Do You Have the Right Scissors?
Tailoring Pre-trained Language Models via Monte-Carlo Methods

Ning Miao Yuxuan Song Hao Zhou Lei Li
ByteDance AI lab
{miaoning,songyuxuan,zhouhao.nlp,lileilab}@bytedance.com

Abstract

It has been a common approach to pre-train a language model on a large corpus and fine-tune it on task-specific data. In practice, we observe that fine-tuning a pre-trained model on a small dataset may lead to over- and/or under-estimation problem. In this paper, we propose MC-Tailor, a novel method to alleviate the above issue in text generation tasks by truncating and transferring the probability mass from over-estimated regions to under-estimated ones. Experiments on a variety of text generation datasets show that MC-Tailor consistently and significantly outperforms the fine-tuning approach. Our code is available at https://github.com/NingMiao/MC-tailor.

1 Introduction

Recently, pre-trained language models (PLM), *e.g.* GPT-2 (Radford et al., 2019), have shown great promise in many applications of natural language generation, such as stylized text generation (Syed et al., 2019) and dialog system (Wolf et al., 2019). PLM is obtained by first *pre-training* on large-scaled raw sentences (always general domain corpus), and then used in downstream tasks by *fine-tuning* on task-specific datasets (always from some specific domains). Specifically, given a pre-trained GPT-2 model, to generate sentences of email domain, we always need to fine-tune the GPT-2 on a small set of email domain corpus.

However, we argue that to get desired sentence outputs, fine-tuning PLM on a specific domain dataset is not necessarily the best, especially when the fine-tuning dataset is of a small size. Typically, fine-tuning is conducted through Maximum Likelihood Estimation (MLE), with which the resulting model distribution will be asymptotically consistent with true distribution when the fine-tuning dataset has infinite data samples. But it is not the case of fine-tuning on small datasets, which always leads to the mismatch problem of the real and model distributions.

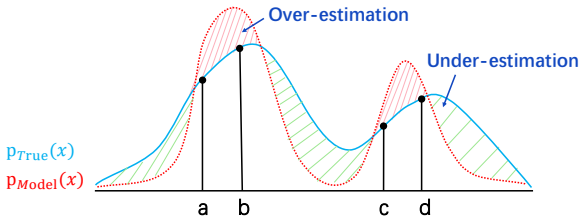


Figure 1: The over- and under-estimation problems of the model distribution. For example, sample **b** represents the simple sentence "Yes .", whose probability is over-estimated. Its model NLL (4.01, negative log-likelihood) is significantly lower than the 95% confidence interval of its real NLL [4.89, 5.37], which is estimated on the training set.

Specifically, MLE minimizes the Kullback–Leibler (KL) divergence between model and true distributions. Theis et al. (2016) point out that minimizing KL avoids assigning an extremely small probability to any data point but assigns a lot of probability mass to non-data regions, which leads to a gap between P_{Real} and P_{Model}. Additionally, simple data patterns in the fine-tuning dataset could be easily memorized and over-estimated. Meanwhile, the complex ones may be under-estimated. The above problem is not severe with adequate data samples, but non-trivial when the size of the fine-tuning dataset is not large enough. (see Figure 1).

To address the over- and under-estimated problem, in this paper, we propose MC-Tailor, which can tailor the resulting density of model distribution by cutting the probability mass of over-estimated zones to under-estimated zones, leading to more realistic model distribution after fine-tuning. Concretely, MC-Tailor consists of two components: a ratio estimator to distinguish over- and under-

estimated regions of model distribution; and an early rejection sampling (ERS) component to tailor (reassign) probability mass and efficiently obtain sampled sentences from the model distribution. Note that the proposed ERS is inspired by Sequential Monte Carlo (SMC, Doucet et al. (2000)), but can avoid the degeneration from SMC, as it directly kills samples rather than performs resampling.

We conduct experiments on various data sets to verify the effectiveness of the proposed MC-Tailor. Empirical results show that MC-Tailor can generate significantly better samples than finetuning, and the resulting model distributions of our model are closer to real data distributions.

2 Pre-Trained Language Model

Language models generally estimate the density of sentences in real context within an autoregressive style:

$$P(x) = \prod_{i=1}^{N} P(x_i|x_{[1:i-1]}), \tag{1}$$

where x is a sentence with length N. Recently, with an extremely large number of parameters, pre-trained language models like GPT-2 (Radford et al., 2019) and Transformer-XL (Dai et al., 2019) have shown great promise in text generation. PLMs are first trained on a huge general domain data set and then fine-tuned on specific domain datasets of different downstream tasks.

