On Importance Sampling-Based Evaluation of Latent Language Models

Robert L. Logan IV
Univ. of California, Irvine
rlogan@uci.edu

Matt Gardner
Allen Institute for AI
mattg@allenai.org

Sameer Singh
Univ. of California, Irvine
sameer@uci.edu

Abstract

Language models that use additional latent structures (e.g., syntax trees, coreference chains, and knowledge graph links) provide several advantages over traditional language models. However, likelihood-based evaluation of these models is often intractable as it requires marginalizing over the latent space. Existing methods avoid this issue by using importance sampling. Although this approach has asymptotic guarantees, analysis is rarely conducted on the effect of decisions such as sample size, granularity of sample aggregation, and the proposal distribution on the reported estimates. In this paper, we measure the effect these factors have on perplexity estimates for three different latent language models. In addition, we elucidate subtle differences in how importance sampling is applied, which can have substantial effects on the final estimates, as well as provide theoretical results that reinforce the validity of importance sampling for evaluating latent language models.

1 Introduction

Latent language models are generative models of text that jointly represent the text and the latent structure underlying it, such as: the syntactic parse, coreference chains between entity mentions, or links of entities and relations mentioned in the text to an external knowledge graph. The benefits of modeling such structure include interpretability (Hayashi et al., 2020), better performance on tasks requiring structure (Dyer et al., 2016; Ji et al., 2017), and improved ability to generate consistent mentions of entities (Clark et al., 2018) and factually accurate text (Logan et al., 2019). Unfortunately, demonstrating that these models provide better performance than traditional language models by evaluating their likelihood on benchmark data can be difficult, as exact computation requires marginalizing over all possible latent structures.

Existing approaches evaluate their models by estimating likelihoods using importance sampling, i.e. a weighted average over latent states sampled from a proposal distribution. Although convergence of importance sampled estimates is asymptotically guaranteed, results are typically produced using a small number of samples for which this guarantee does not necessarily apply. Furthermore, these works employ a variety of heuristics—such as sampling from proposal distributions that are conditioned on future *gold* tokens the model is being evaluated on, and changing the temperature of the proposal distribution—without providing measurements of the effect these decisions have on estimated perplexity, and often omitting details crucial to replicating their results.

In this paper, we seek to fill in this missing knowledge, and put this practice on more rigorous footing. First, we review the theory of importance sampling, providing proof that importance sampled perplexity estimates are stochastic upper bounds of the true perplexity—a previously unnoted justification for this evaluation technique. In addition, we compile a list of common practices used in three previous works—RNNG (Dyer et al., 2016), EntityNLM (Ji et al., 2017) and KGLM (Logan et al., 2019)—and uncover a difference in the granularity at which importance samples are aggregated in these works that has a substantial effect on the final estimates. We also investigate a direct marginalization alternative to importance sampling based on beam search that produces strict bounds, and in some cases, has similar performance. Last, we perform experiments to measure the effect of varying sample size, aggregation method, and choice of proposal distribution for these models, an analysis that is missing from previous work. From these results we conclude a set of best practices to be used in future work.

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 2171–2176
July 5 - 10, 2020.

	$\boldsymbol{x}$	Kawhi	to	join	L.A.	Clippers	.	He	...
EntityNLM	$\boldsymbol{t}$	1	0	0	1	1	0	1	...
	$\boldsymbol{e}$	1	∅	∅	2	2	∅	1	...
	$\boldsymbol{l}$	1	1	1	2	1	1	1	...
KGLM	$\boldsymbol{t}$	new	∅	∅	related		∅	related	...
	$\boldsymbol{s}$	∅	∅	∅	kawhi_leonard		∅	kawhi_leonard	...
	$\boldsymbol{r}$	∅	∅	∅	playerFor		∅	reflexive	...
	$\boldsymbol{o}$	kawhi_leonard	∅	∅	la_clippers		∅	kawhi_leonard	...

