Examination and Extension of Strategies for Improving Personalized Language Modeling via Interpolation

Liqun Shao*, **Sahitya Mantravadi***, **Tom Manzini***, **Alejandro Buendia***,
Manon Knoertzer*, **Soundar Srinivasan, and Chris Quirk**
Microsoft Corp.
{*lishao,samantr,thmanzin,albuendi,maknoert,sosrini,chrisq*}@*microsoft.com*

Abstract

In this paper, we detail novel strategies for interpolating personalized language models and methods to handle out-of-vocabulary (OOV) tokens to improve personalized language models. Using publicly available data from Reddit, we demonstrate improvements in offline metrics at the user level by interpolating a global LSTM-based authoring model with a user-personalized n-gram model. By optimizing this approach with a back-off to uniform OOV penalty and the interpolation coefficient, we observe that over 80% of users receive a lift in perplexity, with an average of 5.2% in perplexity lift per user. In doing this research we extend previous work in building NLIs and improve the robustness of metrics for downstream tasks.

1 Introduction

Natural language interfaces (NLIs) have become a ubiquitous part of modern life. Such interfaces are used to converse with personal assistants (e.g., Apple Siri, Amazon Alexa, Google Assistant, Microsoft Cortana), to search for and gather information (Google, Bing), and to interact with others on social media. One developing use case is to aid the user during composition by suggesting words, phrases, sentences, and even paragraphs that complete the user's thoughts (Radford et al., 2019).

Personalization of these interfaces is a natural step forward in a world where the vocabulary, grammar, and language can differ hugely user to user (Ishikawa, 2015; Rabinovich et al., 2018). Numerous works have described personalization in NLIs in audio rendering devices (Morse, 2008), digital assistants (Chen et al., 2014), telephone interfaces (Partovi et al., 2005), etc. We explore an approach for personalization of language models (LMs) for use in downstream NLIs on composition assistance, and replicate previous work to show that interpolating a global long short-term memory network (LSTM) model with user-personalized n-gram models provides per-user performance improvements when compared with only a global LSTM model (Chen et al., 2015, 2019). We extend that work by providing new strategies to interpolate the predictions of these two models. We evaluate these strategies on a publicly available set of Reddit user comments and show that our interpolation strategies deliver a 5.2% perplexity lift. Finally, we describe methods for handling the crucial edge case of out-of-vocabulary (OOV) tokens[1].

Specifically, the contributions of this work are:

1. We evaluate several approaches to handle OOV tokens, covering edge cases not discussed in the LM personalization literature.

2. We provide novel analysis and selection of interpolation coefficients for combining global models with user-personalized models.

3. We experimentally analyze trade-offs and evaluate our personalization mechanisms on public data, enabling replication by the research community.

2 Related Work

Language modeling is a critical component for many NLIs, and personalization is a natural direction to improve these interfaces.

Several published works have explored personalization of language models using historical search queries (Jaech and Ostendorf, 2018), features garnered from social graphs (Wen et al., 2012; Tseng

* Indicates equal contributions

[1] To the best knowledge of the authors, these edge cases are not clearly defined in the literature when combining two LMs trained on two different datasets.

Proceedings of the First Workshop on Natural Language Interfaces, pages 20–26
July 10, 2020.
https://doi.org/10.18653/v1/P17

et al., 2015; Lee et al., 2016), and transfer learning techniques (Yoon et al., 2017). Other work has explored using profile information (location, name, etc.) as additional features to condition trained models (Shokouhi, 2013; Jaech and Ostendorf, 2018). Specifically, in the NLI domain, Google Smart Compose (Chen et al., 2019) productized the approach described in (Chen et al., 2015) by using a linear interpolation of a general background model and a personalized n-gram model to personalize LM predictions in the email authoring setting. We view our work as a natural extension to this line of research because strategies that improve personalization at the language modeling level drive results at the user interface level.