Specifically, given a pre-trained GPT2 model, to generate sentences of email domain, we always need to fine-tune the GPT2 on a small set of email domain corpus. Additionally, PLMs have some other important applications. Miao et al. (2019) use fine-tuned language models for constrained text generation. Wolf et al. (2019) fine-tune GPT-2 on a dialog data set to boost the performance of dialog system.

However, as stated in the Introduction, directly fine-tuning the PLM on a small dataset may lead to the mismatch problem, namely the over- and under-estimated problem between the true distribution and the model distribution. In the next section, we propose a new method to alleviate this problem.

3 Proposed MC-Tailor

To mitigate the above shortcomings of finetuning, we propose MC-Tailor, which generates samples from a modified sample distribution. MC-Tailor is composed of a ratio estimator, which detects over- and under-estimate regions of model distributions, and the Early Rejection Sampling algorithm (ERS), which accelerates sampling while ensuring sample quality.

3.1 Ratio Estimator

Ratio estimator is a common technique to measure the gap between two related distributions (Yuxuan et al., 2020). In this work, We apply **ratio estimator** $\gamma(x)$ to estimating $\frac{P_{\text{Model}}(x)}{P_{\text{True}}(x)}$, the probability ratio of sentence x in fine-tuned model distribution $P_{\text{Model}}(x)$ and true distribution $P_{\text{True}}(x)$. To tailor the probability from a finetuned PLM, we cut the probabilities of over-fitting samples. Specifically, when $\gamma(x) > 1$, i.e., the model over-estimates the probability of sample x, we remove x with a probability of $1 - \frac{1}{r(x)}$ to approximate $P_{\text{True}}(x)$. After normalization, probabilities of under-estimated areas will increase correspondingly. The resulting new distribution is $P_{\text{Tailor}} \propto \frac{P_{\text{Model}}(x)}{max(\gamma(x),1)}$. In this work, we try several different structures of ratio estimators.

Convolutional Ratio Estimator. Since ratio estimation shares similar properties with classification problems and convolutional neural networks (CNN) are powerful classifiers, our first thought is to build a CNN-based ratio estimator. To be concrete, we use a two-layer CNN to predict whether x is from true or learned distribution. By training with cross-entropy loss,

$$Softmax(\text{CNN}(x)) \rightarrow \frac{P_{\text{Model}}(x)}{P_{\text{True}}(x) + P_{\text{Model}}(x)}. \tag{2}$$

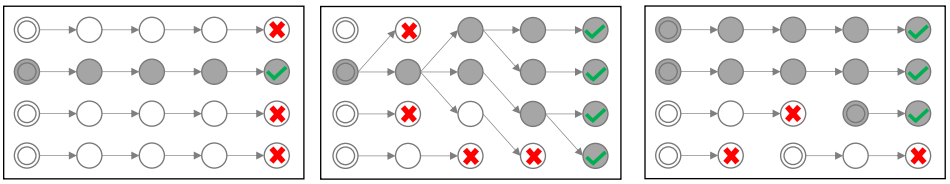
Naturally, we define

$$\gamma(x) = \frac{Softmax(\text{CNN}(x))}{1 - Softmax(\text{CNN}(x))}. \tag{3}$$

Dual Ratio Estimator. Though the basic convolutional ratio estimator is easy to apply, it makes sampling inefficient. For most sentence x, we can roughly predict whether it is in a specific domain or suffering from overfitting by the first a few words. However, $\gamma(x)$ can only be obtained after a full sentence is generated, so massive computing resources are wasted on generating unpromising samples.

To determine whether a prefix $x_{[1:i]}$ is promising, we can estimate

$$\gamma^{'}(\hat{x}_{[1:i]}) = \min_{x_{[1:i]}=\hat{x}_{[1:i]}} (\gamma(x)), \tag{4}$$



(a) RS (b) SMC (c) ERS

Figure 2: Illustration of three sampling algorithms. Concentric circles are newly born particles. Green checkmarks and Red crosses appear when particles are accepted and killed, respectively. Gray circulars represent particles finally accepted while white circulars stand for the opposite.

where $\gamma^{'}(\hat{x}_{[1:i]})$ is the minimum ratio of all sentences with prefix $\hat{x}_{[1:i]}$. If $\gamma^{'}(\hat{x}_{[1:i]})$ is greater than a pre-defined threshold, all sentences with prefix $x_{[1:i]}$ should be rejected. As a result, we do not need to waste time to continue sampling.

