# Clip-Tuning: Towards Derivative-free Prompt Learning with a Mixture of Rewards

**Yekun Chai Shuohuan Wang**
**Yu Sun Hao Tian Hua Wu Haifeng Wang**
Baidu
{chaiyekun,wangshuohuan}@baidu.com
{sunyu02,tianhao,wu_hua,wanghaifeng}@baidu.com

## Abstract

Derivative-free prompt learning has emerged as a lightweight alternative to prompt tuning, which only requires model inference to optimize the prompts. However, existing work did not take full advantage of the over-parameterized characteristics of large pre-trained language models (PLMs). In this paper, we propose Clip-Tuning, a simple yet effective method that adopts diverse frozen "thinned" networks of PLMs to obtain *a mixture of rewards* and thus advance the derivative-free prompt learning. The thinned networks consist of all the hidden units that survive a stationary dropout strategy, whose inference predictions reflect an ensemble of partial views over prompted training samples. Our method outperforms previous gradient-free prompt learning methods and achieves parity with gradient-based counterparts on seven language understanding benchmarks under few-shot settings.

## 1 Introduction

Extensive research has shown that prompt tuning achieves parity with model tuning (*a.k.a.*, full fine-tuning) in few-shot scenarios (Li and Liang, 2021; Lester et al., 2021). However, prompt tuning that relies on backpropagation from very deep pre-trained transformers requires prohibitive computation and time costs, especially those with billions of parameters (Brown et al., 2020; Rae et al., 2021; Wang et al., 2021; Chowdhery et al., 2022). Meanwhile, for many inference-API-based PLMs, researchers either do not have full access to model weights due to commercial restrictions, or cannot afford the training of enormous parameters, which significantly limits the power of derivative-based tuning. Therefore, derivative-free prompt learning has been a promising alternative (Sun et al., 2022; Diao et al., 2022).

By treating PLMs as black boxes, derivative-free approaches seem to be a feasible solution to harnessing large PLMs. Sun et al. (2022) leveraged evolutionary algorithms to optimize the continuous prompts by repeatedly calling the inference APIs of PLMs. It adopts the model performance over a spot of samples as the optimization feedback. Nevertheless, few-shot demonstrations can only result in sparse reward, preventing the prompts from taking sufficient informative signals. Hence, our goal is to acquire a mixture of diversified rewards for prompt optimization, using colossal PLMs with millions or billions of parameters in few-shot settings.

Recent work on *lottery ticket hypothesis* (Frankle and Carbin, 2019; Chen et al., 2020) states that an over-parameterized PLMs contain matching subnetworks capable of reaching the full test performance comparable to the original model. Then, Kobayashi et al. (2022); Havasi et al. (2021) find that ensembling a mixture of subnetworks can improve the diversity of model predictions and achieve the performance gain. As such, the ensemble of subnetwork predictions is a particularly interesting setting to increase the diversity of learning signals for derivative-free prompt optimization.

Since derivative-free prompt learning only conducts model forward pass without backpropagation, clipping a large proportion of model weights will heavily hurt the overall performance. Srivastava et al. (2014) states that applying dropout to a network amounts to "sampling" a thinned network from it. Moreover, previous work (Gao et al., 2021b; Liang et al., 2021) finds that standard dropout can act as "minimal data augmentation" to construct different sample representations. Therefore, we employ dropout during model inference to "sample" different "thinned" subnetworks and diversify the data representations. Note that our subnetworks are deterministic, while the original dropout is random at each time. For each training example, diverse subnetworks produce a variety of hidden representations and fitness rewards, which diversifies the learning feedback for gradient-free prompt learning.



*Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 108–117

**Contributions** (1) We propose a simple yet effective method Clip-Tuning, in which multiple frozen subnetworks act as multi-view critics and provide a mixture of informative rewards for gradient-free prompt optimization (§4.1). The importance and originality of this study are that it explores the new direction of the exploitation of *reward diversity* in gradient-free prompt optimization. (2) Empirical results show that our method has surpassed previous gradient-free prompt learning approaches on seven natural language understanding (NLU) benchmarks in few-shot settings (§5). Surprisingly, the *random search* method can serve as an excellent few-shot baseline to prime large PLMs. (3) Our method sheds light on inference-only PLMs and can be a good fit for commercial PLM providers to build API-based features. Note that our method requires API providers to support dropout operation, whereas API users do not need to make any change based on derivative-free prompt learning.

