Loss	PPL
$\mathcal{L}^{(2)}_{text}(Zero)$	104.08
$\mathcal{L}^{(2)}_{text}\mathcal{L}^{(2)}(Zero^{(T)})$	102.56
$\mathcal{L}^{(2)}_{text}\mathcal{L}^{(2)}\mathcal{L}^{(103)}$	**93.95**

Table 3: Test perplexities of RNNLMs trained on different loss functions. Using n-gram probabilities from a larger corpus (WT-103) improves perplexities.

improve an RNNLM trained using WT-2. In our experiment, we use n-grams up to $n = 5$ with frequency greater than 50. We ignore n-gram containing <unk>, because the vocabulary sets are different. Table 3 shows the result. Since we do not use the same setting as in the original work, we cannot directly compare to that work – they use different optimization settings, more expensive n-gram loss, and Kneser-Ney bi-gram language model. However, we see that the proposed n-gram loss is beneficial when combined with the unigram loss. Importantly, unlike the approach in Noraset et al. (2018), our approach requires no sampling which makes it several times faster.

In addition, we present our preliminary result comparing training with the marginal probability of n-grams to training with the complete data. Given a limited budget of optimization steps, we ask whether training on n-grams is more valuable than training on the full corpus. To keep the results compatible, we use the vocabulary set of WikiText-2 and convert all OOV tokens in the training data of WikiText-103 to the "<unk>" token. Figure 1 shows the loss (average negative log-likelihood) of the validation data as the number of optimization steps increases.

We can see that training with the marginals does not perform as well as training with WikiText-103 training data, but outperforms the model trained only with WikiText-2 training data. This might be due to our choice of n-grams and optimization settings such as a number of n-grams per batch, weight of the n-gram loss, and the learning rate decay rate. We leave exploring these hyperparameters as an item of future work.

4 Conclusion

We investigated how to estimate marginal probabilities of n-grams from an RNNLM, when the preceding context is absent. We presented a simple method to train an RNNLM in which we occasionally reset the RNN's state and also maximize

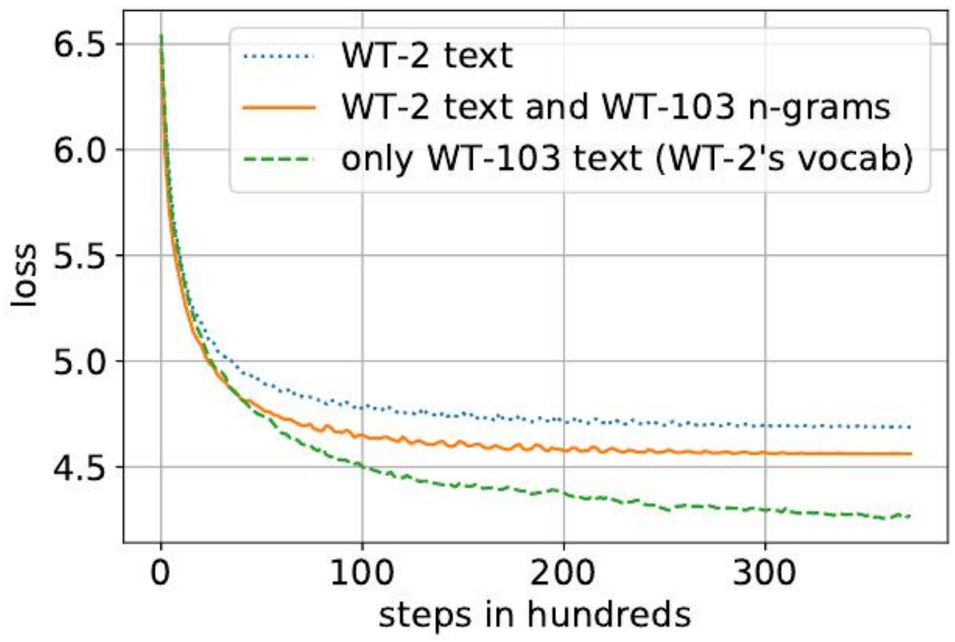

Figure 1: Loss in negative log-likelihood over steps in training. The loss computed using the valid data from WikiText-2 corpus. Training with n-grams from a larger corpus is helpful, but not as well as training with the running text from a larger corpus itself.

unigram likelihood along with the traditional objective. Our experiments showed that an RNNLM trained with our method outperformed other baselines on the marginal estimation task. Finally, we showed how to improve RNNLM perplexity by efficiently using additional n-gram probabilities from a larger corpus.

For future work, we would like to evaluate our approaches in more applications. For example, we can use the marginal statistics for information extraction, or to detect and remove abnormal phrases in text generation. In addition, we would like to continue improving the marginal estimation by experimenting with recent density estimation techniques such as NADE (Uria et al., 2016).

Acknowledgements

This work was supported in part by NSF grant IIS-1351029, the Tencent AI Lab Rhino-Bird Gift Fund, and Faculty of ICT, Mahidol University. We thank the reviewers for their valuable input.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations (ICLR2014)*. ArXiv e-Prints. 1409.0473.

Chandra Sekhar Bhagavatula, Thanapon Noraset, and Doug Downey. 2014. Textjoiner: On-demand information extraction with multi-pattern queries. In *4th Workshop on Automated Knowledge Base Construction at NIPS 2014*. AKBC.