But if we directly train $\gamma^{'}(\hat{x}_{[1:i]})$ to distinguish $P_{True}(x_{[1:i]})$ from $P_{Model}(x_{[1:i]})$, we will end up getting the average value of $\gamma(x)$ for all sentences with prefix $x_{[1:i]}$, rather than the minimum value. If so, some sentences with low $\gamma(x)$ will be erroneously rejected. Luckily, the properties of min-max dual sheds some light on this problem. We first define $\gamma^{''}(x) = \max_i(\gamma^{'}(x_{[1:i]}))$ as the dual form of $\gamma^{'}(x)$. Under some weak conditions, we can prove that if $\gamma^{''}(x)$ approximates $\frac{P_{\text{Model}}(x)}{P_{\text{True}}(x)}$, then $\gamma^{'}(\hat{x}_{[1:i]})$ approximates $\min(\gamma(x))$ for x with prefix $x_{[1:i]}$. Similar to training $\gamma(x)$, we train $\gamma^{''}(x)$ by distinguishing $P_{\text{True}}(x)$ from $P_{\text{Model}}(x)$. Since $\gamma^{''}(x)$ is a function of $\gamma^{'}(\hat{x}_{[1:i]})$, we can get a set of proper parameters for $\gamma^{'}(\hat{x}_{[1:i]})$.

Hierarchical Ratio Estimator. Since a single ratio estimator may not be powerful enough to accurately estimate $\frac{P_{\text{Model}}(x)}{P_{\text{Real}}(x)}$, we break down the workload to several $\gamma_i(x)$ in the spirit of boosting. We first train $\gamma_0(x)$ to estimate $\frac{P_{\text{Model}}(x)}{P_{\text{Real}}(x)}$, and get $P^0_{\text{Tailor}}(x)$. And then we use $\gamma_1(x)$ to estimate the gap between P_{Real} and $P^0_{\text{Tailor}}(x)$... With the collaboration of $\gamma_i(x)$, we can get a more accurate $P^n_{\text{Tailor}}(x)$. Using hierarchical ratio estimators also avoids using a single but complicated ratio estimator, which is prone to over-fitting. Similarly, we can add hierarchy to the dual ratio estimator to make a hierarchical dual ratio estimator.

3.2 Efficient Sampling

In this part, we introduce our specially designed Early Rejection Sampling (ERS) algorithm for MC-Tailor. Improved from Sequential Monte Carlo, ERS can efficiently generate samples with high diversity.

Rejection Sampling By applying RS, we first generate a batch of samples from P_{Model}, and then rejecting some samples by rejection ratio $1 - \frac{1}{max(\gamma(x),1)}$. However, RS is very inefficient in actual use since it rejects samples at the end of sampling. As shown in Figure 2a, lots of computation resources are wasted on ultimately rejected samples.

Sequntial Monte Carlo Instead of rejecting samples at the end of sampling, SMC performs resampling at each step. The unnormalized resampling weight at step i is provided by $\frac{\gamma^{'}(x_{[1:i-1]})}{\gamma^{'}(x_{[1:i]})}$, leading to an asymptotically unbiased estimator. However, SMC suffers from serious degeneracy problem. In other words, samples from SMC tend to share a very small number of the ancestors because most of the ancestors are killed during resampling. As a result, sample diversity of SMC is critically low.

Early Rejection Sampling To overcome the degeneracy problem of SMC and increase sample diversity. We propose Early Rejection Sampling (ERS) algorithm. ERS first uniformly samples a real number r in $(0, 1)$. After step i, if $\gamma^{'}(x[1:i]) > \frac{1}{r}$, this particle is killed immediately and computation resources are released to parallel threads. The main difference between ERS and RS is that ERS kills unpromising particles before they are fully generated. But unlike SMC, there is no correlation between SMC samples, resulting in higher sample diversity.

4 Experiments

In this section, We empirically compare the sample quality of our model and baseline models. We first set up experiments and show results in Section 4.2.

Datasets		#Train	Style	Fine-tune	MC-Tailor$_{RS}$	MC-Tailor$_{ERS}$
Ontonotes						
	-bn	12k	Broadcast news	124	117	**111**
	-bc	12k	Broadcast dialog	268	**144**	153
	-mz	7k	Magazine	126	112	**110**
	-nw	35k	Newswire	111	110	**100**
	-tc	13k	Telephone dialog	140	136	**134**
	-wb	17k	Web	166	138	**136**
Switchboard		203k	Formal dialog	198	**165**	169
DailyDialog		76k	Dialy dialog	120	117	**113**
IWSLT-16		133k	Comference speech	240	217	**213**

Table 1: Rev-PPL of each method. All methods start from the same pre-trained GPT2 model. MC-Tailor$_{RS}$ represents single-layer MC-Tailor with rejection sampling and MC-Tailor$_{ERS}$ is a hierarchical MC-Tailor with 3 layers and ERS algorithm. Results of SMC are not reported since it leads to very poor Rev-PPLs because of the lack of sample diversity.