## 2 Related work

### 2.1 Prompt-based learning

Holding the promise of exploiting the few-shot learning capability of large pre-trained models, prompt-based learning has attracted extensive attention in recent years (Brown et al., 2020; Schick and Schütze, 2021a; Li and Liang, 2021; Lester et al., 2021; Sun et al., 2022). It primes the frozen PLMs using a series of discrete natural language tokens or continuous "soft prompts" to conduct various downstream tasks. Early work employed exemplar language templates to condition the PLMs for task-specific prediction (Schick and Schütze, 2021b; Scao and Rush, 2021). Such methods require manual involvement of humans in the design of prompt templates, making the continuous prompt a promising direction.

**Prompt tuning** Prompt tuning approaches (Li and Liang, 2021; Lester et al., 2021; Liu et al., 2021) prepend a string of continuous word embeddings as "virtual tokens" to prime the pre-trained models, where it optimizes the continuous prompts with backpropagation while freezing the model weights of PLMs. These methods achieve parity with full model tuning and even surpasses the fine-tuning in few-shot settings.

**Prompt search** There has been a surge of interest in automatic prompt learning, which treats the prompt as a parameter space to be optimized over. One line of prompt search methods focuses on *discrete* search space, *i.e.*, natural language tokens. Shin et al. (2020) employ a gradient-based method to find the optimal trigger words to construct the prompt. Prasad et al. (2022) use a gradient-free edit-based search method to refine the instructional language prompts, in which it produces the optimal edited prompts given manually designed ones. Another line in this direction is *continuous* prompt search, where the prompt is optimized as "virtual tokens" in the continuous parameter space. Sun et al. (2022) adopt the Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES) (Hansen et al., 2003) to search over the intrinsic dimension of prompts (Aghajanyan et al., 2021) with only access to the inference API of PLMs. This approach only requires the forward pass of PLMs without the need for gradient backpropagation. This work builds upon this line of research, targeting better exploiting the over-parameterization of PLMs to collect fine-grained rewards for search algorithms.

### 2.2 Derivative-free optimization

Derivative-free optimization targets the settings that the derivative of the objective is unavailable or unreliable to achieve. It iteratively optimizes the parameter candidate by local hill-climbing in the objective landscape. Suppose the objective function $f : A \to \mathbb{R}$ for some set $A$, derivative-free optimization only uses the input $x$ and its fitness $f(x)$ after evaluation for iterative optimization. Examples include evolutionary strategies (Hansen et al., 2003), Bayesian optimization (Frazier, 2018), random search (Zabinsky et al., 2009), and so forth. In this work, we experimented with CMA-ES (Hansen et al., 2003) and pure random search (Zabinsky et al., 2009) algorithms.

## 3 Derivative-free prompt learning

Vanilla derivative-free prompt learning (Sun et al., 2022) employs the model inference to evaluate the fitness of candidate prompts for iterative prompt learning using evolutionary algorithms. Firstly, it prepends a series of soft prompt embeddings $P$ to the input tokens $X$ to feed the prompted input $[P; X]$ into the frozen pre-trained transformers $f$ parameterized by $\theta$. The prompt $P = P_0 + P_\Delta$ is the summation of randomly initialized or pre-trained prompt $P_0 \in \mathbb{R}^D$ and prompt change $P_\Delta \in \mathbb{R}^D$ that is iteratively optimized by the Covariance Matrix Adaptation Evolutionary Strat-

egy (CMA-ES) (Hansen et al., 2003). Meanwhile, Aghajanyan et al. (2021) find that PLMs have a low dimension reparameterization. The search space of $P_\Delta$ can be reparameterized as a intrinsic dimensionality $\mathbf{z} \in \mathbb{R}^d$ ($d \ll D$) with a randomly initialized affine transformation $\mathbf{W} \in \mathbb{R}^{d \times D}$; that is, $P_\Delta = \mathbf{z} \cdot \mathbf{W}$. Thus, the prompt is optimized over an intrinsic dimensionality of $\mathbf{z}$ instead of $P_\Delta$. Then it performs task-specific inference to get the fitness of candidate prompts using an arbitrary objective function $\mathcal{L}(f_\theta([P;X]), y)$, where $y$ denotes the golden standard label for the input $X$. Afterward, the CMA-ES algorithm iteratively optimizes the prompt fitness to get the optimum prompt that satisfies $\arg\min_P \mathcal{L}(f_\theta([P;X]), y)$.

