Efficient Prompt Caching for Large Language Model Inference via Embedding Similarity

Hanlin Zhu
EECS, UC Berkeley
hanlinzhu@berkeley.edu

Banghua Zhu
EECS, UC Berkeley
banghua@berkeley.edu

Jiantao Jiao
EECS and Statistics, UC Berkeley
jiantao@berkeley.edu

Abstract

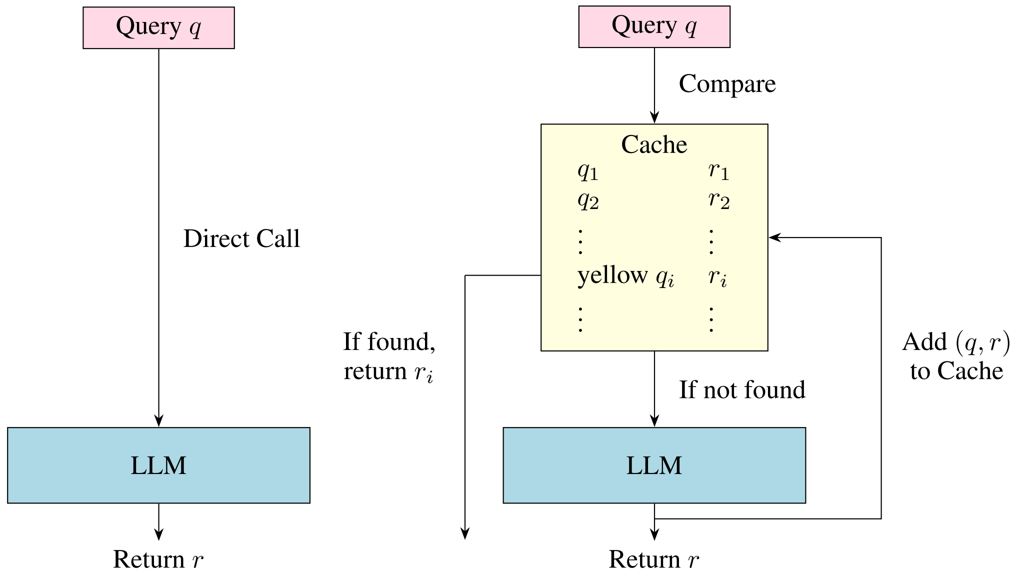
Large language models (LLMs) have achieved huge success in numerous natural language process (NLP) tasks. However, it faces the challenge of significant resource consumption during inference. In this paper, we aim to improve the inference efficiency of LLMs by prompt caching, i.e., if the current prompt can be answered by the same response of a previous prompt, one can directly utilize that response without calling the LLM. Specifically, we focus on the prediction accuracy of prompt caching for single-round question-answering tasks via embedding similarity. The existing embeddings of prompts mostly focus on whether two prompts are semantically similar, which is not necessarily equivalent to whether the same response can answer them. Therefore, we propose a distillation-based method to fine-tune the existing embeddings for better caching prediction. Theoretically, we provide finite-sample guarantees for the convergence of our method under different types of loss functions. Empirically, we construct a dataset based on Kwiatkowski et al. [2019] and fine-tune the embedding from Wang et al. [2022], which improves the AUC of caching prediction from 0.85 to 0.92 within 10 minutes of training. The resulting embedding model improves the throughput over the initial embedding model.

1 Introductions

The recent development of large language models (LLMs) and foundation models has notably enhanced the potential of AI systems [Ziegler et al., 2019, Wei et al., 2022, Chowdhery et al., 2022, Ouyang et al., 2022, Bubeck et al., 2023, Nori et al., 2023, OpenAI, 2023, Beeching et al., 2023, Anil et al., 2023] However, due to the large scale of those models, it causes significant resource consumptions not only during the training process, but also in the inference stage [Sharir et al., 2020, Patterson et al., 2021, Bommasani et al., 2022]. Moreover, the latency of LLMs during inference is not negligible since the model only generates one token at a time due to its auto-regressive nature, which makes it unfavorable to be applied to systems desiring high throughput, such as search engines [Zhu et al., 2023]. Therefore, it would be appealing to reduce the resource consumption and latency without degrading the performance of LLMs.

A natural idea to reduce resource consumption and latency is to reduce the number of calls to LLMs, which can be implemented by caching, a technique that has a long history of being studied and applied to important areas such as computer architecture and web retrieval [Smith, 1982, Wang, 1999, Kumar and Singh, 2016]. Zhu et al. [2023] studies prompt (or query) caching for LLMs, i.e., some of the previous prompt-response pairs are stored in a cache with limited size, and whenever a

Machine Learning for Systems Workshop at 37th NeurIPS Conference, 2023, New Orleans, LA, USA.



(a) Calling LLMs without cache. (b) Calling LLMs with cache.

Figure 1: The procedure of calling LLMs with or without cache. When a cache is available, one can store some of the previous prompt-response pairs in the cache, and for a new prompt, one can search in the cache whether a prompt has the same semantic meaning as the current prompt. If there is a hit, one can directly reuse the response of the previous prompt without calling LLMs.

new prompt arrives, one can search in the cache whether a prompt has the same semantic meaning as the current prompt, and can directly reuse the response of the previous prompt without calling LLMs if there is a hit (see Figure 1 for a figurative illustration).

Zhu et al. [2023] focuses on caching algorithm design and directly assumes a semantic search oracle. Although previous literature studies semantic search or embedding-based methods [Bast et al., 2016, Chang et al., 2020, Kamalloo et al., 2023], which could serve as solutions to the caching hit problem [zilliztech, 2023] [1], it is challenging to obtain a good embedding that can accurately represent the semantic meaning of a prompt. Moreover, a semantically similar prompt pair cannot necessarily be answered by the same response, which implies that we need a different vector embedding specifically for the caching hitting problem that can be used to search a similar prompt more efficiently and better predict the probability that a pair of prompts can be answered by the same response.