4.1 Experimental Setup

We conduct experiments on 9 data sets with different styles and sizes. And we use five different metrics, including human evaluation, to measure the generation performance of each method.

Datasets. We use the following data sets for experiments.

- **Ontonotes** (Pradhan et al., 2013) is a multi-genre data set for sequence annotation. We use sentences from six genres (bn, bc, mz, nw, tc, wb) for the experiment.
- **Switchboard** (Jurafsky et al., 1997) and **DailyDialog** (Li et al., 2017) are large and medium scale dialog data sets, of which only responses are used for the experiment.
- **IWSLT-16** (Cettolo et al., 2016) is a data set of paired conference speeches for machine translation. We use English sentences from De-En pairs to test model performance on the special conference speech domain.

Evaluation Metrics. To evaluate the generation quality and diversity, we use the following metrics.

- **PPL** reflects the average density of samples from test set in a generative model. Models with lower PPLs have more similar model distributions with real contexts. Unlike baseline models, MC-Tailor only has an unnormalized log-probability. We estimate the normalization constant of MC-Tailor by importance sampling and calculate PPLs directly from the normalized log-probability.
- **Rev-PPL** is a good indicator for both sample quality and diversity, which is derived by first training a language model with generated samples and calculating the PPL of test set in the language model.
- **EMD-l** is the earth mover distance between sentence lengths of real and generated data.
- **EMD-f** is the earth mover distance between word frequencies of real and generated data.
- **Human Evaluation Score** is added to reflect the comprehensive sample quality. We ask 4 volunteers to select a score from $\{0, 0.5, 1\}$ for each sample according to their fluency and coherence with the target style. In 85% cases, at least three volunteers give the same score, showing the reliability of the human evaluation.

Model Details. In all the experiments, we use the released GPT-2 with 117M parameters as the pre-trained language model. We first fine-tune GPT-2 on each dataset and then build our tailor on it. Early-stop is applied to avoid over-fitting. For ratio estimators, we use simple CNNs with two convolution layers where (filter_number, kernel_size) is set to (10,5) and (5,5), respectively.

4.2 Experimental Results

Rev-PPLs of different models are shown in Table 1. We find that MC-Tailor significantly reduces Rev-PPLs than fine-tuning baseline in data sets of different sizes, from Ontonotes-mz with only 7k training samples to relatively large Switchboard data set with more than 200k samples. We also notice that multi-layer MC-Tailor$_{ERS}$ performs better than single-layer MC-Tailor$_{RS}$, which confirms the point in Section 3.2 that the gap between P_{Model} and P_{Data} is too complex for a single-layer ratio estimator to estimate. Sample NLLs of each method (Table 2) further confirms that MC-Tailor succeeds in decreasing the probabilities of over-estimated simple patterns and reallocating them to under-estimated samples.

We further compare MC-Tailor with the baseline

Refs	Sentences	NLL (Fine-tune)	NLL (MC-Tailor$_{\text{ERS}}$)
a	Thank you everyone for watching .	18.03	18.65
b	Yes .	4.01	4.77
c	What does that mean in the context of your book ?	26.56	26.44
d	And it did n't hurt too badly .	23.24	22.97

Table 2: NLL comparison of MC-Tailor$_{\text{ERS}}$ and the baseline on Ontonotes-bc. MC-Tailor$_{\text{ERS}}$ successfully reallocates the probabilities of over-estimated samples (simple sentences such as **a** and **b**) to under-estimated ones (complicated sentences such as **c** and **d**).

Methods	Fine-tune	MC-Tailor$_{\text{ERS}}$
Samples	Right .	She should be charged with rape .
	In the case if you think of this -	And do you still feel that way every day ?
	And the man you say 's the father of the ambassador .	That would only affect Japan 's relations in that region .
	Oh well .	But it would be tough .
	I 've been there n't said anything wrong .	He knew about the attack at the Paris offices .