Figure 1: EntityNLM and KGLM latent states. For EntityNLM, $\boldsymbol{z} = (\boldsymbol{t}, \boldsymbol{e}, \boldsymbol{l})$, where $\boldsymbol{t}$ denotes whether the token is part of a mention, $\boldsymbol{e}$ denotes the coreference cluster, and $\boldsymbol{l}$ denotes the remaining mention length. For KGLM, $\boldsymbol{z} = (\boldsymbol{t}, \boldsymbol{s}, \boldsymbol{r}, \boldsymbol{o})$, where $\boldsymbol{t}$ has the same meaning, and $\boldsymbol{s}$, $\boldsymbol{r}$ and $\boldsymbol{o}$ associate tokens to edges in a knowledge graph.

2 Inference in Latent LMs

In this section, we provide an overview of importance sampling-based inference in latent language models, as well as some key theoretical results.

Latent LMs A *latent language model* is a generative model which estimates the joint distribution $p(\boldsymbol{x}, \boldsymbol{z})$ of a sequence of text $\boldsymbol{x} = (x_1, \ldots, x_T)$ and its underlying latent structure $\boldsymbol{z}$.

In this paper, we focus on three models:

- RNNG (Dyer et al., 2016) which models syntactic structure,
- EntityNLM (Ji et al., 2017) which models coreference chains, and
- KGLM (Logan et al., 2019) which models links to an external knowledge graph.

Example latent states for EntityNLM and KGLM are depicted in Figure 1, showing latent coreference chains and links to the knowledge graph. Other notable latent language models include the NKLM (Ahn et al., 2016) and LRLM (Hayashi et al., 2020); we do not study them since they use alternatives to importance sampling (e.g., the forward-backward algorithm).

Perplexity The standard evaluation metric for language models is *perplexity*:

$$\text{PPL} = \exp\left(-\frac{1}{T}\sum_{t=1}^{T}\log p(x_t|x_{<t})\right), \qquad (1)$$

where $p(x_t|x_{<t})$ is the marginal likelihood of the token x_t conditioned on the previous tokens $x_{<t}$. By the chain rule of probabilities $p(\boldsymbol{x}) = \prod_{t=1}^{T} p(x_t|x_{<t})$. Perplexity can be intractable to compute for latent language models since it requires marginalizing out the latent variable (e.g., $p(\boldsymbol{x}) = \sum_{\boldsymbol{z}} p(\boldsymbol{x}, \boldsymbol{z})$) whose state space is often exponential in the length of the text.

Importance Sampling Existing approaches instead use *importance sampling* (Kahn, 1950) to estimate an approximate marginal probability:

$$\hat{p}(\boldsymbol{x}) = \frac{1}{K}\sum_{k=1}^{K}\frac{p(\boldsymbol{x}, \boldsymbol{z}_k)}{q(\boldsymbol{z}_k)}, \qquad (2)$$

where $q(\boldsymbol{z})$ is an arbitrary *proposal distribution* and $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_K \sim q(\boldsymbol{z})$. It is well known that $\hat{p}(\boldsymbol{x})$ is an unbiased estimator:

$$\mathbb{E}_{\boldsymbol{z}_k \sim q(\boldsymbol{z})}\left[\hat{p}(\boldsymbol{x})\right] = p(\boldsymbol{x}), \qquad (3)$$

provided that $q(\boldsymbol{z}) > 0$ whenever $p(\boldsymbol{z}) > 0$. For proof and further details on importance sampling, we refer the reader to Owen (2013).

Stochastic Upper Bound A consequence of Eqn (3) is that, due to Jensen's inequality:

$$\mathbb{E}_{\boldsymbol{z}_k \sim q(\boldsymbol{z})}\left[\log \hat{p}(\boldsymbol{x})\right] \leq \log p(\boldsymbol{x}). \qquad (4)$$

In other words, *importance sampled estimates of a model's perplexity are stochastic upper bounds of the true perplexity*. This property has not been stated in prior work on latent language modeling, yet is an important consideration since it implies that importance sampled perplexities can be reliably used to compare against existing baselines.

