Thompson sampling for interactive Bayesian optimization of dynamic masking-based language model pre-training

Iñigo Urteaga
Applied Physics and Applied Mathematics
Data Science Institute, Columbia University
New York, NY, USA
inigo.urteaga@columbia.edu

Moulay-Zaïdane Draïdia
Data Science Institute
Columbia University
New York, NY, USA
mad2314@columbia.edu

Tomer Lancewicki †
Walmart Global Tech,
USA
tomer.lancewicki@walmart.com

Shahram Khadivi
eBay Inc.,
Aachen, Germany
skhadivi@ebay.com

Abstract

We design and evaluate a Thompson sampling-based interactive Bayesian optimization algorithm that leverages a Gaussian process reward model of the Masked Language Model (MLM) pre-training objective, for its sequential minimization. Transformer-based language model (TLM) pre-training requires large volumes of data and high computational resources, while introducing many unresolved design choices, such as hyperparameter selection of the pre-training procedure. We here fit TLM pre-training validation losses with a Gaussian process, and formulate a Thompson sampling bandit policy that maximizes its sequentially attained cumulative rewards. Instead of MLM pre-training with fixed masking probabilities, the proposed Gaussian process-based Thompson sampling (GP-TS) accelerates and improves MLM pre-training performance by sequentially selecting masking hyperparameters of the language model. GP-TS provides an interactive, efficient framework for pre-training TLMs, as it attains better MLM pre-training loss in less epochs, avoiding costly hyperparameter selection techniques.

1 Introduction

In the field of Natural Language Processing (NLP), models for learning unsupervised representations from unlabeled text based on Transformer architectures (Vaswani et al., 2017) have attained state-of-the-art results on diverse tasks (Kalyan et al., 2021). Transformer-based language models (TLMs), such as BERT (Devlin et al., 2018) and RoBERTa (Liu et al., 2019), rely on the combination of an unsupervised pre-training of the model, and subsequent task-specific fine-tuning procedures.

† Work done while at eBay Inc. San Jose, CA.

Even if conceptually simple and empirically powerful, pre-training is challenging and expensive: the relationship between the Transformer architecture, the training corpus, the evaluation metrics and the tunable hyperparameters is multi-modal and complex. Furthermore, previously overlooked pre-training design choices (such as deciding on the pre-training metric and optimizing its hyperparameters) result in significant performance differences.

In this work, we improve the pre-training procedure of TLMs by designing a Gaussian process-based multi-armed bandit (Lattimore and Szepesvári, 2019) framework for sequentially selecting pre-training hyperparameters that result in optimized performance. We cast the TLM pre-training hyperparameter selection procedure as an interactive sequential decision process, in which at each interaction, a Thompson sampling-based (Thompson, 1933; Russo et al., 2018) bandit agent selects an action (e.g., pre-training hyperparameters) to maximize observed cumulative rewards (e.g., a pre-training metric of interest).

2 Thompson sampling for interactive optimization of TLM pre-training

We propose a Gaussian process based Thompson sampling (GP-TS) algorithm —with pseudo-code provided in Algorithm 1— that views the TLM pre-training procedure as an interactive, black-box minimization task. We define TLM pre-training steps, i.e., a fixed number of stochastic gradient updates u[1], as bandit interactions $t = 1, \cdots, T$; with the goal of minimizing a pre-training objective $l(\cdot|\psi)$ given tunable hyperparameters ψ.

[1] Note that u stochastic gradient updates might or might not correspond to a full pre-training epoch.