## 4 Methodology

We propose Clip-Tuning by adopting multiple deterministic clipping instances of PLMs to form diverse views of critics on candidate soft prompts. It has two key properties: (i) Only an instance of frozen model weights is used to acquire diverse reward signals (*i.e.*, fitness) of candidate prompts. Additional budgets include the storage of clipping dropout masks. (ii) Users only require the model forward pass via API without knowing the model internals during prompt learning.

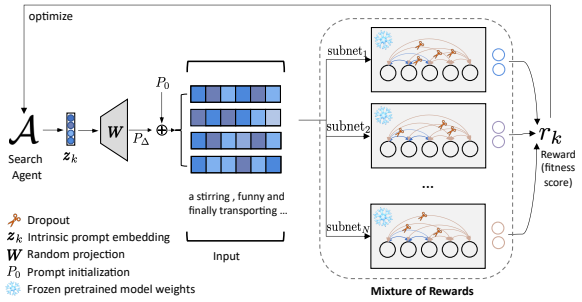


Figure 1: Illustration of Clip-Tuning process.

### 4.1 Dropout as Language Model Clip

As shown in Figure 1 (dashed area), Clip-Tuning employs the dropout clipping multiple ($N$) times to get a suite of subnetworks of frozen PLMs. As aforementioned, we only apply the clipping *in the place of dropout layers during training* to get the subnetworks with minimal costs, making the inference settings the same as training.

For each clipping layer in the $j$-th subnetwork ($j = 1, 2, \cdots, N$), it generates the binary mask $\tilde{\mathbf{M}}_j$ drawn from a Bernoulli distribution with a ratio of probability $p_{\text{clip}}$ and a unique seed $s_j$ generated by a string hash algorithm, *i.e.*, SHA-1 hash function (Burrows, 1995), to keep each instance's clipping mask fixed (see Appendix § A.1 for the pseudocode):

$$\mathbf{M}_j \sim \text{Bernoulli}(p_{\text{clip}}, s_j) \tag{1}$$

As a result, we produce the static binary mask $\tilde{\mathbf{M}}_j$ for target clipping layers of each subnetwork to keep it unique. In this case, each unit is removed with a fixed probability $p_{\text{clip}}$ independent of other units, where $p_{\text{clip}}$ is chosen by the validation set.

After removing a proportion of internal hidden units, the output of each clipped layer is zoomed out to approximately $(1 - p_{\text{clip}})$ of the counterpart of original PLMs. Following the inverted dropout operation (Srivastava et al., 2014), we apply the rescaling by dividing the survived units of clipped layers in each subnetwork by a factor of $(1 - p_{\text{clip}})$ to keep the output expectation unchanged.

Therefore, given the original units $\mathbf{o}$ and $j$-th clipping mask $\mathbf{M}_j$, the resultant units can be :

$$\mathbf{o}_j^{'} = \frac{\mathbf{o}}{1 - p_{\text{clip}}} \odot \mathbf{M}_j \tag{2}$$

where the output units should be rescaled to as per normal to make an inference.

### 4.2 Prompt learning with Clip-Tuning

As shown in Algorithm 1, Clip-Tuning renders diverse predictions from a population of subnetworks, which enhances the diversity of prompt learning rewards. For each subnetwork, we utilize a randomly initialized linear layer on top of "[CLS]" embeddings for class prediction. Given the subnetwork prediction $f_\theta(\cdot)$ and golden labels $y$, we then use the evaluation function $\mathcal{L}$ to compute the fitness score for each subnetwork. With multiple subnetworks, the overall prompt fitness $r_k$ is computed as the average from different subnetworks for the use of prompt optimization. The prompt learning agent $\mathcal{A}$ can take various derivative-free algorithms. Following Sun et al. (2022), we use the CMA-ES algorithm (Hansen et al., 2003) for prompt learning and leave others for future work.