In this paper, we aim to learn a good vector embedding such that the similarity of embeddings of a prompt pair could encode the information of whether the pair of prompts can be answered by the same response, i.e., to better predict the probability that they can be answered by the same response. We propose a distillation-based method, which aims to learn the ground-truth probability of whether a prompt pair can be answered by the response via cosine similarity of the embeddings of the prompt pair, to fine-tune an existing semantic vector embedding from Wang et al. [2022]. Theoretically, we provide finite sample guarantees for the learning error under mild assumptions using cross entropy and squared log difference errors, respectively (Appendix B). Empirically, we construct a dataset based on Kwiatkowski et al. [2019] and fine-tune the embedding from Wang et al. [2022], which improves the AUC of caching prediction from 0.85 to 0.92 within 10 minutes of training using cross entropy error (Section 3). We also show that the fine-tuned embedding improves the throughput over the initial embedding without fine-tuning (Section 3).

[1] i.e., searching whether there exists a prompt in the cache such that the current prompt can be answered by the same response.

2 Preliminaries

We introduce in this section some basic notations, definitions, and assumptions. Let $\mathcal{Q}$ denote the set of all possible prompts (queries). For any prompt pair $(q_1, q_2) \in \mathcal{Q} \times \mathcal{Q}$, we denote the ground-truth probability that (q_1, q_2) can be answered by the same response by $P^\star(q_1 = q_2)$. Assume there exists an underlying distribution μ of prompt pairs (q_1, q_2). Note that we do not have direct access to the μ and instead are given a dataset $\mathcal{D} = \{(q_{i,1}, q_{i,2}, p_i)\}_{i=1}^N$, where $(q_{i,1}, q_{i,2}) \overset{\text{i.i.d.}}{\sim} \mu$ and $p_i \in [0, 1]$ with $p_i \sim \mathcal{P}(\cdot|q_{i,1}, q_{i,2})$ [2] and $\mathbb{E}[p_i|q_{i,1}, q_{i,2}] = P^\star(q_{i,1} = q_{i,2})$. Below, we define the vector embedding of prompts and define probability via embedding similarity.

Definition 2.1 (Embedding of prompts)**.** *For any prompt q, let $v_\theta(q) \in \mathbb{R}^d$ denote its vector embedding where v can be viewed as the mapping of prompts to a specific layer of a language model, and $\theta \in \Theta$ is the parameters of that model.*

Definition 2.2 (Probability via embedding similarity)**.** *For any two prompts q_1, q_2, we denote the induced probability via embedding similarity that q_1, q_2 can be answered by the same response by*

$$P_{\theta,\lambda,c}(q_1 = q_2) \triangleq \sigma(\mathtt{sim}(v_\theta(q_1), v_\theta(q_2))/\lambda - c),$$

where $\mathtt{sim}(\cdot,\cdot)$ denotes the cosine similarity, i.e., $\mathtt{sim}(x, y) = \frac{\langle x,y\rangle}{\|x\|\cdot\|y\|}$ for two vectors $x, y \in \mathbb{R}^d$, $\sigma(x) = \frac{1}{1+\exp(-x)}$ for $x \in \mathbb{R}$, and $\lambda \in \Lambda \subset \mathbb{R}_+, c \in \mathcal{C} \subset \mathbb{R}$ are two real-valued parameters.

3 Experiments

In this section, we show experimental results that our distillation-based method can improve the accuracy of caching prediction and improve the throughput using caching.

Construction of the dataset. We first extract all prompts (queries) from the natural_questions dataset [Kwiatkowski et al., 2019] and compute a vector embedding for each prompt using the last layer of the intfloat/e5-large-v2 model [Wang et al., 2022]. After deleting repeated prompts, for each prompt, we search the five nearest neighbors using FAISS [Johnson et al., 2019]. We sample 1999 prompts uniformly at random, and for each prompt, we choose the farthest three prompts [3] among the five nearest neighbors to form three prompt pairs. Therefore, we get 5997 prompt pairs in total, and we use GPT-4 [OpenAI, 2023] to label whether each prompt pair can be answered by the same response (0 or 1). We split the dataset into a training set of size 5497 and a validation set of size 500.

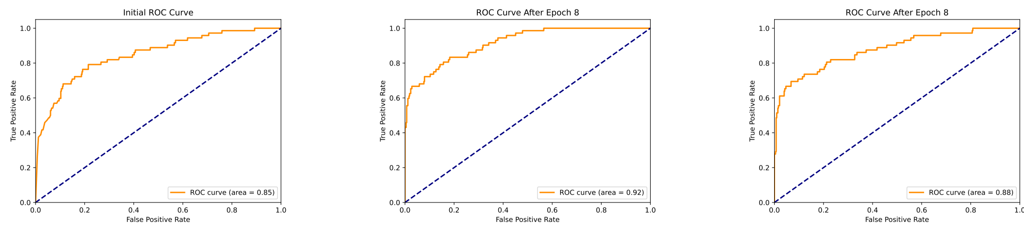
Fine-tuning of embeddings. We fine-tune using cross-entropy loss or squared log difference loss from the embedding of Wang et al. [2022]. Slightly different from the theoretical version, we view λ and c as hyper-parameters and set them to $\lambda = 0.01, c = 80$. For squared log difference loss, we clip the label to $[10^{-10}, 1]$ to avoid calculating $\log 0$, which is not well-defined. We set the learning rate to be 10^{-5} and present the ROC curve as well as AUC on the validation set of the initial embedding and embeddings fine-tuned for eight epochs using two loss functions respectively in Figure 2.

As Figure 2 shows, fine-tuning on our constructed dataset using either loss function can help to improve the AUC, while the cross-entropy loss function shows a better performance than the squared log difference loss function, which is consistent with our theoretical results in Appendix B.

Simulation of the prompt streaming with caching. We also conduct a simulation to validate that the caching through embedding similarity with our fine-tuned embedding model can improve over the initial embedding model (i.e., the intfloat/e5-large-v2 model [Wang et al., 2022] in our case). We first create the prompt streaming dataset, which contains 500 prompts. Specifically, the test set of the above experiments contains 500 pairs of prompts, and we randomly discard 250 pairs with label 0 and use the remaining 250 pairs (500 prompts) to construct the simulated prompt streaming dataset in random order. Since it is not the main focus of this paper to study the tradeoff between the size of the caching and the throughput, and we have a moderate test set, we assume for simplicity that the cache has an unlimited size or size larger than 500.

[2] $\mathcal{P}(\cdot|q_{i,1}, q_{i,2}) \in \Delta([0, 1])$ can be any distribution on $[0, 1]$.