Algorithm 1 GP-TS for interactive optimization of TLM pre-training

1: **Input**: TLM and training corpus
2: **Input**: Pre-training hyperparameter space Ψ
3: **Input**: Number of bandit interactions T
4: **Input**: Number of updates per-interaction u
5: **Input**: GP prior functions $\mu(\cdot)$ and $k(\cdot,\cdot)$
6: **Input**: GP initial hyperparameters θ_0
7: **Initialize**: $\mathcal{A}=\Psi$, $\hat{\theta}_1=\theta_0$, $\mathcal{H}_1=\emptyset$
8: **for** $t=1,\cdots,T$ **do**
9: Draw posterior sample from the posterior GP, i.e., $\mu_a^{(t)} \sim f(\mu_t(a|\hat{\theta}_t), k_t(a,a'|\hat{\theta}_t))$.
10: Select arm based on drawn posterior sample, i.e., $a_t = \mathrm{argmax}_{a' \in \mathcal{A}}\, \mu_{a'}^{(t)}$.
11: Run TLM pre-training for u steps, with hyperparameters $\psi_t = a_t$.
12: Compute validation loss of pre-trained TLM, i.e., $\bar{y}_t$ as in Equation (7).
13: Observe bandit reward, i.e., r_t as in Equation (1).
14: Update bandit history, $\mathcal{H}_{1:t} = \mathcal{H}_{1:t-1} \cup \{a_t, r_t\}$
15: Fit GP model with $\mathcal{H}_{1:t}$, i.e., $\hat{\theta}_{t+1} = \mathrm{argmax}_\theta \log p\left(r_{1:t}|f(a_{1:t}),\theta\right)$.
16: **end for**

We identify the pre-training hyperparameters at interaction t, ψ_t, as the bandit's arms, $a_t=\psi_t$; and define observed rewards as the self-normalized difference in pre-training losses between interactions[2], computed in the validation set D_{val},

$$r_t(\psi_t) = \frac{[-\bar{y}_t(D_{val};\psi_t)] - [-\bar{y}_{t-1}(D_{val};\psi_{t-1})]}{[-\bar{y}_{t-1}(D_{val};\psi_{t-1})]} \; . \tag{1}$$

In practice, TLM pre-training is carried out based on empirical risk minimization, i.e., only empirical estimates $\bar{y}_t(\psi_t)$ of the true objective are available. To accommodate the stochastic nature of these noisy estimates $\bar{y}_t(\psi_t)$ of the black-box loss function $l(\cdot|\psi_t)$ —that we aim to optimize with respect to its hyperparameters ψ— we model the observed rewards via a surrogate Gaussian process,

$$r_t(\psi_t) = f(\psi_t;\theta) + \epsilon_t \; , \tag{2}$$

where $f(\cdot;\theta)$ is a Gaussian process (GP), and ϵ_t is independent and identically distributed noise, reflecting the stochasticity of the empirical rewards.

[2] By normalizing reward differences per-interaction, we aim at mitigating the potential non-stationary effect hyperparameters might have on the TLM pre-training procedure.

Our TLM pre-training use-case is *random* dynamic masking as proposed by Liu et al. (2019): the actions (i.e., the bandit arms) are the dynamic masking choices; and the masked-language model metric, the unknown objective function $l(\cdot|\psi)$ the bandit shall optimize.

The proposed GP-TS algorithm operates by sequentially selecting arms (hyperparameters) $a_t=\psi_t$ and observing rewards $r_t(\psi_t)$ as in Equation (2). At each bandit interaction $t=1,\cdots,T$, we pre-train a TLM for u stochastic updates given selected hyperparameters ψ_t (e.g., the number of tokens to mask and their associated random masking probabilities), by minimizing the MLM loss between a random training set mini-batch $D_b \in D$ and its masked counterpart $\widehat{D_b}$,

$$\begin{aligned} y(D_b;\psi) &= l(D_b,\widehat{D_b};w,\psi) && (3)\\ &= -\sum_{d\in D_b}\sum_{l_d=1}^{L_d} m_{l_d}\log p(l_d|\widehat{l_d};w,\psi) && (4)\\ &= -\sum_{d\in D_b}\sum_{l_d=1}^{L_d} m_{l_d}\log\left(\frac{e^{\left(h(\widehat{l_d};w,\psi)^\top \chi(l_d)\right)}}{\sum_{l'_d=1}^{L_d} e^{\left(h(\widehat{l'_d};w,\psi)^\top \chi(l'_d)\right)}}\right) && (5)\end{aligned}$$

where $h(\widehat{l_d};w,\psi)$ denotes the representation of the TLM for the masked token and $\chi(l_d)$, its original embedding. We explicitly indicate the TLM architecture parameters $w \in W$, the hyperparameters ψ of the pre-training procedure, and denote with $m_{l_d}=\{0,1\}$ the masked tokens l_d in $\widehat{d}$ of the original input sequence $d \in D_b$. After each pre-training interaction t, we evaluate the pre-trained model's *averaged* MLM loss in the validation set,

$$\begin{aligned} \bar{y}_t(D_{val};\psi_t) &= \bar{l}(D_{val},\widehat{D_{val}};w,\psi_t) && (6)\\ &= -\sum_{d\in D_{val}} \frac{\sum_{l_d=1}^{L} m_{l_d}\log p(l_d|\widehat{l_d};w,\psi_t)}{\sum_{l_d=1}^{L} m_{l_d}} \; , && (7)\end{aligned}$$

and compute bandit rewards r_t as in Equation (1).