3 Personalized Interpolation Model

The goal of text prediction is strongly aligned with language modeling. The task of language modeling is to predict which words come next, given a set of context words. In this paper, we explore using a combination of both large scale neural LMs and small scale personalized n-gram LMs. This combination has been studied in the literature (Chen et al., 2015) and has been found to be performant. We describe mechanisms for extending this previous work in this section. Once trained, we compute the perplexity of these models not by exponentiation of the cross entropy, but rather by explicitly predicting the probability of test sequences. In practice this model is to be used to rerank sentence completion sequences. As a result, it is impossible to ignore the observation of OOV tokens.

3.1 Personalized n-gram LMs

Back-off n-gram LMs (Kneser and Ney, 1995) have been widely adopted given their simplicity, and efficient parameter estimation and discounting algorithms further improve robustness (Chen et al., 2015). Compared with DNN-based models, n-gram LMs are computationally cheap to train, lightweight to store and query, and fit well even on small data—crucial benefits for personalization. Addressing the sharp distributions and sparse data issues in n-gram counts is critical. We rely on Modified Kneser-Ney smoothing (James, 2000), which is generally accepted as one of the most effective smoothing techniques.

3.2 Global LSTM

For large scale language modeling, neural network methods can produce dramatic improvements in predictive performance (Jozefowicz et al., 2016). Specifically, we use LSTM cells (Hochreiter and Schmidhuber, 1997), known for their ability to capture long distance context without vanishing gradients. By computing the softmax function on the output scores of the LSTM we can extract the LSTM's per-token approximation as language model probabilities.

3.3 Evaluation

We use perplexity (PP) to evaluate the performance of our LMs. PP is a measure of how well a probability model predicts a sample, i.e., how well an LM predicts the next word. This can be treated as a branching factor. Mathematically, PP is the exponentiation of the entropy of a probability distribution. Lower PP is indicative of a better LM. We define lift in perplexity (PP lift) as

$$\text{PP lift} = \frac{PP_{\text{global}} - PP_{\text{interpolated}}}{PP_{\text{global}}}, \tag{1}$$

where $PP_{\text{interpolated}}$ is the perplexity of the interpolated model and PP_{global} is the perplexity of the global LSTM model, which serves as the baseline. Higher PP lift is indicative of a better LM.

3.4 Interpolation Strategies

Past work (Chen et al., 2015) has described mechanisms for interpolating global models with personalized models for each user. Our experimentation mixes a global LSTM model with the personalized n-gram models detailed above[2].

The interpolation is a linear combination of the predicted token probabilities:

$$P = \alpha P_{\text{personal}} + (1 - \alpha) P_{\text{global}} \tag{2}$$

α indicates how much personalization is added to the global model. We explore constant values of α, either globally or for each user. We compute a set of oracle α values, the values of α per user that empirically minimize interpolated perplexity. We compare our strategies for tuning α to these oracle α values, which present the best possible performance on the given user data in Section 5.3. Intuitively, users whose comments have a high proportion of tokens outside the global vocabulary will

[2] We further detail the hyperparameters and training scheme of our LSTM and n-gram models in the appendices.

need more input from the global model than their own personalized model to accurately model their language habits. Thus, we also explore an inverse relationship between α and each user's OOV rate.

3.5 OOV Mitigation Strategies

When training on datasets with a large proportion of OOV tokens, low PP may not indicate a good model. Specifically, if the proportion of OOV tokens in the data is high, the model may assign too much mass to OOV tokens resulting in a model with a propensity to predict the OOV token. Such a model may have low PP, but only because it frequently predicts the commonly occurring OOV token. While this may be an effective model of the pure sequence of tokens, it does not align with downstream objectives present at the interface level which relies on a robust prediction of non-OOV tokens. Because of this disconnect between model and overall task objective, mitigation strategies must be implemented in order to adequately evaluate the performance of LMs in high OOV settings. We evaluate the following strategies to mitigate this behavior:

1. Do nothing, assigning OOV tokens their estimated probabilities;
2. Skip the OOV tokens, scoring only those items known in the training vocabulary; and
3. Back-off to a uniform OOV penalty, assigning a fixed probability ϕ to model the likelihood of selecting the OOV token[3].