## 5 Experiments and results

**Datasets and evaluation metrics** We conduct experiments on seven language understanding datasets, including SST-2 (Socher et al., 2013), Yelp polarity, AG's News, DBPedia (Zhang et al., 2015), SNLI (Bowman et al., 2015), RTE (Wang et al., 2018), and MRPC (Dolan and Brockett,

**Algorithm 1** Derivative-free prompt learning with Clip-Tuning.

**Require:** Clipping mask set $\mathcal{M} = \{m_1, m_2, ..., m_N\}$; frozen subnetwork tribe $\{f_\theta(\cdot|m)|\forall m \in \mathcal{M}\}$; few-shot samples $X$ and labels $y$; the fitness evaluation function $\mathcal{L}(\cdot)$; the maximum update steps $K$, initialized prompt embedding $P_0 \in \mathbb{R}^D$, randomly initialized affine transformation $\mathbf{W} \in \mathbb{R}^{d \times D}$, search agent $\mathcal{A}$ with derivative-free algorithms.
**for** $k = \{1, 2, \cdots, K\}$ steps **do**
  Sample an intrinsic embedding of candidate prompt $\mathbf{z}_k \in \mathbb{R}^d$ from the agent $\mathcal{A}$;
  Get prompt embeddings:
  $$P_k = P_0 + P_\Delta = P_0 + \mathbf{z}_k \cdot \mathbf{W}$$
  Evaluate the fitness $r_k$ of candidate prompts $P_k$ using subnetworks ▷ **Clip-Tuning**
  $$r_k = \mathbb{E}_{m \sim \mathcal{M}} \mathcal{L}\big(f_\theta([P_k; X]|m), y\big)$$
  Optimize the search agent $\mathcal{A}$ using $r_k$.
**end for**

2005). We randomly draw 16 samples for each class as training and evaluation sets, respectively, to construct true few-shot learning. All datasets except MRPC adopt accuracy for evaluation, whereas MRPC uses F1-measure. Appendix §A.2 reports the detailed data statistics.

**Models and optimization** We follow the same baseline setting as Sun et al. (2022) for few-shot learning. We utilize RoBERTa$_{\text{Large}}$ (Liu et al., 2019) in all experiments. *Derivative-based methods* include prompt tuning (Lester et al., 2021), adapter tuning (Houlsby et al., 2019), Low-Rank Adaptation (LoRA; Hu et al., 2022), full model tuning, and feature-based tuning (Peters et al., 2019) that uses a trainable bi-LSTM or linear classifier on the top for prediction. *Derivative-free methods* consist of a manual prompt that uses manually designed templates for zero-shot evaluation; in-context learning (Brown et al., 2020); random search that randomly samples a large set of search points then selects the optimal after evaluation; and Black-Box Tuning (Sun et al., 2022) that harnesses the CMA-ES algorithm for derivative-free search. *Our method* takes the hinge loss as $\mathcal{L}$ for reward computation, and a clipping ratio of 0.1. Following Sun et al. (2022), we randomly draw tokens from RoBERTa vocabularies as prompt initialization $P_0$ for text classification tasks while adopting prompt embedding pretrained on MNLI (Williams et al., 2018) for two-sentence tasks. Appendix §A.3 lists the optimization details.

**Main results** Table 1 compares the test-set performance of baselines in few-shot settings[1]. Our method outperforms the previous prompt learning method (Black-Box Tuning) on seven benchmarks. The improvement on three sentence-pair classification datasets (+1.77% on average) outperforms the counterpart on four text classification datasets (+0.93%). It is observed that our method confers amenable benefits using both randomly initialized prompts and pre-trained prompts, which achieves more considerable performance gain using randomly initialized prompts. Interestingly, the *pure random search* can serve as a good baseline for few-shot comparison, even achieving competitive results on MRPC.

**Ablation study** To examine the benefits of proposed method, we conduct the ablation test by removing the Clip-Tuning, as shown in Table 2. The ablation test results suggest that our method provides more informative feedback for sentence-pair tasks than single-sentence classification. Moreover, the performance gains grow when the number of categories becomes small.

**Varying clip ratios** Figure 2 illustrates the test-set performance versus various clip ratios. The performance on SST-2 peaks at the clipping ratio ($p_{\text{clip}}$) of 0.1, which is the same as the RoBERTa dropout rate. The performance curve of MRPC shows a saddle trend: it first peaks at 0.1 and then rebounds to the top at 0.5. We conjecture that the pre-trained model weights fit in with the dropout during pre-training, thereby performing smoothly with the identical clip rate.