[3] We choose the farthest three to construct a more "difficult" dataset. Note that for each prompt, the farthest three might still be close to it but cannot be answered by the same response.



(a) ROC before fine-tuning (AUC = 0.85)

(b) ROC after Epoch 8 using cross-entropy loss (AUC = 0.92)

(c) ROC after Epoch 8 using squared loss (AUC = 0.88)

Figure 2: Comparison of ROC curves. Both loss functions can help to improve the AUC, while the cross-entropy loss function shows a better performance than the squared log difference loss function, which is consistent with our theoretical results in Appendix B.

The caching is initialized to be empty. We maintain two counters: `nBadResponse` and `nLLMQuery`. At each time, a prompt q from the dataset arrives, and its embedding $v(q) \in \mathbb{R}^d$ (in our experiments, $d = 1024$) is calculated. We first find the nearest neighbor in that cache which is a tuple $(q_0, v(q_0), r_0)$, where q_0 is a prompt, r_0 is the response, and $v(q_0)$ is the embedding. The distance is measured by the angular distance (or ℓ_2 distance between the normalized vectors). If $\sigma(\texttt{sim}(v(q_1), v(q_2))/\lambda - c) < \tau$ where $\tau \in [0, 1]$ is a threshold, we view it as a hit and use r_0 as the response of q; otherwise, we view it as a miss, and directly query the LLM model to get the response r. In the first case, we will query GPT4 whether r_0 is a good response to q, and if not, we set $\texttt{nBadResponse} \leftarrow \texttt{nBadResponse} + 1$. In the second case, we will add the new tuple $(q, v(q), r)$ to the cache and set $\texttt{nLLMQuery} \leftarrow \texttt{nLLMQuery} + 1$.

Finally, we calculate the throughput as

$$\texttt{thp} = \frac{N - \texttt{nBadResponse}}{\texttt{nLLMQuery} + \texttt{nBadResponse}},$$

where $N = 500$ is the size of the prompt streaming dataset. We fix $\lambda = 0.01, c = 80$ as in the first experiment. We choose v to be one of the three models: intfloat/e5-large-v2 without finetuning, intfloat/e5-large-v2 finetuned using BCE loss for seven epochs, intfloat/e5-large-v2 finetuned using squared log difference (SLD) loss for six epochs. We also choose different threshold τ. The result is presented in Table 1, which shows that after fine-tuning using either loss, the throughput is improved over the initial model without fine-tuning.

threshold	0.6	0.65	0.7	0.75	0.8	0.85	0.9	0.95
Initial model	0.002	0.014	0.038	0.211	0.605	0.897	**1.104**	1.081
BCE model	0.75	0.899	1.017	1.082	1.11	**1.13**	1.102	1.066
SLD model	0.99	1.091	1.113	**1.121**	1.12	1.088	1.067	1.044

Table 1: Comparison of throughput of different embedding models.

4 Conclusions

In this paper, we study efficient prompt caching for LLMs by modeling the ground-truth probability of whether a prompt pair can be answered by the same response via embedding similarity, and fine-tuning existing semantic embeddings on our newly constructed dataset. We provide both theoretical guarantee and empirical evidence that our proposed distillation-based method can improve the accuracy of caching prediction and throuput. Interesting future directions include improving the $O(1/N^{1/4})$ rate and simulating the caching procedure using our fine-tuned embeddings.

References

Rohan Anil, Andrew M. Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, Eric Chu, Jonathan H. Clark, Laurent El Shafey, Yanping Huang, Kathy Meier-Hellstern, Gaurav Mishra, Erica Moreira, Mark Omernick, Kevin Robinson, Sebastian Ruder, Yi Tay, Kefan Xiao, Yuanzhong Xu, Yujing Zhang, Gustavo Hernandez Abrego, Junwhan Ahn, Jacob Austin, Paul Barham, Jan Botha, James Bradbury, Siddhartha Brahma, Kevin Brooks, Michele Catasta, Yong Cheng, Colin Cherry, Christopher A. Choquette-Choo, Aakanksha Chowdhery, Clément Crepy, Shachi Dave, Mostafa Dehghani, Sunipa Dev, Jacob Devlin, Mark Díaz, Nan Du, Ethan Dyer, Vlad Feinberg, Fangxiaoyu Feng, Vlad Fienber, Markus Freitag, Xavier Garcia, Sebastian Gehrmann, Lucas Gonzalez, Guy Gur-Ari, Steven Hand, Hadi Hashemi, Le Hou, Joshua Howland, Andrea Hu, Jeffrey Hui, Jeremy Hurwitz, Michael Isard, Abe Ittycheriah, Matthew Jagielski, Wenhao Jia, Kathleen Kenealy, Maxim Krikun, Sneha Kudugunta, Chang Lan, Katherine Lee, Benjamin Lee, Eric Li, Music Li, Wei Li, YaGuang Li, Jian Li, Hyeontaek Lim, Hanzhao Lin, Zhongtao Liu, Frederick Liu, Marcello Maggioni, Aroma Mahendru, Joshua Maynez, Vedant Misra, Maysam Moussalem, Zachary Nado, John Nham, Eric Ni, Andrew Nystrom, Alicia Parrish, Marie Pellat, Martin Polacek, Alex Polozov, Reiner Pope, Siyuan Qiao, Emily Reif, Bryan Richter, Parker Riley, Alex Castro Ros, Aurko Roy, Brennan Saeta, Rajkumar Samuel, Renee Shelby, Ambrose Slone, Daniel Smilkov, David R. So, Daniel Sohn, Simon Tokumine, Dasha Valter, Vijay Vasudevan, Kiran Vodrahalli, Xuezhi Wang, Pidong Wang, Zirui Wang, Tao Wang, John Wieting, Yuhuai Wu, Kelvin Xu, Yunhan Xu, Linting Xue, Pengcheng Yin, Jiahui Yu, Qiao Zhang, Steven Zheng, Ce Zheng, Weikang Zhou, Denny Zhou, Slav Petrov, and Yonghui Wu. Palm 2 technical report, 2023.

Hannah Bast, Björn Buchhold, Elmar Haussmann, et al. Semantic search on text and knowledge bases. *Foundations and Trends® in Information Retrieval*, 10(2-3):119–271, 2016.

