by MLE on the WT2, and 50% on the PTB.

6 Results and discussion

We turn to describe the results of our experiments.

6.1 Qualitative results on the full vocabulary

PTB

To observe the behavior of the proposed objectives for various choices of \(\alpha, \beta\) and \(\gamma\), we plot the validation cross-entropy at the beginning of learning of models on PTB equipped with a full training vocabulary. We choose values of 0.9 and 1.1 for the power parameters, to experiment with an objective on each side of the baseline MLE objective. We split the words according to frequency into 5 buckets, in order for the buckets to represent have equal size based on word counts, and display both the global cross-entropy and the cross-entropy on the lowest frequency bin in Figure 1. A value of \(\alpha > 1\) seems to initially behave better, especially for rare words. This could be expected: intuitively, these values of \(\alpha\) should make the model ‘stretch’ the probability mass. However, as learning progresses, this phenomenon lessens, and the performance on rare words gets worse. A value of \(\beta < 1\), supposed to make the model less robust to outliers, seems to make learning faster initially, particularly on rare words, but again improvements lessen. Choosing \(\alpha < 1\) or \(\beta > 1\) gives a worse cross-entropy overall. This ‘inverted’ similarity between the behaviors induced by choices of \(\alpha\) and \(\beta\), and the links between the two divergences, have been explored by Patra et al. (2013). Finally, choosing \(\gamma > 1\) gives very interesting results, allowing for a better cross-entropy on rare words while retaining the same overall performance than MLE.

6.2 Penn Treebank and WikiText-2

In this section, we present the results of exploratory experiments. We fully train models with the proposed objectives, for a variety of power parameters. For the PTB, the final validation perplexities are presented in Table 2, while we present the final validation cross-entropies for the objectives derived from MLE, on 5 frequency buckets, in Figure 2. For the WT2, they are shown in Table 3 and Figure 3.

With exact objectives: for both corpora, the results for the high and low-frequency buckets seem to confirm that values of \(\alpha, \gamma\) that are smaller and larger than 1 can help prioritizing learning towards the frequent and rare words. The effect of \(\beta\) depending on frequency seems lighter, especially on the PTB: we hypothesize that this is due to the vocabulary being cut short, and containing no very
Validation cross-entropy values for the best epoch obtained for models trained with $\text{Obj}_\alpha$ (top), $\text{Obj}_\beta$ (middle) and $\text{Obj}_\gamma$ (bottom) on the PTB. Words are grouped into 5 buckets of equal size, following their frequencies. We display values for each bucket from the most frequent words (left) to less frequent ones (right).

![Validation Cross-entropy](image)

**Table 2:** Best validation perplexities obtained on the PTB with $\text{Obj}_\alpha$, $\text{Obj}_\beta$, and $\text{Obj}_\gamma$, derived from MLE. In each cell we give on top the validation perplexity, and below, the ‘counterpart’ to perplexity corresponding to the training objective — which is the value being optimized. Each color corresponds to a different objective: values cannot be compared for varying $\alpha$, $\beta$ and $\gamma$.

<table>
<thead>
<tr>
<th>Objective</th>
<th>$\text{Obj}_\alpha$</th>
<th>$\text{Obj}_\beta$</th>
<th>$\text{Obj}_\gamma$</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLE</td>
<td>0.9</td>
<td>65.8</td>
<td>63.3</td>
</tr>
<tr>
<td></td>
<td>0.99</td>
<td>60.7</td>
<td>61.0</td>
</tr>
<tr>
<td>AS</td>
<td>1.0</td>
<td>60.1</td>
<td>61.0</td>
</tr>
<tr>
<td></td>
<td>1.01</td>
<td>60.8</td>
<td>61.1</td>
</tr>
<tr>
<td></td>
<td>1.1</td>
<td>63.0</td>
<td>63.9</td>
</tr>
<tr>
<td>NCE</td>
<td>0.5</td>
<td>1407.5</td>
<td>88.0</td>
</tr>
<tr>
<td></td>
<td>0.9</td>
<td>61.3</td>
<td>62.8</td>
</tr>
<tr>
<td></td>
<td>0.99</td>
<td>61.7</td>
<td>61.7</td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>61.7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.01</td>
<td>61.8</td>
<td>61.6</td>
</tr>
<tr>
<td></td>
<td>1.1</td>
<td>65.6</td>
<td>61.6</td>
</tr>
<tr>
<td></td>
<td>1.5</td>
<td>97.7</td>
<td>127.6</td>
</tr>
</tbody>
</table>

