

Deriving Optimal Hyperparameters for Decoding Heuristics

Michael Hu (myhu@princeton.edu)

Department of Computer Science, 35 Olden Street
Princeton, NJ 08544 USA

Abstract

When generating text from language models, we often pair the language model with a decoding heuristic to improve generation quality. However, these decoding heuristics are often parameterized, and we lack a rigorous method for selecting the optimal hyperparameter. This study proposes a calibration framework for selecting optimal hyperparameters by collecting statistics on a validation set. We also propose inverse sampling, a new decoding heuristic pre-calibrated on the validation set by design.

Keywords: natural language generation; calibration; decoding heuristics

Introduction

Language models estimate the likelihood of a certain token appearing in a sentence, conditional on the tokens that came before it. Recent architectural and training innovations have greatly advanced the state of the art for language modeling (Radford, Narasimhan, Salimans, & Sutskever, 2018; Devlin, Chang, Lee, & Toutanova, 2018; Yang et al., 2019). In particular, language models can now generate lucid and relatively coherent text. For an impressive example of a language model’s generation capabilities, look no further than the *Ovid’s Unicorn* passage, generated by GPT-2 (Radford et al., 2019).

In terms of model architecture, GPT-2 is an autoregressive (AR) language model. AR language models estimate conditional probability distributions, predicting the likelihood of a token based on the forward or backward context. As demonstrated by Caccia et al. (2018), AR language models remain the dominant language model objective class for natural language generation, followed by language GANs and autoencoding language models like BERT (Devlin et al., 2018).

We produce text from an AR language model by sampling tokens from the model’s conditional probability distributions and feeding the tokens back autoregressively into the model. Sampling is often performed with a decoding heuristic, which perturbs the distribution in some way to improve generation quality. Popular decoding heuristics include top- k sampling and temperature scaling (Fan, Lewis, & Dauphin, 2018; Radford et al., 2019; Guo, Pleiss, Sun, & Weinberger, 2017; Caccia et al., 2018).

Currently, there does not exist a rigorous method for selecting hyperparameters for decoding heuristics. To fill this gap, we propose a framework that computes the optimal hyperparameter with respect to a validation set.

Along the way, we demonstrate that GPT-2, arguably the most successful AR language model for natural language generation, is miscalibrated with respect to generation diversity. Specifically, GPT-2 tends to oversample the most likely token, compared to ground-truth human text. Through our framework, we demonstrate that top- k sampling is guaranteed to degrade calibration quality for any choice of k . In terms of controlling diversity, temperature scaling is a more promising technique. Finally, we propose inverse sampling, a decoding heuristic that is calibrated with respect to our measure of diversity for free.

Background

Here, we define the decoding heuristics analyzed in our study. We do not analyze beam search and greedy decoding, as these techniques tend to produce degenerate text (Holtzman, Buys, Forbes, & Choi, 2019). Mathematical formalisms are adapted from Holtzman et al. (2019).

Let $Pr(W_t|w_{<t})$ be the true conditional distribution over the token W_t , where W_t is a random variable. The language model learns $\widehat{Pr}(W_t|w_{<t})$.

Top- k Sampling

Top- k sampling truncates $\widehat{Pr}(W_t|w_{<t})$ to the top- k most likely tokens. It then renormalizes the remaining k tokens into a probability distribution and samples from the new distribution. Formally, let V be the vocabulary set, and let $V_k := \arg \max_{V'} \sum_{w_t \in V'} \widehat{Pr}(w_t|w_{<t})$, where $|V'| = k$ and $V' \subseteq V$.

We define a new probability distribution $\widehat{Pr}^{(k)}$:

$$\widehat{Pr}^{(k)} := \begin{cases} \frac{\widehat{Pr}(w_t|w_{<t})}{\sum_{w_t \in V_k} \widehat{Pr}(w_t|w_{<t})} & \forall w_t \in V_k \\ 0 & \text{otherwise} \end{cases}$$

Top- k sampling then chooses a token from $\widehat{Pr}^{(k)}$. Two recent and popular papers use top- k sampling (Fan et al., 2018; Radford et al., 2019).