When reporting our results we denote PP_{base} as PP observed when using strategy 1, PP_{skip} as PP observed when using strategy 2, and $\text{PP}_{\text{backoff}}$ as PP observed when using strategy 3.

4 Data

The data for our model comes from comments made by users on the Internet social media website Reddit[4]. Reddit is a rich source of natural-language data with high linguistic diversity due to posts about a variety of topics, informality of language, and sheer volume of data. As a linguistic resource, Reddit comments present in a heavily conversational and colloquial tone, and users frequently use slang and misspell words. Because of this there are a high number of unique tokens. As developers of a machine learning system, we seek to balance having a large vocabulary in order to capture the most data with having a small vocabulary in order to keep the model from overfitting. We construct our vocabulary by empirically selecting the n most common tokens observed by randomly selecting Reddit user comments. We then share this vocabulary, created from the global training set, in both the personalized and global models. This value of n must be tuned based on data. When choosing a size for vocabulary, there exists a tradeoff between performance and capturing varied language. Larger vocabularies adversely impact performance but may encapsulate more variability of language. For a given vocabulary size chosen from training data for the global LSTM, we plot the resulting OOV rates for users. As can be seen when comparing Figure 1 and Figure 2, very few gains in user-level OOV rates are seen when expanding the vocabulary size twenty-fold. Thus, we choose a vocabulary size of 50,000.

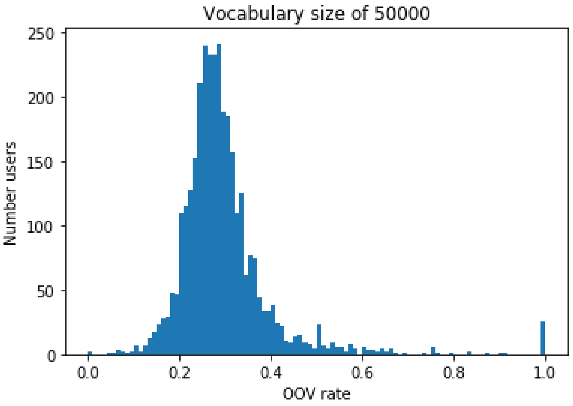

Figure 1: Histogram of OOV rates for 3265 users' training data with a vocabulary size of 50,000.

For the global LSTM, we split the global distribution of Reddit data into training sourced from 2016, validation sourced from 2017, and test sourced from 2018. We sampled such that 70% of users were reserved for training, 20% of users for validation, and 10% of users for test. We allot $100,000$ users for the test set and scale the number of users in the other sets accordingly. There are 10 billion total tokens in the training data, with 29 million unique tokens. 90% of unique tokens occur 6 or fewer times, and half of users have 20 or fewer comments per year with an average comment length of

[3] We consider ϕ to be a hyperparameter which must be tuned for each use case. In our experiments we assign ϕ to be $\frac{1}{V}$, where V is the vocabulary size.

[4] We retrieved copies of www.reddit.com user comments from https://pushshift.io/.

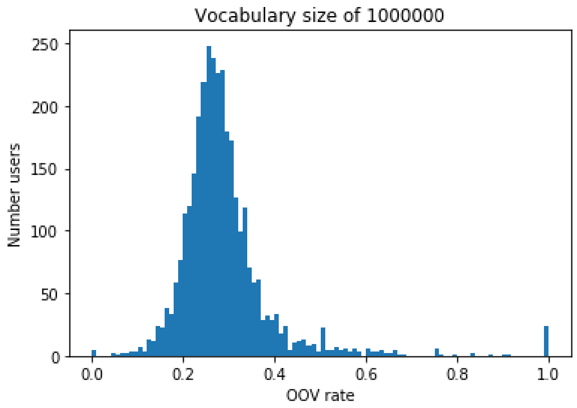

Figure 2: Histogram of OOV rates for 3265 users' training data with a vocabulary size of 1,000,000.