We update (i.e., re-fit) the GP model to the history of observed input (action)-output (rewards) evidence $\mathcal{H}_{1:t}$ after every interaction; for instance, via Type-II MLE as in Step 12 of Algorithm 1. We draw a posterior sample from this updated GP reward model (Step 6 of Algorithm 1) for the GP-TS policy to determine (in Step 7 of Algorithm 1) the hyperparameters $a_t=\psi_t$ of the next interaction of the pre-training procedure, towards maximization of the observed cumulative rewards, i.e., $R_T=\sum_{t=1}^{T} r_t(\psi_t)$.

We note that any TLM architecture can be used within the proposed GP-TS, as long as the pre-training hyperparameter space $\psi \in \Psi$ is identified, and rewards as in Equation (1) can be computed based on a given pre-training objective.

The GP reward model in Equation (2) shall accommodate continuous arms a_t, with dimensionality determined by the TLM pre-training hyperparameter space Ψ, and prior mean and kernel functions decided by the practitioner. We experiment here with zero-mean and RBF kernel GPs with Gaussian observation noise, as closed-form posterior inference expressions can be efficiently computed in this case (Rasmussen and Williams, 2005; Pleiss et al., 2018).

3 Experiments

3.1 Evaluation set-up

We probe the ability of the proposed GP-TS method —given a dataset, a TLM architecture, and a computational budget— to efficiently pre-train well-performing language models[3]. We implement the RoBERTa model (Liu et al., 2019) provided by Fairseq (Ott et al., 2019) and incorporate it as a module in our proposed framework, which consists of a Python implementation of GP-TS as in Algorithm 1 with GP modules in GPyTorch (Gardner et al., 2018) —implementation and configuration details are provided in Appendix A.

We compare pre-training performance of RoBERTa models based on a grid-search over masking hyperparameters —as originally executed by Liu et al. (2019)— to RoBERTa trained by the interactive GP-TS agent.

We study two variants of GP-TS, depending on which masking hyperparameters it optimizes:
(i) `GP-TS` ρ, where the bandit arm is the unidimensional masking probability ρ of replacing an input token with the mask token (we fix other hyperparameters to their default $\gamma = 0.1$ and $\lambda = 0.1$ values suggested by Liu et al. (2019)); and
(ii) `GP-TS` $\psi = (\rho, \gamma, \lambda)$, where GP-TS optimizes over all the dynamic masking hyperparameters involved in MLM pre-training, i.e., the bandit search space is a three-dimensional hypercube $\Psi = (0, 0.5)^3$, with no previous expert guidance on hyperparameter selection.

[3]We scrutinize the pre-training procedure of RoBERTa models under equal experimental conditions and do not compare performance to state-of-the-art, large-scale TLMs.

Pre-training datasets. We gather three distinct datasets, two based on publicly available corpora, and one based on private data from eBay:

- **wiki-c4**: We pre-process and encode the publicly available Wikitext-103 (Merity et al., 2016) and Google's c4 RealNews (Zellers et al., 2019) datasets for pre-training, from scratch, each of the candidate TLMs. This corpora is similar to those originally used by Devlin et al. (2018) and Liu et al. (2019), and is publicly accessible for researchers.
- **mimic**: We pre-process and encode the free-text clinical notes available in the public MIMIC-III Clinical database (Pollard and Johnson, 2016), which contains deidentified nursing and physician notes, ECG reports, imaging reports, and discharge summaries for patients who stayed within the intensive care units at Beth Israel Deaconess Medical Center.
- **e-commerce**: We pre-process and encode a random subset of eBay marketplace inventories, which contains different product titles and descriptions provided by marketplace users, as well as category tags associated with each item and product reviews.