Nucleus Sampling

Nucleus sampling, or top- p sampling, truncates the probability distribution to the minimal subset of tokens that make up 100 $p\%$ of the probability mass ($p \in [0, 1]$). Ceteris paribus as above, let $V_p \in V$ be the smallest set such that $\sum_{w_t \in V_p} \widehat{Pr}(w_t|w_{<t}) \geq p$.

We define a new probability distribution $\widehat{Pr}^{(p)}$:

$$\widehat{Pr}^{(p)} := \begin{cases} \frac{\widehat{Pr}(w_t|w_{<t})}{\sum_{w_t \in V_p} \widehat{Pr}(w_t|w_{<t})} & \forall w_t \in V_p \\ 0 & \text{otherwise} \end{cases}$$

Top- p sampling then chooses a token from $\widehat{Pr}^{(p)}$. Note that top- p sampling is simply a dynamic version of top- k sampling. Top- p sampling was proposed by Holtzman et al. (2019).

Temperature Scaling

Language models like GPT-2 do not directly output a probability distribution. To obtain a distribution, the outputs of the neural network $u_{1:|V|}$, called logits, are passed through a softmax function. Temperature scaling simply divides the logits by a constant $T \in (0, \infty)$.

Concretely:

$$\widehat{Pr}^{(T)}(w_t|w_{<t}) := \frac{\exp\{u_t/T\}}{\sum_{i=1}^{|V|} \exp\{u_i/T\}}$$

We then sample from $\widehat{Pr}^{(T)}$. Note that for $T = 1$, $\widehat{Pr}^{(T)} = \widehat{Pr}$. To increase diversity in word choice, one sets $T > 1$. To decrease diversity, one sets $T \in (0, 1)$.

Methods

Our setting is the open-ended generation setting, where we ask the language model to generate the next T tokens when provided with C tokens of context $w_{1:C}$. All experiments were conducted on GPT-2 124M. Language statistics were collected from the Google Billion Words (GBW) corpus (Chelba et al., 2013). For our validation set, we use 1000 randomly sampled lines from GBW.

Approach

Let $w^{(k)}$ be the k th most likely token in the learned distribution $\widehat{Pr}(W_t|w_{<t})$, where $w^{(1)}$ is the most likely token. k is $w^{(k)}$'s token rank. We use token ranks to calibrate the model \widehat{Pr} on a validation set.

Our calibration framework works as follows: First, we obtain frequency statistics of token ranks, as ranked by the model \widehat{Pr} , on the validation set. From these frequency counts, we generate an empirical probability distribution on the token ranks: on average, rank 1 tokens appear 30% of the time, rank 2 tokens 15%, and so forth. These statistics follow a Zipfian distribution (Ha, Sicilia-Garcia, Ming, & Smith, 2002). To estimate the true distribution on ranks, we fit a Zipf distribution to the empirical data using maximum likelihood estimation (MLE) (Bauke, 2007). Let P be the estimated population distribution. Concretely, P is the red line in Figure 1. (Algorithm 1, step 2.)

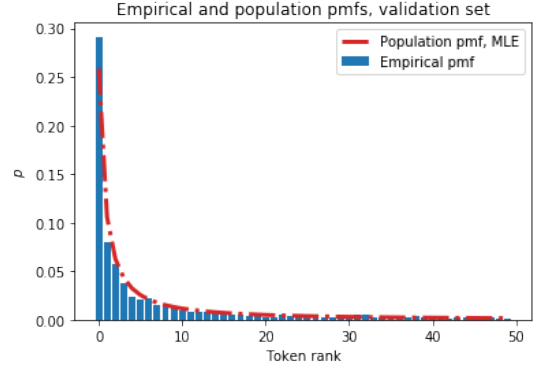


Figure 1: Blue: Empirical distribution on token ranks on a validation set. Calculated on 1000 randomly sampled lines of GBW. Red: MLE of population distribution on token ranks.