Table 4: Final validation results obtained on the PTB and WT2, with the exact objectives $\text{Obj}_\alpha$, $\text{Obj}_\beta$, and $\text{Obj}_\gamma$, derived from MLE. In each cell we give on top the validation perplexity, and below, the ‘counterpart’ to perplexity corresponding to the training objective — which is the value being optimized. Each color corresponds to a different objective: values cannot be compared for varying $\alpha$, $\beta$ and $\gamma$.

<table>
<thead>
<tr>
<th>Objective</th>
<th>$\text{Obj}_\alpha$</th>
<th>$\text{Obj}_\beta$</th>
<th>$\text{Obj}_\gamma$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset</td>
<td>PTB</td>
<td>PTB</td>
<td>PTB</td>
</tr>
<tr>
<td>0.9</td>
<td>65.8</td>
<td>63.3</td>
<td>63.2</td>
</tr>
<tr>
<td></td>
<td>63.8</td>
<td>63.9</td>
<td>63.3</td>
</tr>
<tr>
<td>0.99</td>
<td>60.7</td>
<td>61.0</td>
<td>60.6</td>
</tr>
<tr>
<td></td>
<td>62.4</td>
<td>60.6</td>
<td>65.9</td>
</tr>
<tr>
<td>1.0</td>
<td>60.1</td>
<td>61.0</td>
<td>60.5</td>
</tr>
<tr>
<td></td>
<td>61.4</td>
<td>64.9</td>
<td>64.9</td>
</tr>
<tr>
<td>1.01</td>
<td>60.0</td>
<td>61.1</td>
<td>61.0</td>
</tr>
<tr>
<td></td>
<td>64.9</td>
<td>64.9</td>
<td>64.9</td>
</tr>
<tr>
<td>1.1</td>
<td>63.0</td>
<td>63.9</td>
<td>63.3</td>
</tr>
<tr>
<td></td>
<td>68.3</td>
<td>68.1</td>
<td></td>
</tr>
</tbody>
</table>

With approximated objectives: for objectives derived from AS and NCE, we observe far less impact of the choice of the power parameter on frequent or rare words, which is probably due to the fact that only a small subset of the vocab-

---

13 See Appendix A.3.
14 See figures in Appendix A.5.
Validation cross-entropy values for the best epoch obtained for models trained with $\text{Obj}_\alpha$ (top), $\text{Obj}_\beta$ (middle) and $\text{Obj}_\gamma$ (bottom) on the WT2. Words are grouped into 5 buckets of equal size, following their frequencies. We display values for each bucket from the most frequent words (left) to less frequent ones (right).

Table 3: Best validation perplexities obtained on the WT2 with $\text{Obj}_\alpha$, $\text{Obj}_\beta$, and $\text{Obj}_\gamma$, derived from MLE, and approximated objectives AS and NCE, on single models with the same initialization.