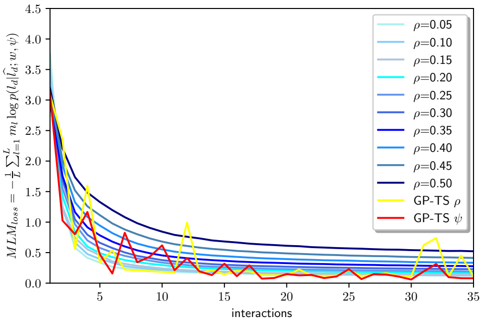
Each dataset contains text of very different linguistic characteristics and sizes (see summary statistics in Appendix A.2), which we leverage to investigate TLM pre-training across a variety of settings.

We evaluate candidate TLMs both (i) when pre-training from *scratch*, i.e., from a randomly initialized architecture; and (ii) with *continual* pre-training, i.e., when continuing pre-training a TLM architecture previously trained in other NLP corpora (Kalyan et al., 2021).

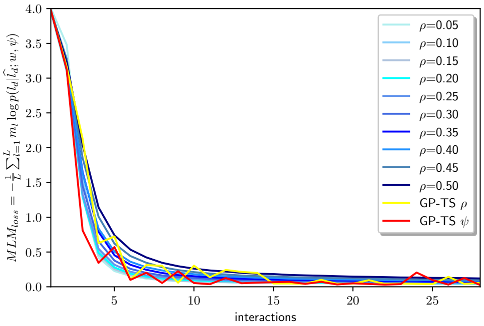
Continual pre-training results presented here are for the RoBERTa-base architecture as pre-trained by Facebook Research (2022) that we continue to pre-train in domain-specific datasets, i.e., `mimic` and `e-commerce`.

3.2 GP-TS pre-training of RoBERTa models

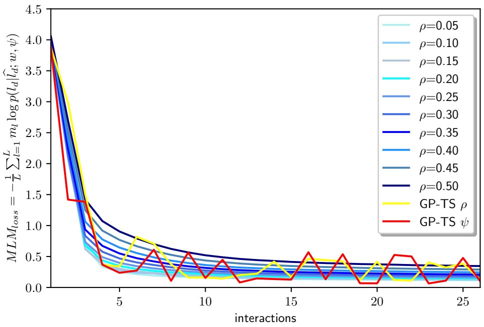
We compare from *scratch* pre-training performance of all RoBERTa-base models, pre-trained either with fixed hyperparameters or guided by the proposed GP-TS, in Figure 1; where we illustrate the averaged MLM validation loss of each model over pre-training interactions. We observe that GP-TS provides accelerated and successful pre-training performance across all studied datasets.



(a) `wiki-c4`.



(b) `mimic`.



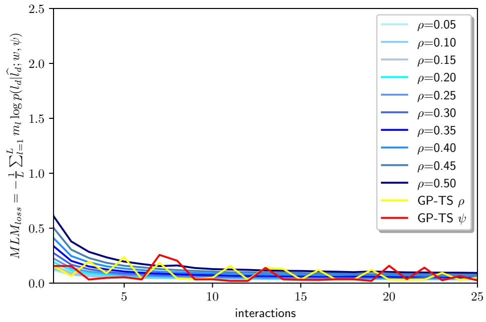
(c) `e-commerce`.

Figure 1: Averaged MLM validation loss performance comparison (lower is better) of grid-search based and GP-TS based from *scratch* pre-trained RoBERTa models, over interactions.

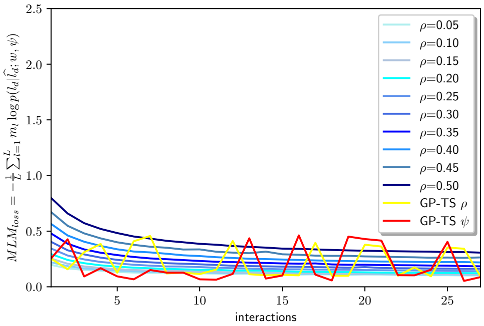
MLM loss values for GP-TS pre-trained models fluctuate across interactions, depending on the value selected by GP-TS at each interaction. However, GP-TS pre-trains the best performing RoBERTa models, the fastest: i.e., it pre-trains models with the lowest MLM in less interactions. Namely, the benefits of interactive GP-TS pre-training do not only pertain to attained MLM values, but to an accelerated procedure as well.

Results for *continual* pre-training performance are provided in Figure 2, where we observe that the RoBERTa architecture, when continually pre-trained with GP-TS, achieves the best MLM loss in fewer epochs across the studied in-domain datasets.