13 tokens. For the personalized n-grams, we selected all comment data from 3265 random Reddit users[6] who made at least one comment in each of 2016, 2017, and 2018. Then, for each user, we selected the data from 2016 as training data, the data from 2017 as validation data, and the data from 2018 as testing data.

5 Results

Here we discuss the results observed when evaluating the interpolated global LSTM and user-personalized n-gram model on users' comments using various OOV mitigation and α interpolation strategies.

5.1 OOV Mitigation Strategies

In our data used for personalization, 68% users have more than 25% OOV rate for validation data, and 65% users have more than 25% OOV rate for training data. This empirically causes large deviations between the different $\text{PP}_{\text{backoff}}$, PP_{skip}, and PP_{base}. We find that a personalized n-gram model can't handle OOV tokens very well in high OOV settings, because it assigns higher probabilities to OOV tokens than some of the tokens in the vocabulary. As discussed in Section 3.5 high OOV rates at the per-user level PP_{base} present a view of the results that is disconnected from downstream use in an NLI. At the same time, PP_{skip} presents the view most aligned with the downstream task because in an NLI the OOV token should never be shown. However, PP_{skip} comes with some mathematical baggage. Specifically, when all tokens are OOV, the PP_{skip} will be infinite. These two approaches represent the extremes of the strategies which could be used. We argue that $\text{PP}_{\text{backoff}}$ represents the best of both worlds.

Figure 3 shows that $\text{PP}_{\text{backoff}}$ provides measurements near the minima that are closely aligned with PP_{skip} while also being free of the mathematical and procedural issues associated with PP_{skip} and PP_{base}. We provide an example to further illustrate the above statement. Consider a high OOV rate comment such as "re-titled jaff ransomware only fivnin." with OOV tokens *re-titled, jaff, ransomware, fivnin*. Following encoding, the mode would see this sequence as "OOV OOV OOV only OOV". When measuring the probability of this sequence a model evaluated using PP_{base} would have lower perplexity because it has been trained to overweight the probability of OOV tokens as they occur more frequently than the tokens they represent. However, this sequence should have far lower probability, and thus higher perplexity, because the model is in fact failing to adequately model the true sequence. We argue that assigning a uniform value θ to OOV tokens will more accurately represent the performance of the model when presented with data with a high quantity of OOV tokens.

Because we believe that $\text{PP}_{\text{backoff}}$ presents the most accurate picture of model performance, we have chosen to present our results in Section 5.2 and 5.3 using $\text{PP}_{\text{backoff}}$.

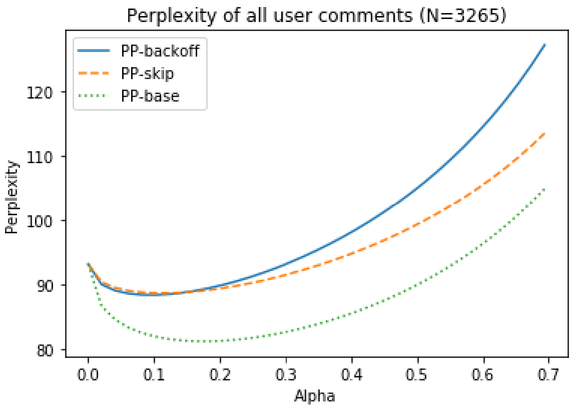

Figure 3: Average of interpolated PP for all users for varied values of $\alpha \leq 0.7$ for each method of approaching OOV tokens.