<table>
<thead>
<tr>
<th>Objective</th>
<th>$\text{Obj}_\alpha$</th>
<th>$\text{Obj}_\beta$</th>
<th>$\text{Obj}_\gamma$</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLE</td>
<td>0.9</td>
<td>0.99</td>
<td>0.99</td>
</tr>
<tr>
<td></td>
<td>75.8</td>
<td>68.3</td>
<td>70.1</td>
</tr>
<tr>
<td></td>
<td>66.5</td>
<td>65.6</td>
<td>65.9</td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>64.9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.01</td>
<td>65.6</td>
<td>65.4</td>
</tr>
<tr>
<td></td>
<td>1.1</td>
<td>66.7</td>
<td>68.3</td>
</tr>
<tr>
<td>AS</td>
<td>0.5</td>
<td>203.6</td>
<td>82.1</td>
</tr>
<tr>
<td></td>
<td>0.9</td>
<td>69.9</td>
<td>65.6</td>
</tr>
<tr>
<td></td>
<td>0.99</td>
<td>67.4</td>
<td>65.0</td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>65.3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.01</td>
<td>67.5</td>
<td>65.1</td>
</tr>
<tr>
<td></td>
<td>1.1</td>
<td>69.8</td>
<td>67.8</td>
</tr>
<tr>
<td></td>
<td>1.5</td>
<td>100.7</td>
<td>211.1</td>
</tr>
<tr>
<td></td>
<td>0.5</td>
<td>2401.2</td>
<td>84.1</td>
</tr>
<tr>
<td></td>
<td>0.9</td>
<td>70.6</td>
<td>66.6</td>
</tr>
<tr>
<td></td>
<td>0.99</td>
<td>66.8</td>
<td>65.7</td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>65.3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.01</td>
<td>65.8</td>
<td>65.9</td>
</tr>
<tr>
<td></td>
<td>1.1</td>
<td>64.7</td>
<td>68.3</td>
</tr>
<tr>
<td></td>
<td>1.5</td>
<td>83.8</td>
<td>196.4</td>
</tr>
<tr>
<td>NCE</td>
<td>1.0</td>
<td>65.8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.01</td>
<td>65.8</td>
<td>65.9</td>
</tr>
<tr>
<td></td>
<td>1.1</td>
<td>64.7</td>
<td>68.3</td>
</tr>
<tr>
<td></td>
<td>1.5</td>
<td>83.8</td>
<td>196.4</td>
</tr>
</tbody>
</table>

Figure 4: Validation and test perplexities obtained for particular values of $\alpha$, $\beta$ or $\gamma$ with sampling-based objectives in 4 possible pairings of AS and NCE-based objectives trained on the PTB and WT2, on single models with the same initialization.

6.3 Searching for the Optimal Power Parameter

In order to verify the potential benefits of our generalization of sampling-based objectives, we use these preliminary results to search for the ‘best’ power parameter, and check that improvements are consistent for several versions of the model.
Table 5: Best validation and test perplexities obtained on the best performing configurations of power parameter for both corpora and each category of objectives, averaged over 5 models with different initializations.

<table>
<thead>
<tr>
<th>Objective</th>
<th>Validation</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MLE</td>
<td>60.7</td>
<td>58.6</td>
</tr>
<tr>
<td>AS KL</td>
<td>62.2</td>
<td>59.8</td>
</tr>
<tr>
<td>$\alpha = 0.95$</td>
<td>61.2</td>
<td>59.9</td>
</tr>
<tr>
<td>NCE KL</td>
<td>61.5</td>
<td>59.2</td>
</tr>
<tr>
<td>$\beta = 1.1$</td>
<td>61.2</td>
<td>59.0</td>
</tr>
<tr>
<td>WT2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MLE</td>
<td>65.5</td>
<td>62.8</td>
</tr>
<tr>
<td>AS KL</td>
<td>65.1</td>
<td>62.6</td>
</tr>
<tr>
<td>$\gamma = 1.075$</td>
<td>64.8</td>
<td>62.1</td>
</tr>
<tr>
<td>NCE KL</td>
<td>65.8</td>
<td>63.4</td>
</tr>
<tr>
<td>$\alpha = 1.1$</td>
<td>64.7</td>
<td>62.0</td>
</tr>
</tbody>
</table>

Acknowledgments

We thank the anonymous reviewers for helpful feedback. We gratefully acknowledge the support of Huawei Technologies.

References


Yoshua Bengio and Jean-Sébastien Senécal. 2003. Quick training of probabilistic neural nets by importance sampling. In *Proceedings of the conference on Artificial Intelligence and Statistics (AISTATS)*.