Edward Beeching, Younes Belkada, Kashif Rasul, Lewis Tunstall, Leandro von Werra, Nazneen Rajani, and Nathan Lambert. Stackllama: An rl fine-tuned llama model for stack exchange question and answering, 2023. URL `https://huggingface.co/blog/stackllama`.

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Ben Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, Julian Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Rob Reich, Hongyu Ren, Frieda Rong, Yusuf Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishnan Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. On the opportunities and risks of foundation models, 2022.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR, 2022.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

Archana Bura, Desik Rengarajan, Dileep Kalathil, Srinivas Shakkottai, and Jean-Francois Chamberland. Learning to cache and caching to learn: Regret analysis of caching algorithms. *IEEE/ACM Transactions on Networking*, 30(1):18–31, 2021.

Wei-Cheng Chang, Felix X Yu, Yin-Wen Chang, Yiming Yang, and Sanjiv Kumar. Pre-training tasks for embedding-based large-scale retrieval. *arXiv preprint arXiv:2002.03932*, 2020.

Zheng Chang, Lei Lei, Zhenyu Zhou, Shiwen Mao, and Tapani Ristaniemi. Learn to cache: Machine learning for network edge caching in the big data era. *IEEE Wireless Communications*, 25(3): 28–35, 2018.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

Fathima Zarin Faizal, Priya Singh, Nikhil Karamchandani, and Sharayu Moharir. Regret-optimal online caching for adversarial and stochastic arrivals. In *EAI International Conference on Performance Evaluation Methodologies and Tools*, pages 147–163. Springer, 2022.

Edouard Grave, Armand Joulin, and Nicolas Usunier. Improving neural language models with a continuous cache. *arXiv preprint arXiv:1612.04426*, 2016.

Edouard Grave, Moustapha M Cisse, and Armand Joulin. Unbounded cache model for online language modeling with open vocabulary. *Advances in neural information processing systems*, 30, 2017.

Ying He, Zheng Zhang, F Richard Yu, Nan Zhao, Hongxi Yin, Victor CM Leung, and Yanhua Zhang. Deep-reinforcement-learning-based optimization for cache-enabled opportunistic interference alignment wireless networks. *IEEE Transactions on Vehicular Technology*, 66(11):10433–10445, 2017.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Few-shot learning with retrieval augmented language models. *arXiv preprint arXiv:2208.03299*, 2022.

Wei Jiang, Gang Feng, Shuang Qin, Tak Shing Peter Yum, and Guohong Cao. Multi-agent reinforcement learning for efficient content caching in mobile d2d networks. *IEEE Transactions on Wireless Communications*, 18(3):1610–1622, 2019.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2019.

Ehsan Kamalloo, Xinyu Zhang, Odunayo Ogundepo, Nandan Thakur, David Alfonso-Hermelo, Mehdi Rezagholizadeh, and Jimmy Lin. Evaluating embedding apis for information retrieval. *arXiv preprint arXiv:2305.06300*, 2023.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models. *arXiv preprint arXiv:1911.00172*, 2019.

Swadhesh Kumar and PK Singh. An overview of modern cache memory and performance analysis of replacement policies. In *2016 IEEE International Conference on Engineering and Technology (ICETECH)*, pages 210–214. IEEE, 2016.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*, 2019.

Donghee Lee, Jongmoo Choi, Jong-Hun Kim, Sam H Noh, Sang Lyul Min, Yookun Cho, and Chong Sang Kim. Lrfu: A spectrum of policies that subsumes the least recently used and least frequently used policies. *IEEE transactions on Computers*, 50(12):1352–1361, 2001.

Sewon Min, Weijia Shi, Mike Lewis, Xilun Chen, Wen-tau Yih, Hannaneh Hajishirzi, and Luke Zettlemoyer. Nonparametric masked language modeling. *arXiv preprint arXiv:2212.01349*, 2022.

Samrat Mukhopadhyay and Abhishek Sinha. Online caching with optimal switching regret. In *2021 IEEE International Symposium on Information Theory (ISIT)*, pages 1546–1551. IEEE, 2021.

Harsha Nori, Nicholas King, Scott Mayer McKinney, Dean Carignan, and Eric Horvitz. Capabilities of gpt-4 on medical challenge problems. *arXiv preprint arXiv:2303.13375*, 2023.

OpenAI. Gpt-4 technical report, 2023.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35: 27730–27744, 2022.

David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. Carbon emissions and large neural network training. *arXiv preprint arXiv:2104.10350*, 2021.

Or Sharir, Barak Peleg, and Yoav Shoham. The cost of training nlp models: A concise overview. *arXiv preprint arXiv:2004.08900*, 2020.

Junaid Shuja, Kashif Bilal, Waleed Alasmary, Hassan Sinky, and Eisa Alanazi. Applying machine learning techniques for caching in next-generation edge networks: A comprehensive survey. *Journal of Network and Computer Applications*, 181:103005, 2021.

Alan Jay Smith. Cache memories. *ACM Computing Surveys (CSUR)*, 14(3):473–530, 1982.

William Stallings. *Operating systems: internals and design principles*. Prentice Hall Press, 2011.

Jia Wang. A survey of web caching schemes for the internet. *ACM SIGCOMM Computer Communication Review*, 29(5):36–46, 1999.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2022.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

Wenhao Zhan, Baihe Huang, Audrey Huang, Nan Jiang, and Jason Lee. Offline reinforcement learning with realizability and single-policy concentrability. In *Conference on Learning Theory*, pages 2730–2775. PMLR, 2022.

Zexuan Zhong, Tao Lei, and Danqi Chen. Training language models with memory augmentation. *arXiv preprint arXiv:2205.12674*, 2022.

Banghua Zhu, Ying Sheng, Lianmin Zheng, Clark Barrett, Michael I Jordan, and Jiantao Jiao. On optimal caching and model multiplexing for large model inference. *arXiv preprint arXiv:2306.02003*, 2023.

Hanlin Zhu and Amy Zhang. Provably efficient offline goal-conditioned reinforcement learning with general function approximation and single-policy concentrability. *arXiv preprint arXiv:2302.03770*, 2023.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

zilliztech. Gptcache: Semantic cache for llms. fully integrated with langchain and llama_index., 2023. URL `https://github.com/zilliztech/GPTCache`.