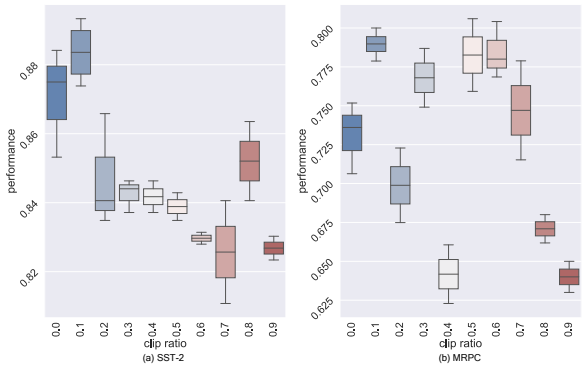


Figure 2: Varying clip ratios vs. performance.

[1]Note that the reproduced DBPedia results of Black-Box Tuning are different from (Sun et al., 2022). We conjecture it is due to experimental factors such as random seeds.

| | Tunable params | SST-2 | Yelp Polarity | AG's News | DBPedia | MRPC | SNLI | RTE | Avg. |
|---|---|---|---|---|---|---|---|---|---|
| Discrete prompts | | | | | | | | | |
| Manual Prompt | 0 | 79.82 | 89.65 | 76.96 | 41.33 | 67.40 | 31.11 | 51.62 | 62.56 |
| In-Context Learning | 0 | 79.79±3.06 | 85.38±3.92 | 62.21±13.46 | 34.83±7.59 | 45.81±6.67 | 47.11±0.63 | 60.36±1.56 | 59.36 |
| Derivative-based optimization | | | | | | | | | |
| (Full) Model Tuning | 355M | 85.39±2.84 | 91.82±0.79 | **86.36**±1.85 | 97.98±0.14 | 77.35±5.70 | 54.64±5.29 | 58.60±6.21 | 78.88 |
| Feature-based (MLP) | 1M | 64.80±1.78 | 79.20±2.26 | 70.77±0.67 | 87.78±0.61 | 68.40±0.86 | 42.01±0.33 | 53.43±1.57 | 66.63 |
| Feature-based (LSTM) | 17M | 65.95±0.99 | 74.68±0.10 | 77.28±2.83 | 90.37±3.10 | 71.55±7.10 | 46.02±0.38 | 52.17±0.25 | 68.29 |
| Prompt Tuning | 50K | 68.23±3.78 | 61.02±6.65 | 84.81±0.66 | 87.75±1.48 | 51.61±8.67 | 36.13±1.51 | 54.69±3.79 | 64.07 |
| w/ pre-trained prompt | 50K | - | - | - | - | 77.48±4.85 | 64.55±2.43 | 77.13±0.83 | 74.42 |
| Adapter Tuning | 2.4M | 83.91±2.90 | 90.99±2.86 | 86.01±2.18 | **97.99**±0.07 | 69.20±3.58 | 57.46±6.63 | 48.62±4.74 | 76.31 |
| LoRA | 786K | 88.49±2.90 | 90.21±4.00 | 87.09±0.85 | 97.86±0.17 | 72.14±2.23 | 61.03±8.55 | 49.22±5.12 | 78.01 |
| Derivative-free optimization | | | | | | | | | |
| Black-Box Tuning | 500 | 89.34±1.07 | 91.26±0.40 | 82.04±0.63 | 79.43±0.02 | 61.56±4.34 | 46.58±1.33 | 52.59±2.21 | 71.83 |
| w/ pre-trained prompt | 500 | - | - | - | - | 75.58±1.63 | 81.02±1.39 | 77.62±1.65 | 82.33 |
| Random Search | 500 | 86.35 | 89.92 | 79.96 | 37.15 | 76.03 | 40.08 | 51.63 | 65.87 |
| Clip-Tuning (ours) | 500 | **90.44**±0.98 | **92.15**±0.43 | 82.81±0.12 | 80.38±0.03 | 75.12 | 50.62 | 53.43 | 74.99 |
| w/ pre-trained prompt | 500 | - | - | - | - | **77.78**±1.22 | **83.50**±1.38 | **78.25**±1.54 | **83.62** |

Table 1: Overall *16-shot* performance of gradient-based or gradient-free methods on seven NLU benchmarks.

| Dataset | Task | #class | w/ Clip-Tuning | w/o Clip-Tuning | Δperf |
|---|---|---|---|---|---|
| SST-2 | SA | 2 | 90.14 | 88.65 | -1.49 |
| Yelp Polarity | SA | 2 | 92.36 | 91.16 | -1.20 |
| AG's News | TC | 4 | 82.66 | 81.72 | -0.94 |
| DBPedia | TC | 14 | 80.80 | 79.43 | -1.34 |
| MRPC | PC | 2 | 78.69 | 76.56 | -2.09 |
| SNLI | NLI | 3 | 83.16 | 80.84 | -2.32 |
| RTE | NLI | 2 | 79.78 | 75.45 | -4.33 |

Table 2: Ablation test performance. SA: sentiment analysis; TC: topic classification; PC: paraphrase identification; NLI: natural language inference.