Table 3: Generated samples of each method on Ontonotes-bc. Samples from MC-Tailor$_{\text{ERS}}$ are more informative and coherent with the target style than the baseline method.

model under other metrics. From table 4, we find MC-Tailor greatly reduce PPL, which means increased probabilities of generating samples similar to test samples. And we can draw the conclusion that sample distributions of MC-Tailor are closer to real sample distributions, with lower EMD-l and EMD-f. What's more, human evaluation scores of MC-Tailor are about 10% higher than fine-tuning, which indicates better sample quality to human eyes. Cases shown in Table 3 further demonstrate the advantage of MC-Tailor in fluency and informativeness. Seq-GAN is also compared in our experiment. However, rev-ppls of GANs are even higher than directly fine-tuning GPT-2, and they are especially difficult to train. So we remove Seq-GAN from baseline models.

The acceleration effect of ERS is also verified in the experiment. For MC-Tailor with 1, 2, and 3 layers of ratio estimator, ERS reduces 30%, 79%, and 90% of computation wasted on unpromising samples, achieving 1.5x, 2.8x, 5x accelerations, respectively.

Data	MCT	PPL	EMD-l	EMD-f	Human
Onto-bn	✗	34.1	4.31	0.57	0.60
	✓	**30.1**	**1.90**	**0.53**	**0.81**
Onto-bc	✗	30.9	6.74	0.67	0.40
	✓	**23.1**	**1.62**	**0.55**	**0.67**
Onto-mz	✗	43.4	5.60	0.69	0.71
	✓	**39.7**	**3.33**	**0.64**	**0.76**
Onto-nw	✗	37.0	4.94	0.61	0.65
	✓	**36.1**	**3.66**	**0.54**	**0.70**
Onto-tc	✗	24.8	4.19	**0.64**	0.54
	✓	**23.8**	**2.46**	**0.64**	**0.54**
Onto-wb	✗	60.9	3.31	0.61	0.46
	✓	**52.8**	**2.40**	**0.51**	**0.60**
SB	✗	19.7	8.75	0.60	0.48
	✓	**18.9**	**5.21**	**0.51**	**0.54**
DD	✗	30.3	5.25	0.47	0.60
	✓	**29.1**	**3.32**	**0.45**	**0.62**
IWSLT	✗	23.3	5.21	0.61	0.32
	✓	**20.9**	**2.99**	**0.55**	**0.40**

Table 4: PPL, EMD-l, EMD-f and human evaluation score of MC-Tailor$_{\text{ERS}}$ with 3 layers and fine-tuning. MCT means whether to use our proposed MC-Tailor or to direct fine-tune. SB and DD represent the Switchboard and DailyDialog data sets, respectively. By one-tail t-tests, we find that improvements in human evaluation scores are significant, with p-values smaller than 0.05.

5 Conclusion

In this paper, we propose MC-Tailor to alleviate the over- and under-estimation problem between true and model distributions. MC-Tailor is composed of a ratio estimator, which adjusts the probabilities of MLE fine-tuned PLMs to approximate true distributions, and the ERS to accelerate sampling while ensuring sample quality. Experiments on various datasets show the effectiveness and efficiency of MC-Tailor.

References

Mauro Cettolo, Niehues Jan, Stüker Sebastian, Luisa Bentivogli, Roldano Cattoni, and Marcello Federico. 2016. The iwslt 2016 evaluation campaign. In *IWSLT*.

Zihang Dai, Zhilin Yang, Yiming Yang, William W Cohen, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. In *Proceedings of ACL*.

Arnaud Doucet, Simon Godsill, and Christophe Andrieu. 2000. On sequential monte carlo sampling

methods for bayesian filtering. *Statistics and computing*, 10(3):197–208.

Daniel Jurafsky, Elizabeth Shriberg, and Debra Biasca. 1997. Switchboard SWBD-DAMSL shallow-discourse-function annotation coders manual, draft 13. Technical report.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. Dailydialog: A manually labelled multi-turn dialogue dataset. In *Proceedings of IJCNLP*.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2019. Cgmh: Constrained sentence generation by metropolis-hastings sampling. In *Proceedings of AAAI*.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using ontonotes. In *Proceedings of CoNLL*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Bakhtiyar Syed, Gaurav Verma, Balaji Vasan Srinivasan, Vasudeva Varma, et al. 2019. Adapting language models for non-parallel author-stylized rewriting. *In arXiv:1909.09962*.

Lucas Theis, Aäron van den Oord, and Matthias Bethge. 2016. A note on the evaluation of generative models. In *Preceedings of ICLR*.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. Transfertransfo: A transfer learning approach for neural network based conversational agents. *In arXiv:1901.08149*.

Song Yuxuan, Miao Ning, Zhou Hao, and Li Lei. 2020. Improving maximum likelihood training for text generation with density ratio estimation. In *Proceedings of AISTATS*.