A Related Works

Caching. Caching algorithms are important to computer architecture and systems and have long been explored [Lee et al., 2001, Stallings, 2011, Bura et al., 2021]. In recent years, caching has also been applied to online learning analysis and machine learning advice [He et al., 2017, Chang et al., 2018, Jiang et al., 2019, Shuja et al., 2021, Mukhopadhyay and Sinha, 2021, Faizal et al., 2022]. Zhu et al. [2023] is the most related work and studies optimal caching algorithm for prompt in both online and offline learning settings. Instead of studying caching policy, we aim to study how to efficiently search and accurately predict whether there is a caching hit.

Retrieval-based LLMs. A line of work studies augmenting a language model by retrieval-based method [Grave et al., 2016, 2017, Khandelwal et al., 2019, Borgeaud et al., 2022, Izacard et al., 2022, Zhong et al., 2022, Min et al., 2022]. For example, the kNN-LM model [Khandelwal et al., 2019] interpolates a distribution obtained by the vector embedding of k nearest neighbors with the distribution of language models. Our formulation of probability via embedding similarity is inspired by these works.

B Theoretical Results

In this section, we provide finite sample guarantees for the convergence of the learning error. We compare two different loss functions, i.e., binary cross entropy loss (Appendix B.1) and squared log difference loss (Appendix B.2).

First, we make the following assumptions for theoretical analysis.

Assumption B.1 (Realizability). *Assume there exists $\theta^\star \in \Theta, \lambda^\star \in \Lambda, c^\star \in \mathcal{C}$, s.t. for any prompt pairs $(q_1, q_2) \in \mathcal{Q} \times \mathcal{Q}$, it holds that*

$$P_{\theta^\star,\lambda^\star,c^\star}(q_1 = q_2) = P^\star(q_1 = q_2).$$

Assumption B.2 (Boundedness). *Assume there exist constants $L_\lambda, B_c > 0$, s.t.*

$$\lambda \geq L_\lambda, |c| \leq B_c, \ \ \forall \lambda \in \Lambda, c \in \mathcal{C}.$$

For convenience, for any $p \in [0,1]$, we denote $\bar{p} = 1 - p$. Also, for any prompt pairs q_1, q_2, we denote $\bar{P}^\star(q_1 = q_2) = 1 - P^\star(q_1 = q_2)$ and $\bar{P}_{\theta,\lambda,c}(q_1 = q_2) = 1 - P_{\theta,\lambda,c}(q_1 = q_2)$.

B.1 Convergence guarantee for cross-entropy loss

For any $\theta \in \Theta, \lambda \in \Lambda, c \in \mathcal{C}$, we denote the (binary) cross-entropy loss function as

$$\mathcal{L}_\mu^{\text{BCE}}(\theta, \lambda, c) = -\mathbb{E}_{(q_1,q_2)\sim\mu}[P^\star(q_1 = q_2) \log P_{\theta,\lambda,c}(q_1 = q_2) + \bar{P}^\star(q_1 = q_2) \log \bar{P}_{\theta,\lambda,c}(q_1 = q_2)]. \tag{1}$$

To recover the ground-truth parameter $(\theta^\star, \lambda^\star, c^\star)$, one only needs to solve the optimization problem

$$\min_{\theta\in\Theta,\lambda\in\Lambda,c\in\mathcal{C}} \mathcal{L}_\mu^{\text{BCE}}(\theta, \lambda, c). \tag{2}$$

One may observe that (2) is equivalent to minimizing the expected KL divergence between the ground truth probability $P^\star$ and $P_{\theta,\lambda,c}$.

Our algorithm minimizes the empirical version of the loss function $\mathcal{L}_\mathcal{D}^{\text{BCE}}(\theta, \lambda, c)$, where

$$\mathcal{L}_\mathcal{D}^{\text{BCE}}(\theta, \lambda, c) = \frac{-1}{N} \sum_{(q_{i,1},q_{i,2},p_i)\in\mathcal{D}} \left(p_i \log P_{\theta,\lambda,c}(q_{i,1} = q_{i,2}) + \bar{p}_i \log \bar{P}_{\theta,\lambda,c}(q_{i,1} = q_{i,2})\right).$$

Let $(\hat{\theta}, \hat{\lambda}, \hat{c})$ denote the minimizer of the empirical loss, i.e.,

$$(\hat{\theta}, \hat{\lambda}, \hat{c}) \in \operatorname*{arg\,min}_{\theta\in\Theta,\lambda\in\Lambda,c\in\mathcal{C}} \mathcal{L}_\mathcal{D}^{\text{BCE}}(\theta, \lambda, c).$$

The following theorem provides a finite sample guarantee of the convergence rate of the empirical minimizer:

Theorem B.3 (Convergence rate of the main algorithm, BCE loss)**.** *Under Assumptions B.1 and B.2, for any $\delta > 0$, with probability at least $1 - \delta$, it holds that*

$$\mathbb{E}_{(q_1,q_2)\sim\mu}\left[\left|P^\star(q_1 = q_2) - P_{\hat\theta,\hat\lambda,\hat c}(q_1 = q_2)\right|\right] \le O\left(\frac{\sqrt{L_\lambda^{-1} + B_c}\cdot(\log(1/\delta))^{1/4}}{N^{1/4}}\right).$$

The proof of Theorem B.3 is deferred to Appendix C.1.

B.2 Convergence guarantee for squared log difference loss

In this section, we analyze the convergence rate for another loss function. Define the squared log difference loss function as follows:

$$\mathcal{L}_\mu^{\text{sld}}(\theta,\lambda,c) = \mathbb{E}_{(q_1,q_2)\sim\mu}\left[\left(\log P^\star(q_1 = q_2) - \log P_{\theta,\lambda,c}(q_1 = q_2)\right)^2\right]. \tag{3}$$

Similarly, we also define the empirical squared log difference loss as

$$\mathcal{L}_\mu^{\text{sld}}(\theta,\lambda,c) = \frac{1}{N}\sum_{(q_{i,1},q_{i,2},p_i)\in\mathcal{D}}\left(\log p_i - \log P_{\theta,\lambda,c}(q_{i,1} = q_{i,2})\right)^2. \tag{4}$$

Since $\log 0$ is not well-defined, for theoretical analysis, we assume that each p_i in the dataset $\mathcal{D}$ satisfies $p_i = P^\star(q_{i,1} = q_{i,2})$, i.e., the label is exact the ground-truth probability.