Limiting Behavior Another important observation is that *importance sampled estimates of perplexity are consistent*, e.g., will converge as the number of samples approaches infinity. To prove this, we first observe that $\hat{p}(\boldsymbol{x})$ is consistent, which is a well-known consequence of the strong law of large numbers (Geweke, 1989). Accordingly, $\log \hat{p}(\boldsymbol{x})$ is also consistent due to the continuous mapping theorem (Van der Vaart, 2000).

3 Common Practices

Implementing importance sampling for evaluating latent language models involves a number of decisions that need to be made. We need to select the number of samples, choose the proposal distribution, and decide whether to aggregate importance sampled estimates at the instance or corpus level. We list the practices used in previous work.[1]

Sample Size Typically, only 100 samples are used for computing the perplexity. A notable exception is Kim et al. (2019)'s follow-up to RNNG that uses 1000 samples.

Proposal Distribution Previous work uses proposal distributions $q(\boldsymbol{z}|\boldsymbol{x})$ that are essentially *discriminative* versions of the generative model (e.g., they are models that predict the latent state conditioned on the text), with one key distinction: they are conditioned not only on the sequence of tokens that have been observed so far, but also on *future* tokens that the model will be evaluated on (a trait we will refer to as *peeking*). This conditioning behavior does not contradict any of the assumptions in Eqn's (3) and (4), and is useful in preventing generation of invalid structures (for instance, parse trees with more leaves then there are words in the text), or ones that are inconsistent with future tokens. Dyer et al. (2016) and Kim et al. (2019) also increase the entropy of the proposal distribution by dividing logits by a temperature parameter τ (respectively using $\tau = 1.25$ and $\tau = 2.0$).

Aggregation An oft-overlooked fact (unnoted in previous work) is that Eqn (2) can be substituted into Eqn (1) in multiple ways. Letting $\boldsymbol{x}_C = \{\boldsymbol{x}_1, \ldots \boldsymbol{x}_N\}$ denote a corpus of evaluation data comprised of instances (token sequences) $\boldsymbol{x}_n$, estimates can be formed at the *instance level*:

$$\widehat{\text{PPL}}_I = \exp\left(-\frac{1}{T}\sum_{n=1}^{N}\log \hat{p}(\boldsymbol{x}_n)\right), \quad (5)$$

or at the *corpus level*:

$$\widehat{\text{PPL}}_C = \exp\left(-\frac{1}{T}\log \hat{p}(\boldsymbol{x}_C)\right), \quad (6)$$

i.e., average is either over each instance or the whole corpus.[2] RNNG and EntityNLM perform instance-level aggregation, whereas KGLM performs corpus-level aggregation. Note that these formulations are equivalent when not aggregating over samples, i.e. for non-latent language models.