5.2 Analysis of Personalization

We next present an interesting dichotomy in Figure 4 not previously discussed in the personalization literature. In the constant α for all users setting we can optimize to either minimize the overall $\text{PP}_{\text{backoff}}$ for all users or to maximize the average $\text{PP}_{\text{backoff}}$ lift across users. These two objectives result in different constant α[5]. Specifically, minimizing $\mathrm{PP}_{\mathrm{backoff}}$ over users yields $\alpha = 0.105$, providing an improvement for 67.3% of users and an average $\mathrm{PP}_{\mathrm{backoff}}$ lift of 2.5%. Maximizing the average $\mathrm{PP}_{\mathrm{backoff}}$ lift per user yields $\alpha = 0.041$, providing an improvement for 74.2% of users and an average $\mathrm{PP}_{\mathrm{backoff}}$ lift of 2.7%[6].

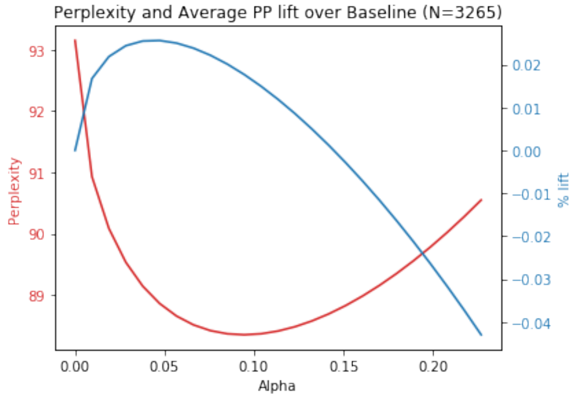

Figure 4: $\mathrm{PP}_{\mathrm{backoff}}$ and average $\mathrm{PP}_{\mathrm{backoff}}$ lift over baseline for various values of $\alpha < 0.22$.

5.3 Constant and Personalized Interpolation Coefficient α Optimization

When searching for a constant value for α for all users, $\alpha = 0.105$ achieves the minimum mean interpolated $\mathrm{PP}_{\mathrm{backoff}}$, with an average $\mathrm{PP}_{\mathrm{backoff}}$ lift of 2.5%.

Next, we personalize the value of α for each user. We first produce a set of oracles[6] as described in Section 3.4. With this set of oracle values of α, the average $\mathrm{PP}_{\mathrm{backoff}}$ lift is 6.1% with the best average $\mathrm{PP}_{\mathrm{backoff}}$ achievable in this context. While it is possible to compute the oracle values for each user in a production setting, this may not be tractable when user counts are high and there exist latency constraints.

Thus, we try an inverse linear relationship: $\alpha = k \cdot (1 - \mathrm{OOV\ rate})$. To illustrate the effect of this relationship, we perform this optimization 10 times, using a different random subset of users each time to optimize k and then evaluate on the rest of the users. On average, we observe a $\mathrm{PP}_{\mathrm{backoff}}$ lift of 5.2%, and 80.1% of users achieve an improvement in $\mathrm{PP}_{\mathrm{backoff}}$. In Figure 5 we see that a heuristic approach of lower complexity achieves near-oracle performance, with the distribution of $\mathrm{PP}_{\mathrm{backoff}}$ for this method closely matching the oracle distribution of $\mathrm{PP}_{\mathrm{backoff}}$. We also find that this method of α personalization yields lower $\mathrm{PP}_{\mathrm{backoff}}$ for more users than using a constant value for α.

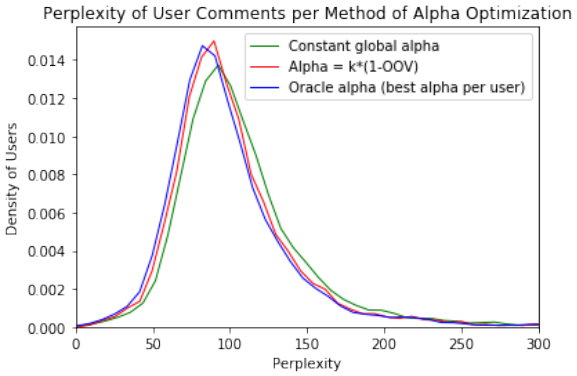

Figure 5: Distribution of interpolated $\mathrm{PP}_{\mathrm{backoff}}$ for users using each method of α optimization. The values for $\alpha = k \cdot (1 - \mathrm{OOV\ rate})$ are averaged over 10 random selections.