Second, we obtain frequency statistics of token ranks for tokens generated by the model *without any seed context*. In essence, we let the model babble, and observe how many times it produces tokens of rank 1, rank 2, etc. From these statistics, we calculate an empirical distribution Q' . (Algorithm 1, step 3.)

If the model is well-calibrated, we would expect P and Q' to be similar. The intuition here is straight forward. If token rank 1 is sampled 30% of the time in the validation set, then we would expect a well-calibrated model to sample token rank 1 around 30% of the time as well. To measure how similar P and Q' are, we take the cross entropy loss between P' and Q' , defined as:

$$CE(P||Q') := \mathbb{E}_{w_t \sim P} \left[\log \frac{1}{Q'(w_t)} \right]$$

Using this framework, we can compute a metric for how well an arbitrary decoding heuristic matches the validation set. Simply modify Step 2 to generate tokens using a decoding heuristic and compute the empirical distribution $Q^{(h)}$, where h refers to the parameter of the decoding heuristic. Taking the cross entropy loss between P and $Q^{(h)}$ then gives us a measure of how badly the decoding heuristic degrades diversity. Smaller values indicate P and $Q^{(h)}$ are more similar, and vice versa.

Critically, we can use this framework to calibrate hyperparameter choices for any decoding heuristic by minimizing the cross entropy loss. Let T be the number of words we generate, and let Q be the empirical distribution for model generations without a decoding heuristic as $T \rightarrow \infty$. We can estimate Q from the empirical babbling distribution Q' by fitting a Zipf distribution to Q' using MLE. (Algorithm 1, step 3.)

Let \widehat{Q} be our estimate of Q . Concretely, \widehat{Q} is the green line in Figure 2(a). We perturb \widehat{Q} using a decoding heuristic to obtain $\widehat{Q}^{(h)}$, where h is the initial hyperparameter. We obtain the optimal hyperparameter h^* by minimizing the cross entropy

loss between P and $\hat{Q}^{(h)}$:

$$h^* = \arg \min_h CE(P || \hat{Q}^{(h)}) = \arg \min_h \mathbb{E}_{w_t \sim P} \left[\log \frac{1}{\hat{Q}^{(h)}(w_t)} \right]$$

Our approach is summarized in Algorithm 1.

Algorithm 1 Calibration with respect to a validation set

- 1: Input: Model $\hat{P}r$, decoding heuristic with parameter h .
- 2: Collect an empirical distribution of token ranks on a validation set. Fit a Zipf distribution P to the empirical distribution using MLE.
- 3: Collect an empirical distribution on model $\hat{P}r$'s babblings. Fit a Zipf distribution \hat{Q} to the empirical distribution using MLE.
- 4: Apply decoding heuristic to \hat{Q} , obtain $\hat{Q}^{(h)}$.

$$h^* = \arg \min_h CE(P || \hat{Q}^{(h)})$$

5: **return** h^* .

Algorithm 1 is fast in practice, taking about 10 minutes to run from start to finish. For our minimizations, we used Brent's Algorithm (Brent, 1973). A core assumption to this calibration approach is that applying the decoding heuristic with hyperparameter h to the individual samples results in the sampling distribution $Q^{(h)}$. We verify this with an empirical result on top- k sampling, demonstrating that the distributions are similar. (See Figure 2(b), green line and purple bars.)

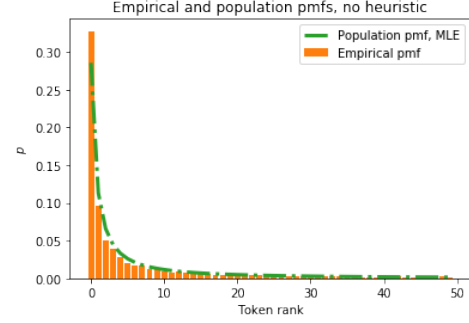
Last, we propose *inverse sampling*. Inverse sampling generates an empirical inverse CDF using the validation set. We estimate the true inverse CDF I using a Gaussian kernel density estimate (Rosenblatt, 1956). To generate, we simply sample $p \sim I$ and convert p into a token index. In practice, inverse sampling is just as fast as other decoding heuristics, taking the same amount of time to run. Furthermore, inverse sampling is pre-calibrated, as it is derived from statistics of the validation set itself.