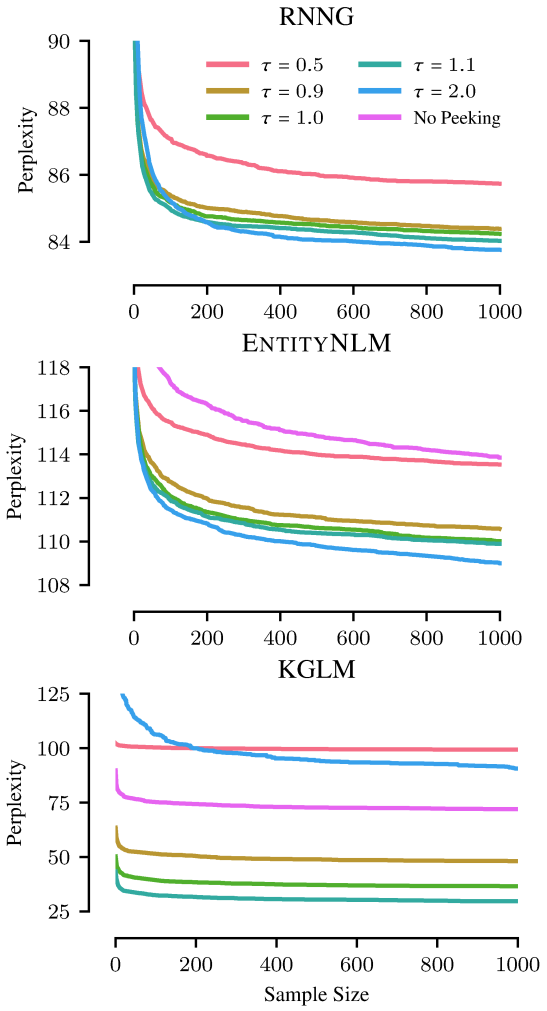

Figure 2: Effect of increasing the number of samples on instance-level perplexity estimates for different proposal distributions.

4 Critical Evaluation

Thus far, research has neglected to measure the effectiveness of the practices detailed in Section 3. In the following section, we perform experiments to determine whether reporting estimates obtained from small sample sizes is warranted, as well as better understand the consequences of peeking and scaling the temperature of the proposal distribution.

Setup For our experiments, we use Kim et al. (2019)'s RNNG implementation[3], and Logan et al. (2019)'s EntityNLM and KGLM implementations[4]. For RNNG and KGLM we use the pre-

[1] Based both on the cited papers and available source code.

[2] One could also consider *token-level* estimates. To our knowledge, these have been unused by existing work.

[3] https://github.com/harvardnlp/urnng

[4] https://github.com/rloganiv/kglm-model

trained model weights. For EntityNLM we train the model from scratch following the procedure described by Ji et al. (2017); results may not be directly comparable due to differences in data preprocessing and hyperparameters. We evaluate models on the datasets used in their original papers: RNNG is evaluated on the Penn Treebank corpus (Marcus et al., 1993), EntityNLM is evaluated on English data from the CoNLL 2012 shared task (Pradhan et al., 2014), and KGLM is evaluated on the Linked WikiText-2 corpus (Logan et al., 2019).

Experiments For EntityNLM and KGLM, we experiment with two kinds of proposal distributions: (1) the standard *peeking* proposal distribution that conditions on future evaluation data, and (2) a *non-peeking* variant that is conditioned only on the data observed by the model (this is akin to estimating perplexity by ancestral sampling). For RNNG we only experiment with peeking proposals, since a non-peeking variant generates invalid parse trees. For the peeking proposal distribution, we experiment with applying temperatures $\tau \in [0.5, 0.9, 1.0, 1.1, 2.0, 5.0]$. We report both corpus-level and instance-level estimates, as well as bounds produced using a direct, beam marginalization method we describe later.

Sample Size We plot instance-level perplexity estimates as sample size is varied in Figures 2 and 3. We observe that the curves are monotonically decreasing in all settings. Consistent with our observation that importance sampled estimates of perplexity are a stochastic upper bound, this demonstrates that the bound is improved as sample size increases. Furthermore, none of the curves exhibit any signs of convergence even after drawing orders of magnitude more samples (Figure 3); the estimated model perplexities continue to improve. Thus, the performance of these models is likely better than the originally reported estimates.

Aggregation Final estimates of perplexity computed using both corpus- and instance-level estimates are provided in Table 1. We note that instance-level estimates are uniformly lower by a wide margin. For example, using a temperature of $\tau = 1.1$ the estimated KGLM perplexity is approximately 10 nats lower using instance-level estimates. This is substantially better than the perplexity of 43 nats reported by Logan et al. (2019).