6 Conclusion & Future Work

In this paper we presented new strategies for interpolating personalized LMs, discussed strategies for handling OOV tokens to give better vision into model performance, and evaluated these strategies on public data allowing the research community to build upon these results. Furthermore, two directions could be worth exploring: Investigate on when personalization is useful at a user level to better interpret the results; Research on user-specific vocabularies for personalized models instead of using a shared vocabulary for both the personalized and global background models.

As NLIs move closer to the user, personalization mechanisms will need to become more robust. We believe the results we have presented form a natural step in building that robustness.

Acknowledgments

We would like to thank the Microsoft Search, Assistant and Intelligence team, and in particular Geisler Antony, Kalyan Ayloo, Mikhail Kulikov, Vipul Agarwal, Anton Amirov, Nick Farn and Kunho Kim, for their invaluable help and support in this research. We also thank T. J. Hazen, Vijay Ramani, and the anonymous reviewers for their insightful feedback and suggestions.

[5] There may be other trade-offs to examine.

[6] Further details are included in the appendices.

References

Lik Harry Chen, Adam John Cheyer, Didier Rene Guzzoni, and Thomas Robert Gruber. 2014. Personalized vocabulary for digital assistant. US Patent 8,903,716.

Mia Xu Chen, Benjamin N Lee, Gagan Bansal, Yuan Cao, Shuyuan Zhang, Justin Lu, Jackie Tsay, Yinan Wang, Andrew M Dai, Zhifeng Chen, et al. 2019. Gmail smart compose: Real-time assisted writing. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2287–2295.

Xie Chen, Xunying Liu, Mark JF Gales, and Philip C Woodland. 2015. Investigation of back-off based interpolation between recurrent neural network and n-gram language models. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 181–186. IEEE.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Yuka Ishikawa. 2015. Gender differences in vocabulary use in essay writing by university students. *Procedia-Social and Behavioral Sciences*, 192:593–600.

Aaron Jaech and Mari Ostendorf. 2018. Personalized language model for query auto-completion. *arXiv preprint arXiv:1804.09661*.

Frankie James. 2000. Modified kneser-ney smoothing of n-gram models. *Research Institute for Advanced Computer Science, Tech. Rep. 00.07*.

Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. 2016. Exploring the limits of language modeling.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *1995 International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 181–184. IEEE.

Hung-Yi Lee, Bo-Hsiang Tseng, Tsung-Hsien Wen, and Yu Tsao. 2016. Personalizing recurrent-neural-network-based language model by social network. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(3):519–530.

Lee Morse. 2008. System and method for personalizing the user interface of audio rendering devices. US Patent App. 11/779,256.

Hadi Partovi, Roderick Steven Brathwaite, Angus Macdonald Davis, Michael S McCue, Brandon William Porter, John Giannandrea, Eckart Walther, Anthony Accardi, and Zhe Li. 2005. Method and apparatus for content personalization over a telephone interface with adaptive personalization. US Patent 6,842,767.

Ella Rabinovich, Yulia Tsvetkov, and Shuly Wintner. 2018. Native language cognate effects on second language lexical choice. *Transactions of the Association for Computational Linguistics*, 6:329–342.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Milad Shokouhi. 2013. Learning to personalize query auto-completion. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 103–112.

Bo-Hsiang Tseng, Hung-yi Lee, and Lin-Shan Lee. 2015. Personalizing universal recurrent neural network language model with user characteristic features by social network crowdsourcing. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 84–91. IEEE.

Tsung-Hsien Wen, Hung-Yi Lee, Tai-Yuan Chen, and Lin-Shan Lee. 2012. Personalized language modeling by crowd sourcing with social network data for voice access of cloud applications. In *2012 IEEE Spoken Language Technology Workshop (SLT)*, pages 188–193. IEEE.