GP-TS efficiently pre-trains RoBERTa models —across datasets and pre-training approaches (from-scratch and continual)— not only when optimizing over ρ, but even when operating over the 3-dimensional ψ_t hyperparameter space.



(a) `mimic`.



(b) `e-commerce`.

Figure 2: Averaged MLM validation loss performance comparison (lower is better) of grid-search based and GP-TS based *continually* pre-trained RoBERTa models, over interactions.

We conclude that GP-TS finds better TLM models than grid-search based alternatives in less interactions, as it interactively finds sequences of dynamic masking hyperparameters —even when no good guesses for them are available— that minimize MLM pre-training loss across datasets, when pre-training both from-scratch and continually.

4 Conclusion

We present a Gaussian process-based Thompson sampling (GP-TS) interactive framework for TLM pre-training loss minimization, by modeling noisy evaluations of the pre-training objective (e.g., the MLM loss) as drawn from a surrogate Gaussian process that the bandit agent aims to maximize.

We provide empirical evidence of how GP-TS, when applied to MLM dynamic masking optimization, attains superior and accelerated (both from-scratch and continual) pre-training performance. Pre-training efficiency is of critical importance in practice, due to the significant resource utilization savings afforded: a grid-search over hyperparameters can be avoided, as GP-TS is able to sequentially select dynamic masking hyperparameters that result in fast and performant pre-trained models.

Future work consists on evaluating the downstream performance benefits of TLMs interactively pre-trained via GP-TS: by fine-tuning GP-TS pre-trained TLM models in downstream tasks, and by leveraging GP-TS to directly maximize downstream metrics of interest.

Limitations

There are several limitations to account for in the presented work. First, the large GPU requirements for the execution and replication of the presented experiments. Second, the lack of empirical results beyond English corpora, and how morphologically and syntactically more complex corpora may affect the presented evidence. Finally, our conclusions are limited to RoBERTa models pre-trained via dynamic masking, and therefore, investigation of how GP-TS generalizes to other hyperparameter selection and TLM architectures is lacking.

Ethics Statement

This work does not raise any significant ethical considerations beyond those associated with the use and biases of pre-collected data, the energetic and environmental impact of extensive GPU resource usage, and the downstream applications of language models.

We acknowledge the potential implicit biases within the publicly available datasets used. E.g., `mimic` reports are limited to the population attended at Beth Israel Deaconess Medical Center, and may contain implicit biases of health practitioners there. We have carefully sampled data for the `e-commerce` dataset to avoid biases over specific products, users and sellers.

In addition, we are aware of the rising concerns pertaining to the carbon footprint of large language models (Patterson et al., 2021), and the significant impact hyperparameter selection techniques have on resource utilization and power consumption (Puvis de Chavannes et al., 2021).

Finally, we acknowledge the wide range of established and anticipated risks that language models pose to society (Weidinger et al., 2021).

Acknowledgements

Iñigo Urteaga and Moulay-Zaïdane Draïdia were partially supported by a project funded by eBay's Research and University Partnership for Technology (eRUPT) program.

We also acknowledge computing resources from Columbia University's Shared Research Computing Facility project, which is supported by NIH Research Facility Improvement Grant 1G20RR030893-01, and associated funds from the New York State Empire State Development, Division of Science Technology and Innovation (NYSTAR) Contract C090171, both awarded April 15, 2010.

References

Fairseq by Facebook Research. 2022. RoBERTa: A Robustly Optimized BERT Pretraining Approach, pre-trained model using the BERT-base architecture. Available online at `https://dl.fbaipublicfiles.com/fairseq/models/roberta.base.tar.gz`.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jacob R Gardner, Geoff Pleiss, David Bindel, Kilian Q Weinberger, and Andrew Gordon Wilson. 2018. GPyTorch: Blackbox Matrix-Matrix Gaussian Process Inference with GPU Acceleration. In *Advances in Neural Information Processing Systems*.

Katikapalli Subramanyam Kalyan, Ajit Rajasekharan, and Sivanesan Sangeetha. 2021. Ammus: A survey of transformer-based pretrained models in natural language processing. *arXiv preprint arXiv:2108.05542*.