Proposal Distribution These results also appear to indicate that choice of proposal distribution has a substantial effect on estimated perplexity. However,

	RNNG	Ent	KGLM
Corpus-level			
$\tau = 0.5$	**94.4**	122.6	101.9
$\tau = 0.9$	96.0	122.7	59.3
$\tau = 1.0$	96.7	120.8	48.2
$\tau = 1.1$	97.9	120.7	**41.7**
$\tau = 2.0$	121.6	**120.5**	170.0
$\tau = 5.0$	734.0	152.5	7,468.7
No Peeking	-	131.7	86.8
Instance-level			
$\tau = 0.5$	85.3	113.5	99.3
$\tau = 0.9$	84.4	110.6	48.1
$\tau = 1.0$	84.2	110.0	36.6
$\tau = 1.1$	84.0	109.9	**29.6**
$\tau = 2.0$	**83.8**	**109.0**	90.7
$\tau = 5.0$	97.2	129.6	3,756.1
No Peeking	-	113.9	71.9

Table 1: Final perplexity estimates using different proposal distributions, estimated at both the instance and corpus level. τ is temperature, and *No Peeking* refers to proposal distributions that are not conditioned on future outputs.

	RNNG	Ent	KGLM
$k = 1$	96.3	150.2	153.7
$k = 10$	87.0	147.1	152.6
$k = 100$	84.3	144.5	-

Table 2: Strict perplexity upper bounds obtained by marginalizing over the top-k states predicted by $q(\boldsymbol{z}|\boldsymbol{x})$ using beam search.

it could also be the case that the observed differences in performance across proposal distributions are due to random chance. We investigate whether this is the case for EntityNLM by examining the approximate density of perplexity estimates after drawing 100 importance samples (shown in Figure 4).[5] Our results illustrate that the estimates are relatively stable; although there is some overlap between the better performing temperature values, the order of the modes matches the order reported in Table 1, and there is clear separation from the estimates produced when $\tau = 0.5$ or by the non-peeking proposal distribution. Due to the relative cost of sampling we did not replicate this experiment for RNNG and KGLM.[6]

[5] Obtained by Monte Carlo sampling 100 times.

[6] Figs 3 & 4 took 1 week on a cluster of 15 NVidia 1080Tis.

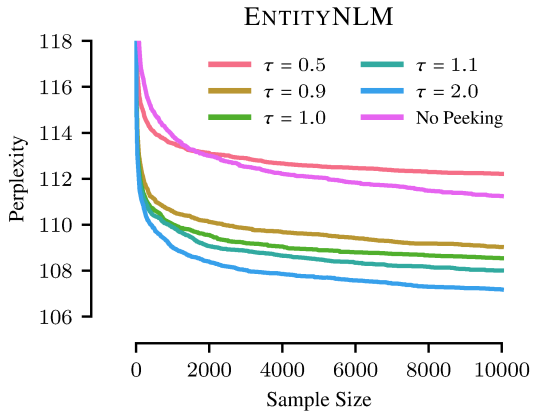

Figure 3: EntityNLM instance-level perplexity estimates as the number of samples is increased to 10K.

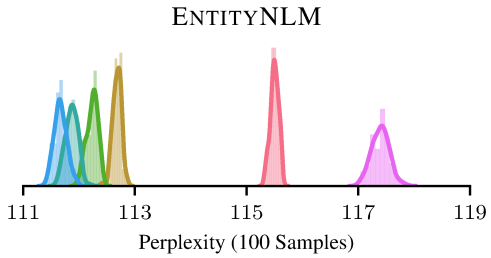

Figure 4: Approximate density of EntityNLM perplexity estimates after drawing 100 importance samples (colors same as Figure 3).

In general, we observe the peeking proposal distributions produce better estimates, and that better performance is obtained using temperatures that slightly increase the entropy of the proposal distribution (e.g., $\tau \in [1.1, 2.0]$), although the ideal amount varies across models. We also observe that the relative performance of proposal distributions is mostly preserved as the number of samples is increased. This suggests that good temperature parameters can be quickly identified by running many experiments with a small number of samples.

Beam Marginalization

An alternative to importance sampling is to directly marginalize over a subset of z values where we expect $p(x|z)$ is large. Specifically, we propose using the top-k most likely values of z identified by performing beam search using the proposal distribution $q(z|x)$. We will refer to this as *beam marginalization*. Because marginalization is only performed over a subset of the space, this method produces a strict upper bound of the true perplexity.