Results and Discussion

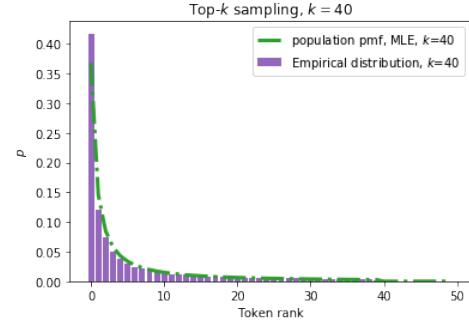
Empirical Distributions

The empirical distributions of various decoding heuristics for $>10,000$ generations are shown in Figure 2. We chose hyperparameter settings from recent papers (Holtzman et al., 2019; Radford et al., 2019).

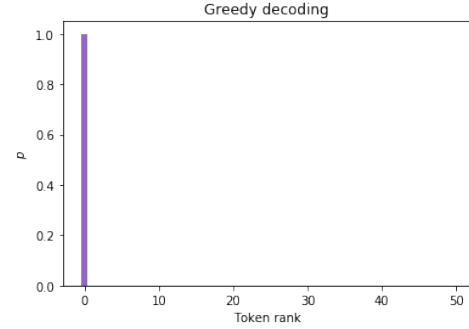
Greedy decoding is known to lead to repetitive output (Holtzman et al., 2019). By Figure 2(c), we also note that greedy decoding fails to capture the variance of the empirical distribution on the validation set. Unsurprisingly, the empirical distribution of greedy decoding is a point mass on token rank 1, which is completely unrepresentative of the Zipfian distribution we would expect to see from a well-calibrated model. As a result, greedy decoding suffers from the greatest cross entropy loss. (Table 1)



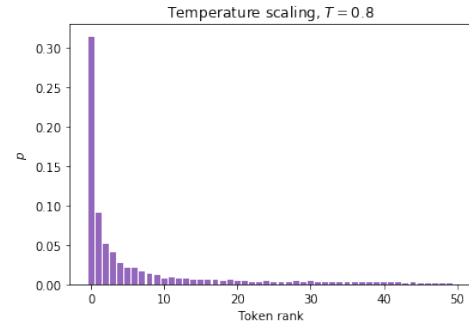
(a) Orange: Empirical distribution on token ranks. Green: MLE of population distribution on token ranks.



(b) Purple: Empirical distribution for top- k sampling, $k = 40$. Green: MLE of population distribution, perturbed using top-40 sampling.



(c) Purple: Empirical distribution for greedy decoding.



(d) Purple: Empirical distribution for temperature scaling, $T = 0.8$

Figure 2: All empirical distributions were calculated over $>10,000$ token generations.

Decoding heuristic	$CE(P Q)$
No heuristic (baseline)	7.712
Top- k sampling, $k = 40$	13.928
Greedy decoding	34.134
Temperature scaling, $T = 0.8$	7.686
Inverse sampling	7.694

Table 1: Cross entropy loss for various hyperparameter settings.

By comparing figures 1 and 2(a), we note that the empirical distribution on the validation set is less peaked than the empirical distribution from babbling. This difference in distributions indicates that GPT-2 is miscalibrated with respect to the validation set. Quantitatively, we found the Zipf coefficient for the validation set to be $\alpha = 1.29$, while the Zipf coefficient for babbling is $\alpha = 1.34$. A larger coefficient assigns more mass to the top tokens, with the distribution approaching a point mass as $\alpha \rightarrow \infty$.

Given that the babbling distribution is more peaked, top- k sampling will always degrade the cross entropy loss. Top- k sampling essentially redistributes mass from the distribution tail to the top- k tokens. Tail tokens suffer additional loss by being set to 0, and the top- k tokens suffer additional loss by being assigned additional mass. Thus, calibration with respect to the validation set worsens when using top- k sampling for any k less than the vocabulary size. We show this empirically in the next section. Temperature scaling with $T = 0.8$ and inverse sampling both improve the cross entropy loss with respect to the baseline.