Now, we provide a finite sample convergence guarantee for the squared log difference loss:

Theorem B.4 (Convergence rate of the main algorithm, squared log difference loss)**.** *Under Assumptions B.1 and B.2, for any $\delta > 0$, with probability at least $1 - \delta$, it holds that*

$$\mathbb{E}_{(q_1,q_2)\sim\mu}\left[\left|P^\star(q_1 = q_2) - P_{\hat\theta,\hat\lambda,\hat c}(q_1 = q_2)\right|\right] \le O\left(\frac{(L_\lambda^{-1} + B_c)\cdot(\log(1/\delta))^{1/4}}{N^{1/4}}\right).$$

The proof of Theorem B.4 is deferred to Appendix C.2.

Remark B.5. *Compared to the bound for BCE loss, there is an additional $\sqrt{L_\lambda^{-1} + B_c}$ factor in the bound for squared log difference loss.*

C Missing Proofs

C.1 Proof of Theorem B.3

To prove Theorem B.3, we first present and prove Lemmas C.1 and C.2. Our proof strategy is similar to that of Zhan et al. [2022], Zhu and Zhang [2023].

Lemma C.1. *Under Assumptions B.1 and B.2, for any $\delta > 0$, with probability at least $1 - \delta$, it holds that*

$$\left|\mathcal{L}_\mathcal{D}^{BCE}(\theta,\lambda,c) - \mathcal{L}_\mu^{BCE}(\theta,\lambda,c)\right| \le O\left((L_\lambda^{-1} + B_c)\sqrt{\frac{\log(|\Theta||\Lambda||\mathcal{C}|/\delta)}{N}}\right) \triangleq \epsilon_{stat}^{BCE}$$

for any $\theta \in \Theta, \lambda \in \Lambda, c \in \mathcal{C}$.

Proof. We first consider any fixed $\theta \in \Theta, \lambda \in \Lambda, c \in \mathcal{C}$. One can observe that $\mathcal{L}_\mathcal{D}^{\text{BCE}}(\theta,\lambda,c)$ is an unbiased estimator of $\mathcal{L}_\mu^{\text{BCE}}(\theta,\lambda,c)$ since

$$\begin{aligned}
&\mathbb{E}[\mathcal{L}_\mathcal{D}^{\text{BCE}}(\theta,\lambda,c)]\\
=&-\mathbb{E}_{(q_1,q_2)\sim\mu,p\sim\mathcal{P}(\cdot|q_1,q_2)}\left[p\log P_{\theta,\lambda,c}(q_1 = q_2) + \bar p\log \bar P_{\theta,\lambda,c}(q_1 = q_2)\right]\\
=&-\mathbb{E}_{(q_1,q_2)\sim\mu}\left[\mathbb{E}_{p\sim\mathcal{P}(\cdot|q_1,q_2)}\left[p\log P_{\theta,\lambda,c}(q_1 = q_2) + \bar p\log \bar P_{\theta,\lambda,c}(q_1 = q_2)|q_1,q_2\right]\right]\\
=&-\mathbb{E}_{(q_1,q_2)\sim\mu}\left[P^\star(q_1 = q_2)\log P_{\theta,\lambda,c}(q_1 = q_2) + \bar P^\star(q_1 = q_2)\log \bar P_{\theta,\lambda,c}(q_1 = q_2)\right]\\
=&\mathcal{L}_\mu^{\text{BCE}}(\theta,\lambda,c),
\end{aligned}$$

where the first equality holds since the data points in the dataset are i.i.d. distributed, the second equality holds due to tower property, and the third equality holds by the linearity of expectation.

Also, we note that the empirical loss for each data point can be upper bounded by

$$\begin{aligned}
&\left|p_i \log P_{\theta,\lambda,c}(q_{i,1} = q_{i,2}) + \bar{p}_i \log \bar{P}_{\theta,\lambda,c}(q_{i,1} = q_{i,2})\right| \\
&\leq \max\left\{\left|\log P_{\theta,\lambda,c}(q_{i,1} = q_{i,2})\right|, \left|\log \bar{P}_{\theta,\lambda,c}(q_{i,1} = q_{i,2})\right|\right\} \\
&= \log\left(\max\left\{\frac{1}{P_{\theta,\lambda,c}(q_{i,1} = q_{i,2})}, \frac{1}{1 - P_{\theta,\lambda,c}(q_{i,1} = q_{i,2})}\right\}\right).
\end{aligned}$$

Since $\sigma(-x) = 1 - \sigma(x)$ and $\mathtt{sim}(v_\theta(q_1), v_\theta(q_2))/\lambda - c \in [-L_\lambda^{-1} - B_c, L_\lambda^{-1} + B_c]$ by Assumption B.2, we can obtain that

$$\frac{1}{P_{\theta,\lambda,c}(q_1 = q_2)} = \frac{1}{\sigma(\mathtt{sim}(v_\theta(q_1), v_\theta(q_2))/\lambda - c)} \leq 1 + \exp(L_\lambda^{-1} + B_c).$$

By the symmetry of $\sigma(\cdot)$ and the range of $\mathtt{sim}(v_\theta(q_1), v_\theta(q_2))/\lambda - c$, it also holds that

$$\frac{1}{1 - P_{\theta,\lambda,c}(q_1 = q_2)} \leq 1 + \exp(L_\lambda^{-1} + B_c).$$

Therefore,

$$\begin{aligned}
&\left|p_i \log P_{\theta,\lambda,c}(q_{i,1} = q_{i,2}) + \bar{p}_i \log \bar{P}_{\theta,\lambda,c}(q_{i,1} = q_{i,2})\right| \\
&\leq \log\left(1 + \exp(L_\lambda^{-1} + B_c)\right) \leq O(L_\lambda^{-1} + B_c).
\end{aligned}$$

By Hoeffding's inequality, we have with probability at least $1 - \delta$, it holds that

$$\left|\mathcal{L}_{\mathcal{D}}^{\mathrm{BCE}}(\theta, \lambda, c) - \mathcal{L}_{\mu}^{\mathrm{BCE}}(\theta, \lambda, c)\right| \leq O\left((L_\lambda^{-1} + B_c)\sqrt{\frac{\log(1/\delta)}{N}}\right).$$