Perplexity bounds obtained using beam marginalization are reported in Table 2. This method produces bounds close to the instance-level importance sampled estimates for RNNG, but does not perform well for the other models. This is likely due to the fact that latent space of RNNG (which operates on sentences and parse trees) is much smaller than EntityNLM and KGLM (which operate on documents and coreference chains/knowledge graphs).

Best Practices From these results we recommend the following practices for future work utilizing importance sampling: (1) aggregate importance samples at the instance level, (2) condition on all available information when designing proposals, (3) try increased temperatures when generating samples from the proposal distribution, good temperatures can be identified using relatively few samples, and (4) utilize as many samples as possible. In addition, consider using beam marginalization in applications where strict upper bounds are needed.

5 Conclusion

We investigate the application of importance sampling to evaluating latent language models. Our contributions include: (1) showing that importance sampling produces stochastic upper bounds of perplexity, thereby justifying the use of such estimates for comparing language model performance, (2) a concise description of (sometimes unstated) common practices used in applying this technique, (3) a simple direct marginalization-based alternative to importance sampling, and (4) experimental results demonstrating the effect of sample size, sampling distribution, and granularity on estimates.

While this work helps clarify and validate existing results, we also observe that none of the estimates appear to converge even after drawing large numbers of samples. Thus, we encourage future research into obtaining tighter bounds on latent LM perplexity, possibly by using more powerful proposal distributions that consider entire documents as context, or by considering methods such as annealed importance sampling.

Acknowledgements

We would like to thank Alex Boyd for helpful discussions. This work was funded in part by Allen Institute of Artificial Intelligence, the NSF award #IIS-1817183, and in part by the DARPA MCS program under contract No. N660011924033 with the United States Office of Naval Research.

References

Sungjin Ahn, Heeyoul Choi, Tanel Pärnamaa, and Yoshua Bengio. 2016. A neural knowledge language model. *arXiv preprint arXiv:1608.00318*.

Elizabeth Clark, Yangfeng Ji, and Noah A. Smith. 2018. Neural text generation in stories using entity representations as context. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2250–2260, New Orleans, Louisiana. Association for Computational Linguistics.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A. Smith. 2016. Recurrent neural network grammars. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 199–209, San Diego, California. Association for Computational Linguistics.

John Geweke. 1989. Bayesian inference in econometric models using monte carlo integration. *Econometrica*, 57(6):1317–1339.

Hiroaki Hayashi, Zecong Hu, Chenyan Xiong, and Graham Neubig. 2020. Latent relation language models. In *Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI)*, New York, USA.

Yangfeng Ji, Chenhao Tan, Sebastian Martschat, Yejin Choi, and Noah A. Smith. 2017. Dynamic entity representations in neural language models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1830–1839, Copenhagen, Denmark. Association for Computational Linguistics.

Herman Kahn. 1950. Random sampling (monte carlo) techniques in neutron attenuation problems–i. *Nucleonics*, 6(5):27–passim.

Yoon Kim, Alexander Rush, Lei Yu, Adhiguna Kuncoro, Chris Dyer, and Gábor Melis. 2019. Unsupervised recurrent neural network grammars. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1105–1117, Minneapolis, Minnesota. Association for Computational Linguistics.

Robert Logan, Nelson F. Liu, Matthew E. Peters, Matt Gardner, and Sameer Singh. 2019. Barack's wife hillary: Using knowledge graphs for fact-aware language modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5962–5971, Florence, Italy. Association for Computational Linguistics.

Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank.

Art B. Owen. 2013. *Monte Carlo theory, methods and examples*.

Sameer Pradhan, Xiaoqiang Luo, Marta Recasens, Eduard Hovy, Vincent Ng, and Michael Strube. 2014. Scoring coreference partitions of predicted mentions: A reference implementation. In *Proceedings of the conference. Association for Computational Linguistics. Meeting*, volume 2014, page 30. NIH Public Access.

Aad W Van der Vaart. 2000. *Asymptotic statistics*, volume 3. Cambridge university press.