Calibration Results

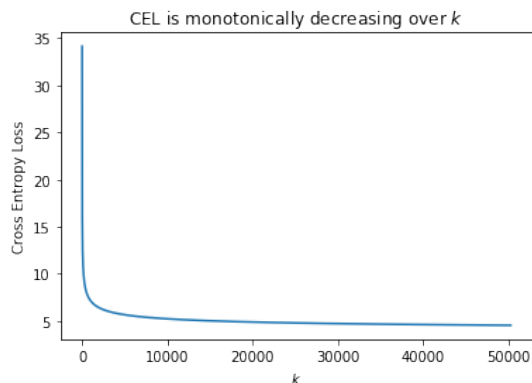


Figure 3: Blue: Cross entropy loss as a function of k . The loss is monotonically decreasing.

We found that the minimizing k for cross entropy loss is $k^* = 50257$, or the size of the vocabulary. As a result, any truncation using top- k sampling is guaranteed to degrade diversity with respect to the validation set. Since nucleus sampling, or top- p sampling, is a dynamic version of top- k sampling, the same results apply.

Results for temperature scaling are more promising. We obtain an optimal temperature $T^* = 0.968$, which corresponds to slightly suppressing diversity (recall section on temperature scaling). $T^* = 0.968$ obtains a cross entropy loss of **6.886**. This value is the smallest cross entropy loss we observe, indicating that our calibration approach works in practice. Generations with $T^* = 0.968$ are coherent and on topic (Figure 4).

Prompt: Models of Cognition is a great class.

GPT-2 response: This lecture aims to give you a working understanding of the brain and all the experiments involved. We will look at models of brain plasticity and how it changes over time. We will talk about using two different means to achieve different results.

Figure 4: Sample GPT-2 generation, with temperature $T^* = 0.968$

Inverse Sampling

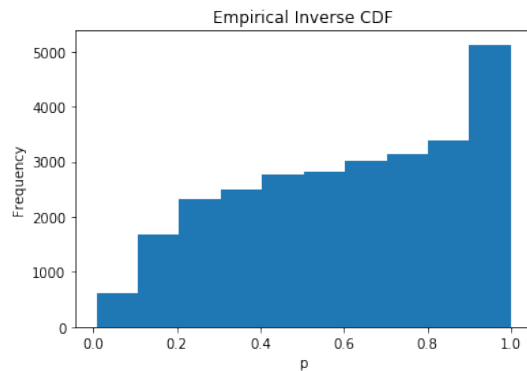


Figure 5: Blue: Empirical inverse CDF, computed from validation set.

Figure 5 shows the empirical inverse CDF, collected on the 1000-line subset of GBW. The empirical inverse CDF has a natural interpretation. If the distribution is uniform over the interval $[0, 1]$, then the model is perfectly calibrated on the validation set. A perfectly uniform distribution would imply that generating from the model without heuristics is sufficient. Unfortunately, our inverse CDF is not uniform—additional evidence that GPT-2 is miscalibrated. The inverse CDF is peaked for large values of p .

This peak in the distribution is due to the fact that GPT-2 tends to overestimate the probability mass of the top tokens, a common issue in neural networks (Guo et al., 2017). Relatedly, this leads to oversampling of the top tokens, which explains the higher peak of the babbling distribution, as discussed in the section on empirical distributions. Overestimating probability mass in the top tokens leads to more samples in the range $[0.9, 1]$, which explains the shape of our inverse CDF.

To generate tokens, we sample a value from the kernel density estimate of our empirical inverse CDF. We then convert the value to a token number. Using inverse sampling obtains a cross entropy loss of **7.694**, a slight improvement over the baseline. However, generations are worse qualitatively and tend to be off topic (Figure 6). More work is required to develop inverse sampling into a robust decoding heuristic.

Prompt: Models of Cognition is a great class.

GPT-2 response: There are numerous books on the subject by well over twenty historians, and many other interesting works in Physics, Chemistry, Geological Science, and other areas. The Autobiography of Perceptual Processes has a wonderful area of technical detail.