Tomas Mikolov, Stefan Kombrink, Lukas Burget, and Jan Honza Cernocky. 2011a. Empirical evaluation and combination of advanced language modeling techniques. In Interspeech. ISCA.


A Supplemental Material

A.1 Scaling properties of power objectives

The scaling properties of both $\alpha$ and $\beta$-divergence, shown in Table 1, imply that if we do not enforce the scale of the model to be 1 by normalizing it, the model will be unable to learn. Indeed, we can rewrite $D_\alpha(p_D||\exp(s_0))$ as:

$$D_\alpha(p_D||\exp(s_0)) = D_\alpha(p_D||p_0 \times Z_0)$$

$$= \frac{1}{\alpha(\alpha - 1)} \sum_{(x,y) \in D} (Z_0(x))^{1-\alpha} (p_0(y|x))^{1-\alpha}$$

That makes the objective possible to minimize by simply minimizing $Z_0(x)\forall x \in X$ — which does not imply any learning from the data. It is also the case with $D_\beta(p_D||\exp(s_0))$:

$$D_\beta(p_D||\exp(s_0)) = D_\beta(p_D||p_0 \times Z_0)$$

$$= \frac{1}{\beta(\beta - 1)} \sum_{(x,y) \in D} [-\beta (Z_0(x))^{1-\beta} (p_0(y|x))^{1-\beta}$$

$$+ (Z_0(x))^\beta \sum_{y' \in Y} (p_0(y'|x))^\beta]$$

$$= \frac{1}{\beta(\beta - 1)} \sum_{(x,y) \in D} (Z_0(x))^{1-\beta} \times$$

$$[-\beta (p_0(y|x))^{1-\beta} + Z_0(x) \sum_{y' \in Y} (p_0(y'|x))^\beta]$$

However, it is easy to derive that it is not the case for the $\gamma$-divergence:

$$D_\gamma(p_D||p_0)$$

$$= \sum_{(x,y) \in D} \log p_0(y|x) - \frac{1}{\gamma} \log \sum_{y' \in Y} (p_0(y'|x))^\gamma$$

$$= \sum_{(x,y) \in D} s_0(x,y) - \log Z_0(x)$$

$$- \frac{1}{\gamma} \log \left(\sum_{y' \in Y} \exp(\gamma s_0(x,y'))\right)$$

$$= \sum_{(x,y) \in D} \left[s_0(x,y) - \frac{1}{\gamma} \log \sum_{y' \in Y} \exp(\gamma s_0(x,y'))\right]$$

$$= D_\gamma(p_D||\exp(s_0))$$

A.2 NCE as a binary divergence

The divergence $D_{KL}(p_D^C||p_0^C)$ can be written as:

$$D_{KL}(p_D^C||p_0^C) = \sum_{(x,y) \in X \times Y} p_D^C(x,y) \log \frac{p_D^C(x,y)}{p_0^C(x,y)}$$

$$= \sum_{(x,y) \in X \times Y} p_D(y|x) \log \frac{p_D(y|x)}{p_0(y|x) + kp_n(y)}$$

We can remove the first term, which is not dependent on $\theta$, and will not intervene in the objective function. If we do the same with the divergence $D_{KL}(1 - p_D^C||1 - p_0^C)$ and add them, we obtain the following:

$$- \sum_{(x,y) \in X \times Y} \left(\frac{p_D(y|x)}{p_D(y|x) + kp_n(y)} \log \frac{p_D(y|x)}{p_0(y|x) + kp_n(y)}ight)$$

$$\quad + kp_n(y) \log \frac{kp_n(y)}{p_0(y|x) + kp_n(y)}$$

With NCE, we consider that examples are coming from the mixture $\frac{1}{1+k}p_D + \frac{k}{1+k}p_n$, instead of being uniformly spread, which transforms the objective into:

$$- \sum_{(x,y) \in X \times Y'} \left(\frac{p_D(y|x)}{p_D(y|x) + kp_n(y)} \log \frac{p_D(y|x)}{p_0(y|x) + kp_n(y)}ight)$$