Applying a union bound over all $\theta \in \Theta, \lambda \in \Lambda, c \in \mathcal{C}$ concludes the result. □

Lemma C.2. *Under Assumptions B.1 and B.2, with probability at least* $1 - \delta$*, it holds that*

$$\mathcal{L}_{\mu}^{BCE}(\hat{\theta}, \hat{\lambda}, \hat{c}) - \mathcal{L}_{\mu}^{BCE}(\theta^\star, \lambda^\star, c^\star) \leq 2\epsilon_{stat}^{BCE}.$$

where ϵ_{stat}^{BCE} *is defined in Lemma C.1 and*

$$(\hat{\theta}, \hat{\lambda}, \hat{c}) \in \underset{\theta \in \Theta, \lambda \in \Lambda, c \in \mathcal{C}}{\arg\min}\, \mathcal{L}_{\mathcal{D}}^{BCE}(\theta, \lambda, c).$$

Proof. We condition on the high probability event in Lemma C.1. Note that

$$\begin{aligned}
&\mathcal{L}_{\mu}^{\mathrm{BCE}}(\hat{\theta}, \hat{\lambda}, \hat{c}) - \mathcal{L}_{\mu}^{\mathrm{BCE}}(\theta^\star, \lambda^\star, c^\star) \\
&= \underbrace{\mathcal{L}_{\mu}^{\mathrm{BCE}}(\hat{\theta}, \hat{\lambda}, \hat{c}) - \mathcal{L}_{\mathcal{D}}^{\mathrm{BCE}}(\hat{\theta}, \hat{\lambda}, \hat{c})}_{(1)} + \underbrace{\mathcal{L}_{\mathcal{D}}^{\mathrm{BCE}}(\hat{\theta}, \hat{\lambda}, \hat{c}) - \mathcal{L}_{\mathcal{D}}^{\mathrm{BCE}}(\theta^\star, \lambda^\star, c^\star)}_{(2)} \\
&\quad + \underbrace{\mathcal{L}_{\mathcal{D}}^{\mathrm{BCE}}(\theta^\star, \lambda^\star, c^\star) - \mathcal{L}_{\mu}^{\mathrm{BCE}}(\theta^\star, \lambda^\star, c^\star)}_{(3)}.
\end{aligned}$$

$(1), (3) \leq \epsilon_{\mathrm{stat}}^{\mathrm{BCE}}$ by Lemma C.1 and $(2) \leq 0$ by the optimality of $(\hat{\theta}, \hat{\lambda}, \hat{c})$, which completes the proof. □

Equipped with Lemmas C.1 and C.2, we are now able to prove Theorem B.3.

Proof of Theorem B.3. We condition on the high probability event in Lemma C.2. Note that by the realizability of the ground-truth probability (Assumption B.1) and the property of binary cross-entropy, we have

$$(\theta^\star, \lambda^\star, c^\star) \in \underset{\theta \in \Theta, \lambda \in \Lambda, c \in \mathcal{C}}{\arg\min}\, \mathcal{L}_{\mu}^{\mathrm{BCE}}(\theta, \lambda, c).$$

For any $\theta \in \Theta, \lambda \in \Lambda, c \in \mathcal{C}$, we map (θ, λ, c) to a function $f_{\theta,\lambda,c}(\cdot,\cdot) : \mathcal{Q} \times \mathcal{Q} \to [0, 1]$, where

$$f_{\theta,\lambda,c}(q_1, q_2) = P_{\theta,\lambda,c}(q_1 = q_2), \ \ \forall q_1, q_2 \in \mathcal{Q}.$$

Moreover, we define functional h s.t. for any function $f : \mathcal{Q} \times \mathcal{Q} \to [0, 1]$,

$$h(f) = -\mathbb{E}_{(q_1,q_2)\sim\mu}[P^\star(q_1 = q_2) \log f(q_1, q_2) + \bar{P}^\star(q_1 = q_2) \log(1 - f(q_1, q_2))].$$

By the definition of BCE loss, $h(f_{\theta,\lambda,c}) = \mathcal{L}^{\text{BCE}}_\mu(\theta, \lambda, c)$. Therefore, Lemma C.2 translates to

$$h(f_{\hat{\theta},\hat{\lambda},\hat{c}}) - h(f_{\theta^\star,\lambda^\star,c^\star}) \le 2\epsilon^{\text{BCE}}_{\text{stat}}. \tag{5}$$

Note that $f_{\theta^\star,\lambda^\star,c^\star}$ is still the minimizer of $h(f)$ even if f cannot be induced by some θ, λ, c.

We also observe that $h(f)$ is 1-strongly convex w.r.t. f in $\|\cdot\|_{2,\mu}$ norm. To see why this is the case, one can calculate the second-order derivative of h w.r.t. $f(q_1, q_2)$ for any $(q_1, q_2) \in \mathcal{Q} \times \mathcal{Q}$, which is

$$\frac{\partial^2 h}{\partial f^2}(q_1, q_2) = \frac{P^\star(q_1 = q_2)}{f^2(q_1, q_2)} + \frac{\bar{P}^\star(q_1 = q_2)}{(1 - f(q_1, q_2))^2} \ge P^\star(q_1 = q_2) + \bar{P}^\star(q_1 = q_2) = 1,$$

which demonstrates the strong convexity. Therefore, by strong convexity and the optimality of $f_{\theta^\star,\lambda^\star,c^\star}$, we can obtain that

$$h(f_{\hat{\theta},\hat{\lambda},\hat{c}}) \ge h(f_{\theta^\star,\lambda^\star,c^\star}) + \frac{1}{2}\|f_{\theta^\star,\lambda^\star,c^\star} - f_{\hat{\theta},\hat{\lambda},\hat{c}}\|^2_{2,\mu}.$$

Combining (5), we have

$$\|f_{\theta^\star,\lambda^\star,c^\star} - f_{\hat{\theta},\hat{\lambda},\hat{c}}\|_{2,\mu} \le 2\sqrt{\epsilon^{\text{BCE}}_{\text{stat}}}.$$