Figure 6: A GPT-2 generation using inverse sampling.

Conclusion

We propose a novel framework for selecting hyperparameter values for decoding heuristics. Specifically, we calibrate decoding heuristics to match the diversity of a validation set by minimizing cross entropy loss. We show that top- k sampling and top- p sampling, two popular decoding heuristics, are guaranteed to degrade our measure of diversity. Our calibration method works well when applied to temperature scaling, leading to coherent output and better-calibrated model behavior with respect to the validation set. Finally, we propose inverse scaling, a new sampling method that is pre-calibrated on the validation set.

Related Work

Caccia et al. (2018) showed that diversity and quality are inversely related. Guo et al. (2017) used temperature scaling to calibrate neural networks on standard NLP and computer vision data sets. Wang and Cho (2019) produce text from BERT by modeling it as a Markov random field. Diverse Beam Search is a decoding heuristic that specifically selects for diversity (Vijayakumar et al., 2016).

Acknowledgments

I would like to thank Professor Tom Griffiths and AIs Fred Callaway, Ruairidh McLennan Battleday, and Rachit Dubey for a great semester.

References

Bauke, H. (2007, Jul). Parameter estimation for power-law distributions by maximum likelihood methods. *The European Physical Journal B*, 58(2), 167–173. Retrieved from <http://dx.doi.org/10.1140/epjb/e2007-00219-y> doi: 10.1140/epjb/e2007-00219-y

Brent, R. P. (1973). Chapter 4: An algorithm with guaranteed convergence for finding a zero of a function. In *Algorithms for minimization without derivatives*. Englewood Cliffs, NJ.

Caccia, M., Caccia, L., Fedus, W., Larochelle, H., Pineau, J., & Charlin, L. (2018). Language gans falling short. *CoRR*, abs/1811.02549. Retrieved from <http://arxiv.org/abs/1811.02549>

Chelba, C., Mikolov, T., Schuster, M., Ge, Q., Brants, T., & Koehn, P. (2013). One billion word benchmark for measuring progress in statistical language modeling. *CoRR*, abs/1312.3005. Retrieved from <http://arxiv.org/abs/1312.3005>

Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2018). BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805. Retrieved from <http://arxiv.org/abs/1810.04805>

Fan, A., Lewis, M., & Dauphin, Y. N. (2018). Hierarchical neural story generation. *CoRR*, abs/1805.04833. Retrieved from <http://arxiv.org/abs/1805.04833>

Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). On calibration of modern neural networks. *CoRR*, abs/1706.04599. Retrieved from <http://arxiv.org/abs/1706.04599>

Ha, L. Q., Sicilia-Garcia, E. I., Ming, J., & Smith, F. J. (2002). Extension of zipf’s law to words and phrases. In *Proceedings of the 19th international conference on computational linguistics - volume 1* (p. 1–6). USA: Association for Computational Linguistics. Retrieved from <https://doi.org/10.3115/1072228.1072345> doi: 10.3115/1072228.1072345

Holtzman, A., Buys, J., Forbes, M., & Choi, Y. (2019). *The curious case of neural text degeneration*.

Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training..

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners.

Rosenblatt, M. (1956, 09). Remarks on some nonparametric estimates of a density function. *Ann. Math. Statist.*, 27(3), 832–837. Retrieved from <https://doi.org/10.1214/aoms/1177728190> doi: 10.1214/aoms/1177728190

Vijayakumar, A. K., Cogswell, M., Selvaraju, R. R., Sun, Q., Lee, S., Crandall, D., & Batra, D. (2016). *Diverse beam search: Decoding diverse solutions from neural sequence models*.

Wang, A., & Cho, K. (2019). BERT has a mouth, and it must speak: BERT as a markov random field language model. *CoRR*, abs/1902.04094. Retrieved from <http://arxiv.org/abs/1902.04094>

Yang, Z., Dai, Z., Yang, Y., Carbonell, J. G., Salakhutdinov, R., & Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237. Retrieved from <http://arxiv.org/abs/1906.08237>