Tor Lattimore and Csaba Szepesvári. 2019. *Bandit algorithms*. Preprint.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A Fast, Extensible Toolkit for Sequence Modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

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

Geoff Pleiss, Jacob Gardner, Kilian Weinberger, and Andrew Gordon Wilson. 2018. Constant-Time Predictive Distributions for Gaussian Processes. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4114–4123. PMLR.

Tom J Pollard and Alistair EW Johnson. 2016. The mimic-iii clinical database (version 1.4). *The MIMIC-III Clinical Database. PhysioNet*.

Lucas Høyberg Puvis de Chavannes, Mads Guldborg Kjeldgaard Kongsbak, Timmie Rantzau, and Leon Derczynski. 2021. Hyperparameter power impact in transformer language model training. In *Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing*, pages 96–118, Virtual. Association for Computational Linguistics.

Carl Edward Rasmussen and Christopher K. I. Williams. 2005. *Gaussian Processes for Machine Learning*. The MIT Press.

Daniel J. Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, and Zheng Wen. 2018. A Tutorial on Thompson Sampling. *Foundations and Trends in Machine Learning*, 11(1):1–96.

William R. Thompson. 1933. On the Likelihood that One Unknown Probability Exceeds Another in View of the Evidence of Two Samples. *Biometrika*, 25(3/4):285–294.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Laura Weidinger, John Mellor, Maribeth Rauh, Conor Griffin, Jonathan Uesato, Po-Sen Huang, Myra Cheng, Mia Glaese, Borja Balle, Atoosa Kasirzadeh, et al. 2021. Ethical and social risks of harm from language models. *arXiv preprint arXiv:2112.04359*.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

A Appendix: Implementation and experimentation details

A.1 Gaussian process

We implement Gaussian process modules based on GPyTorch (Gardner et al., 2018), and execute all experiments with a GP process prior and GP fitting details as described in Table 1.

Table 1: Gaussian Process prior and hyperparameters.

Hyperparameter	Initial Value
GP Model	
Mean Function	Constant
Prior constant	0
Kernel Function	Scaled RBF Kernel
Prior output-scale	1
Prior length-scale	0.25
Observation Model	
Likelihood function	Gaussian
Noise variance	1
Training details	
Loss function	ExactMarginalLogLikelihood
train max iters	100
loss epsilon	0.01
Optimizer	
optimizer	adam
lr	0.1

A.2 Summary statistics of the datasets

We split each dataset into 80%-10%-10% training, validation and test sets for our experiments, with summary statistics of each set provided in Table 2.

Table 2: Summary statistics of the pre-training datasets.

Dataset		Total word count	Average words per sentence
`wiki-c4`	Training	4,517,625,794	35.9
	Validation	735,950,955	35.6
	Test	735,571,833	35.6
`mimic`	Training	402,720,632	216.7
	Validation	82,340,235	658.7
	Test	18,735,884	187.3
`e-commerce`	Training	3,935,845,017	5.6
	Validation	494,802,278	5.5
	Test	482,733,197	5.5

A.3 RoBERTa pre-training

We pre-train all RoBERTa models (based on the BERT-base architecture of 125M parameters) by minimizing the MLM loss with dynamic masking in a server with 8 Tesla V100-SXM2-32GB GPUs.

We execute the RoBERTa pre-training procedure as described in Fairseq's RoBERTa pre-training tutorial[4], with specific hyperparameters as described in Table 3.

The interactions for `wiki-c4` and `e-commerce` contain 1000 updates each (i.e., $u = 1000$), while we reduce the number of updates per-interaction to $u = 500$ when pre-training with `mimic` notes.

Table 3: RoBERTa pre-training hyperparameters.

Hyperparameter	Value
Architecture	RoBERTa base
Task	masked lm
Criterion	masked lm
Model details	
dropout	0.1
attention-dropout	0.1
weight-decay	0.01
Training details	
batch-size	32
update-freq	16
sample-break-mode	complete
tokens-per-sample	512
Optimizer	
optimizer	adam
adam-betas	(0.9,0.98)
adam-eps	1e-6
clip-norm	1.0
Learning rate	
lr	0.0005
lr-scheduler	polynomial decay
linear-warmup-updates	1000
Dynamic masking	
mask-prob	ρ
leave-unmasked-prob	0.1
random-token-prob	0.1

[4]Available at https://github.com/pytorch/fairseq/blob/main/examples/roberta/README.pretraining.md