Finally, by Cauchy–Schwarz inequality, we can conclude

$$\begin{aligned}&\mathbb{E}_{(q_1,q_2)\sim\mu}\left[\left|P^\star(q_1 = q_2) - P_{\hat{\theta},\hat{\lambda},\hat{c}}(q_1 = q_2)\right|\right] \\ =&\|f_{\theta^\star,\lambda^\star,c^\star} - f_{\hat{\theta},\hat{\lambda},\hat{c}}\|_{1,\mu} \le \|f_{\theta^\star,\lambda^\star,c^\star} - f_{\hat{\theta},\hat{\lambda},\hat{c}}\|_{2,\mu} \le 2\sqrt{\epsilon^{\text{BCE}}_{\text{stat}}}.\end{aligned}$$

□

C.2 Proof of Theorem B.4

Proof of Theorem B.4. The proof is similar to the proof of Theorem B.3. First, it is easy to see that the empirical loss $\mathcal{L}^{\text{sld}}_{\mathcal{D}}(\theta, \lambda, c)$ is an unbiased estimator of $\mathcal{L}^{\text{sld}}_\mu(\theta, \lambda, c)$ by definition since $p_i = P^\star(q_{i,1} = q_{i,2})$. Also,

$$(\log p_i - \log P_{\theta,\lambda,c}(q_{i,1} = q_{i,2}))^2 \le \left(\log\left(1 + \exp(L_\lambda^{-1} + B_c)\right)\right)^2 = O\left((L_\lambda^{-1} + B_c)^2\right).$$

Therefore, by Hoeffding's inequality and union bound, we have that with probability at least $1 - \delta$, it holds that

$$\left|\mathcal{L}^{\text{sld}}_{\mathcal{D}}(\theta, \lambda, c) - \mathcal{L}^{\text{sld}}_\mu(\theta, \lambda, c)\right| \le O\left((L_\lambda^{-1} + B_c)^2\sqrt{\frac{\log(|\Theta||\Lambda||\mathcal{C}|/\delta)}{N}}\right) \triangleq \epsilon^{\text{sld}}_{\text{stat}}.$$

Similar to Lemma C.2, we can obtain that

$$\mathcal{L}^{\text{sld}}_\mu(\hat{\theta}, \hat{\lambda}, \hat{c}) - \mathcal{L}^{\text{sld}}_\mu(\theta^\star, \lambda^\star, c^\star) \le 2\epsilon^{\text{sld}}_{\text{stat}}. \tag{6}$$

Now, for any $\theta \in \Theta, \lambda \in \Lambda, c \in \mathcal{C}$, we map (θ, λ, c) to a function $f_{\theta,\lambda,c}(\cdot,\cdot) : \mathcal{Q} \times \mathcal{Q} \to [0, +\infty)$, where

$$f_{\theta,\lambda,c}(q_1, q_2) = -\log P_{\theta,\lambda,c}(q_1 = q_2), \ \ \forall q_1, q_2 \in \mathcal{Q}.$$

Moreover, we define functional h s.t. for any function $f : \mathcal{Q} \times \mathcal{Q} \to [0, 1]$,

$$h(f) = \mathbb{E}_{(q_1,q_2)\sim\mu}[(f(q_1, q_2) + \log P^\star(q_1 = q_2))^2].$$

By the definition of squared log difference loss, $h(f_{\theta,\lambda,c}) = \mathcal{L}_\mu^{\mathrm{sld}}(\theta,\lambda,c)$. Therefore, (6) translates to

$$h(f_{\hat{\theta},\hat{\lambda},\hat{c}}) - h(f_{\theta^\star,\lambda^\star,c^\star}) \le 2\epsilon_{\mathrm{stat}}^{\mathrm{sld}}. \tag{7}$$

Note that $f_{\theta^\star,\lambda^\star,c^\star}$ is still the minimizer of $h(f)$ even if f cannot be induced by some θ, λ, c.

We also observe that $h(f)$ is 2-strongly convex w.r.t. f in $\|\cdot\|_{2,\mu}$ norm by calculating the second-order derivative of h w.r.t. f. Therefore, by strong convexity and the optimality of $f_{\theta^\star,\lambda^\star,c^\star}$, we can obtain that

$$h(f_{\hat{\theta},\hat{\lambda},\hat{c}}) \ge h(f_{\theta^\star,\lambda^\star,c^\star}) + \|f_{\theta^\star,\lambda^\star,c^\star} - f_{\hat{\theta},\hat{\lambda},\hat{c}}\|_{2,\mu}^2.$$

Combining (7), we can obtain

$$\mathbb{E}_{(q_1,q_2)\sim\mu}\left[(\log P^\star(q_1=q_2) - \log P_{\theta,\lambda,c}(q_1=q_2))^2\right] = \|f_{\theta^\star,\lambda^\star,c^\star} - f_{\hat{\theta},\hat{\lambda},\hat{c}}\|_{2,\mu}^2 \le 2\epsilon_{\mathrm{stat}}^{\mathrm{sld}}.$$

Note that by mean value theorem, for any $0 < x < y < 1$, $\log x - \log y = (x-y)/z$ for some $z \in (x,y)$. Therefore, $(\log x - \log y)^2 = (x-y)^2/z^2 > (x-y)^2$. This implies

$$\begin{aligned}
&\mathbb{E}_{(q_1,q_2)\sim\mu}\left[(P^\star(q_1=q_2) - P_{\theta,\lambda,c}(q_1=q_2))^2\right] \\
\le&\mathbb{E}_{(q_1,q_2)\sim\mu}\left[(\log P^\star(q_1=q_2) - \log P_{\theta,\lambda,c}(q_1=q_2))^2\right] = 2\epsilon_{\mathrm{stat}}^{\mathrm{sld}}.
\end{aligned}$$

By Cauchy–Schwarz inequality, we can conclude

$$\begin{aligned}
&\mathbb{E}_{(q_1,q_2)\sim\mu}\left[\left|P^\star(q_1=q_2) - P_{\hat{\theta},\hat{\lambda},\hat{c}}(q_1=q_2)\right|\right] \\
\le&\sqrt{\mathbb{E}_{(q_1,q_2)\sim\mu}\left[(P^\star(q_1=q_2) - P_{\theta,\lambda,c}(q_1=q_2))^2\right]} \le \sqrt{2\epsilon_{\mathrm{stat}}^{\mathrm{sld}}}.
\end{aligned}$$

□