$$\quad + kp_n(y) \log \frac{kp_n(y)}{p_0(y|x) + kp_n(y)}$$

We can then rewrite it as a sum of expectations:

$$- \sum_{x \in X} \left(\mathbb{E}_{y \sim p_D(x)} \left[\log \frac{p_0(y|x)}{p_D(y|x) + kp_n(y)}\right]\right)$$

$$\quad + k \mathbb{E}_{y \sim p_n} \left[\log \frac{kp_n(y)}{p_0(y|x) + kp_n(y)}\right]$$

That becomes the NCE objective once we approximate the second expectation over $k$ samples:

$$- \sum_{(x,y) \in D} \left[\log \frac{p_D(y|x)}{p_0(y|x) + kp_n(y)}\right]$$

$$\quad + k \sum_{i=1}^k \log \frac{kp_n(y_i)}{p_0(y_i|x) + kp_n(y_i)}$$

We should note that minimizing the NCE objective is then equivalent to minimizing both a $f$-divergence and a Bregman divergence. For example, by making a variable change between $p_D^C$ and
we can circle back to writing the NCE objective as a Bregman divergence $D_{\phi}(p_D||p_\theta)$, with
$$\phi(x) = x \log x - (1 + x) \log(1 + x),$$
as shown in Gutmann and Hirayama (2011).

### A.3 Objective-specific ‘perplexity’

See Figure 5. We can observe that the behavior of the objective-specific counterparts to perplexity closely mirrors it, even when the values are quite distant.

![Figure 5](image-url)

Figure 5: Validation results by epoch obtained on the PTB with the exact objectives, derived from MLE, during training. We give the validation perplexity (dotted gray) and the ‘counterpart’ to perplexity corresponding to the training objective (in color). Each color corresponds to a different objective, and we use different shades to indicate that changing the value of the power parameter makes the tracked values different.

### A.4 Complete sampling-based objectives

See Table 6.

### A.5 Detailed performance of sampling based-objectives

See Figures 6, 7, 8 and 9.
Table 6: Complete objectives of power generalizations of the Approximated Softmax and Noise Contrastive Estimation objective functions, based on $\alpha$, $\beta$, and $\gamma$ divergences. All the samples $(\hat{y}_i)_{i=1}^k$ are drawn from the auxiliary distribution $p_n$. 

<table>
<thead>
<tr>
<th>Objective</th>
<th>(- \sum_{(x,y) \in D} \left[ s_g(x, y) - \log p_n(y) - \log \sum_{i=1}^k \exp (s_g(x, \hat{y}_i) - \log p_n(\hat{y}_i)) \right] )</th>
</tr>
</thead>
<tbody>
<tr>
<td>\text{Approximated Softmax}</td>
<td>( \alpha \in \mathbb{R} \setminus {0, 1} ) [ \frac{1}{\alpha(\alpha-1)} \sum_{(x,y) \in D} \left( \frac{\exp (s_g(x,y))}{\sum_{i=1}^k \exp (s_g(x,\hat{y}<em>i))} \right)^{1-\alpha} + \sum</em>{i=1}^k \left( \frac{k p_n(\hat{y}_i)}{\exp (s_g(x,\hat{y}_i))} \right)^{1-\alpha} ]</td>
</tr>
<tr>
<td>( \beta \in \mathbb{R} \setminus {0, 1} ) [ \frac{1}{\beta(\beta-1)} \sum_{(x,y) \in D} \left[ \frac{\sum_{i=1}^k \exp (\beta s_g(x,y,\hat{y}_i) - \log p_n(\hat{y}<em>i))}{\sum</em>{i=1}^k \exp (s_g(x,\hat{y}_i) - \log p_n(\hat{y}_i))} \right]^{\beta-1} - \frac{1}{\beta} \left( \frac{\exp (s_g(x,y) + k p_n(y))^{\beta}}{\exp (s_g(x,y) + k p_n(y))} \right)^{\beta-1} ]</td>
<td></td>
</tr>
<tr>
<td>( \gamma \in \mathbb{R} \setminus {0, 1} ) [ \sum_{(x,y) \in D} \left[ - \log \frac{\exp (s_g(x,y))}{\exp (s_g(x,y) + k p_n(y))} - \frac{1}{\gamma-1} \log \sum_{i=1}^k \left( \frac{k p_n(\hat{y}_i)}{\exp (s_g(x,\hat{y}_i))} \right)^{\gamma-1} \right] + \frac{1}{\gamma} \log \left( \frac{\exp (\gamma s_g(x,y) + k p_n(y))}{\exp (s_g(x,y) + k p_n(y))} \right)^{\gamma-1} ]</td>
<td></td>
</tr>
</tbody>
</table>