**Varying thinned subnetworks ($N$)** Figure 3(a) reveals that there has been a fluctuated increment as the number of thinned subnetworks increases. More subnetworks can deliver more fine-grained learning rewards but may be impeded by the numerous inference costs. We thus take the first peak ($N \approx 5$) for economic consideration.

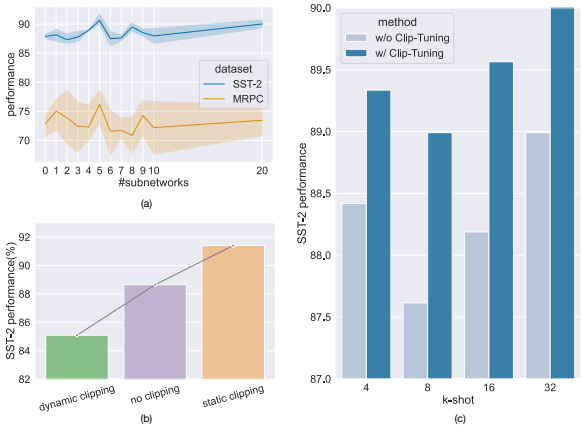


Figure 3: (a) Varying subnetworks ($N$) vs. performance. (b) Dynamic / no / static clipping vs. performance. (c) Size of training samples (SST-2).

**Static vs. dynamic clipping** Our method applies the static clipping for each thinned subnetwork. Figure 3(b) compares the dynamic clipping with random dropout masks every time, static clipping (ours), and no clipping on SST-2. It is obvious that static clipping confers benefits whereas dynamic clipping hurts. This is because dynamic clipping results in unexpected noise and thereby misdirecting the search direction during prompt learning.

**Size of training data** Figure 3(c) exhibits the effect of our method on various training data sizes: Clip-Tuning consistently improves upon the original soft prompts. There has been a generally stable performance gain on prompt learning after applying the Clip-Tuning approach.

**Discussion** Our method improves from the API provider side, which requires trivial modifications of API services (*i.e.*, supporting dropout), while it still remains a black box for API users. It can be easily scaled for deployment in a distributed manner. Each serving PLMs only needs to support the dropout operation to form the subnetwork and return the score over few-shot samples.

## 6 Conclusion and future work

We propose Clip-Tuning, a simple method to enhance derivative-free prompt learning for NLU tasks. It focuses on acquiring fine-grained informative feedback for derivative-free optimization, beating the previous rivals on seven NLU benchmarks. Our method sheds light on the exploration of inference-only API-based PLMs. In the future, it is a promising direction to reduce the cost of green derivative-free prompt optimization.

## Limitations

Although our method optimizes over a small size ($d = 500$) of intrinsic dimensionality, it evaluates the fitness of training samples with multiple subnetworks, costing extra computation resources. Besides, derivative-free prompt learning is constrained by the limited intrinsic dimensionality, grounding on the findings of large PLMs (Aghajanyan et al., 2021). Further, we empirically find it more suitable for NLU tasks with a small class number. However, it is still a good fit for exploiting the API-based tuning by only applying the model inference.

## Acknowledgements

The authors thank all anonymous reviewers for their insightful and constructive comments.

## References

Armen Aghajanyan, Luke Zettlemoyer, and Sonal Gupta. 2021. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *ACL*.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *ArXiv*, abs/2005.14165.

James H Burrows. 1995. Secure hash standard. Technical report, Department of Commerce Washington DC.

Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. 2020. The lottery ticket hypothesis for pre-trained bert networks. *NeurIPS*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek B Rao, Parker Barnes, Yi Tay, Noam M. Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Benton C. Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier García, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Díaz, Orhan Firat, Michele Catasta, Jason Wei, Kathleen S. Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways. *ArXiv*, abs/2204.02311.

Shizhe Diao, Xuechun Li, Yong Lin, Zhichao Huang, and Tong Zhang. 2022. Black-box prompt learning for pre-trained language models. *ArXiv*, abs/2201.08531.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *IJCNLP*.