Seunghyun Yoon, Hyeongu Yun, Yuna Kim, Gyu-tae Park, and Kyomin Jung. 2017. Efficient transfer learning schemes for personalized language modeling using recurrent neural network. In *Workshops at the Thirty-First AAAI Conference on Artificial Intelligence*.

A Appendices

A.1 Hyperparameters and Model Training

The global LSTM model trained token embeddings of size 300, and had hidden unit layers of size 256 and 128, an output projection of dimension 100, and a vocabulary of 50,000 tokens. It was trained with dropout using the Adam optimizer, and we parallel-trained our global LSTM on an Azure[7] Standard_NC24s_v2 machine which includes 24 vCPUs and 4 NVIDIA Tesla P100 GPUs.

The personalized n-gram models were 3-gram modified Kneser-Ney smoothed models with discounting values of 0.5 (1-grams), 1 (2-grams), and 1.5 (3-grams).

A.2 User Analysis Plots

The average size of the user-personalized corpus is around 140 comments, while the median size is 23 comments. The average comment length for each user is around 14 tokens.

[7] www.azure.com

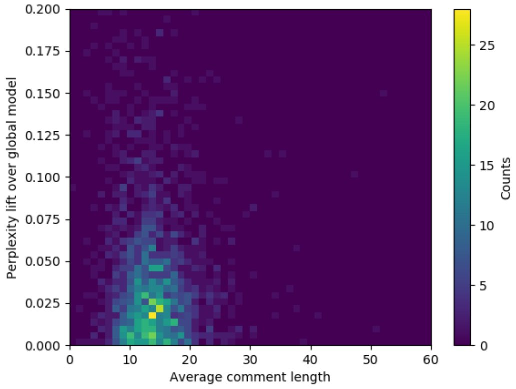

Figure 6: Histogram of PP lift over global model vs. average comment length ($\alpha = 0.105$).

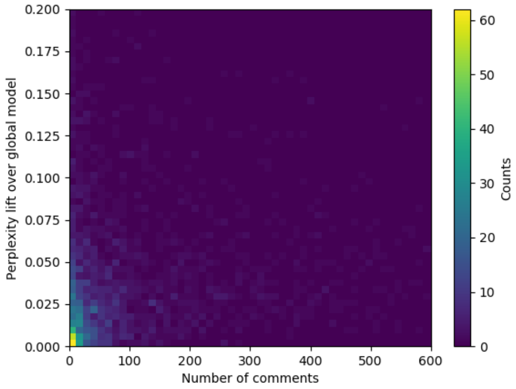

Figure 7: Histogram of PP lift over global model vs. number of comments ($\alpha = 0.105$).

By analyzing the results with the lowest interpolated PP$_{\text{backoff}}$ ($\alpha = 0.105$ for all users), we make two observations: users with average comment length less than around 30 tokens don't get much benefit from personalization, and users with less than around 100 comments don't get much benefit from personalization.

A.3 Oracle α Distribution

Figure 8 shows the distribution of the empirically-computed "oracle" values for α.

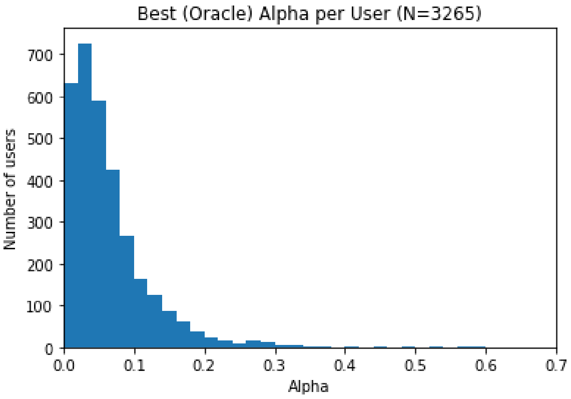

Figure 8: Distribution of oracle values of α per user.