| \text{Noise Contrastive Estimation} | \( \alpha \in \mathbb{R} \setminus \{0, 1\} \) \[ \frac{1}{\alpha(\alpha-1)} \sum_{(x,y) \in D} \left( \frac{\exp (s_g(x,y))}{\exp (s_g(x,y) + k p_n(y))} \right)^{1-\alpha} + \sum_{i=1}^k \left( \frac{k p_n(\hat{y}_i)}{\exp (s_g(x,\hat{y}_i))} \right)^{1-\alpha} \] |
| \( \beta \in \mathbb{R} \setminus \{0, 1\} \) \[ \sum_{(x,y) \in D} \left[ \frac{\sum_{i=1}^k \exp (\beta s_g(x,y,\hat{y}_i) + k p_n(y))}{\sum_{i=1}^k \exp (s_g(x,\hat{y}_i) + k p_n(y))} \right]^{\beta-1} + \sum_{i=1}^k \left( \frac{k p_n(\hat{y}_i)}{\exp (s_g(x,\hat{y}_i) + k p_n(y))} \right)^{\beta-1} \] |
| \( \gamma \in \mathbb{R} \setminus \{0, 1\} \) \[ \sum_{(x,y) \in D} \left[ - \log \frac{\exp (s_g(x,y))}{\exp (s_g(x,y) + k p_n(y))} - \frac{1}{\gamma-1} \log \sum_{i=1}^k \left( \frac{k p_n(\hat{y}_i)}{\exp (s_g(x,\hat{y}_i))} \right)^{\gamma-1} \right] + \frac{1}{\gamma} \log \left( \frac{\exp (\gamma s_g(x,y) + k p_n(y))}{\exp (s_g(x,y) + k p_n(y))} \right)^{\gamma-1} \] |
Figure 6: Validation cross-entropy values for the best epoch obtained for models trained with objectives derived from the AS objective with $\alpha$-divergences (top), $\beta$-divergences (middle) and $\gamma$-divergences (bottom) on the PTB. Words are grouped into 5 buckets of equal size, following their frequencies. We display values for each bucket from the most frequent words (left) to less frequent ones (right).

Figure 7: Validation cross-entropy values for the best epoch obtained for models trained with objectives derived from the NCE objective with $\alpha$-divergences (top), $\beta$-divergences (middle) and $\gamma$-divergences (bottom) on the PTB. Words are grouped into 5 buckets of equal size, following their frequencies. We display values for each bucket from the most frequent words (left) to less frequent ones (right).
Figure 8: Validation cross-entropy values for the best epoch obtained for models trained with objectives derived from the AS objective with $\alpha$-divergences (top), $\beta$-divergences (middle) and $\gamma$-divergences (bottom) on the WT2. Words are grouped into 5 buckets of equal size, following their frequencies. We display values for each bucket from the most frequent words (left) to less frequent ones (right).

Figure 9: Validation cross-entropy values for the best epoch obtained for models trained with objectives derived from the NCE objective with $\alpha$-divergences (top), $\beta$-divergences (middle) and $\gamma$-divergences (bottom) on the WT2. Words are grouped into 5 buckets of equal size, following their frequencies. We display values for each bucket from the most frequent words (left) to less frequent ones (right).