Jonathan Frankle and Michael Carbin. 2019. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv: Learning*.

Peter I Frazier. 2018. A tutorial on bayesian optimization. *arXiv preprint arXiv:1807.02811*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021a. Making pre-trained language models better few-shot learners. *ArXiv*, abs/2012.15723.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021b. Simcse: Simple contrastive learning of sentence embeddings. *ArXiv*, abs/2104.08821.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2021. Ppt: Pre-trained prompt tuning for few-shot learning. *ArXiv*, abs/2109.04332.

Nikolaus Hansen, Sibylle D. Müller, and Petros Koumoutsakos. 2003. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). *Evolutionary Computation*, 11:1–18.

Marton Havasi, Rodolphe Jenatton, Stanislav Fort, Jeremiah Zhe Liu, Jasper Snoek, Balaji Lakshminarayanan, Andrew M. Dai, and Dustin Tran. 2021. Training independent subnetworks for robust prediction. *ICLR*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. *ArXiv*, abs/2106.09685.

Sosuke Kobayashi, Shun Kiyono, Jun Suzuki, and Kentaro Inui. 2022. Diverse lottery tickets boost ensemble from a single pretrained model.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *ArXiv*, abs/2104.08691.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, abs/2101.00190.

Xiaobo Liang, Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, and Tie-Yan Liu. 2021. R-drop: Regularized dropout for neural networks. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 10890–10905.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. Gpt understands, too. *arXiv preprint arXiv:2103.10385*.

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*, abs/1907.11692.

Matthew E. Peters, Sebastian Ruder, and Noah A. Smith. 2019. To tune or not to tune? adapting pretrained representations to diverse tasks. *ArXiv*, abs/1903.05987.

Archiki Prasad, Peter Hase, Xiang Zhou, and Mohit Bansal. 2022. Grips: Gradient-free, edit-based instruction search for prompting large language models. *arXiv preprint arXiv:2203.07281*.

Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche, Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, Amelia Glaese, Johannes Welbl, Sumanth Dathathri, Saffron Huang, Jonathan Uesato, John F. J. Mellor, Irina Higgins, Antonia Creswell, Nathan McAleese, Amy Wu, Erich Elsen, Siddhant M. Jayakumar, Elena Buchatskaya, David Budden, Esme Sutherland, Karen Simonyan, Michela Paganini, L. Sifre, Lena Martens, Xiang Lorraine Li, Adhiguna Kuncoro, Aida Nematzadeh, Elena Gribovskaya, Domenic Donato, Angeliki Lazaridou, Arthur Mensch, Jean-Baptiste Lespiau, Maria Tsimpoukelli, N. K. Grigorev, Doug Fritz, Thibault Sottiaux, Mantas Pajarskas, Tobias Pohlen, Zhitao Gong, Daniel Toyama, Cyprien de Masson d'Autume, Yujia Li, Tayfun Terzi, Vladimir Mikulik, Igor Babuschkin, Aidan Clark, Diego de Las Casas, Aurelia Guy, Chris Jones, James Bradbury, Matthew G. Johnson, Blake A. Hechtman, Laura Weidinger, Iason Gabriel, William S. Isaac, Edward Lockhart, Simon Osindero, Laura Rimell, Chris Dyer, Oriol Vinyals, Kareem W. Ayoub, Jeff Stanway, L. L. Bennett, Demis Hassabis, Koray Kavukcuoglu, and Geoffrey Irving. 2021. Scaling language models: Methods, analysis & insights from training gopher. *ArXiv*, abs/2112.11446.

Teven Le Scao and Alexander M Rush. 2021. How many data points is a prompt worth? *arXiv preprint arXiv:2103.08493*.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *EACL*.

Timo Schick and Hinrich Schütze. 2021b. Few-shot text generation with natural language instructions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 390–402.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, A. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15:1929–1958.

Tianxiang Sun, Yunfan Shao, Hong Qian, Xuanjing Huang, and Xipeng Qiu. 2022. Black-box tuning for language-model-as-a-service. *ArXiv*, abs/2201.03514.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *ArXiv*, abs/1804.07461.

Shuohuan Wang, Yu Sun, Yang Xiang, Zhihua Wu, Siyu Ding, Weibao Gong, Shi Feng, Junyuan Shang, Yanbin Zhao, Chao Pang, Jiaxiang Liu, Xuyi Chen, Yuxiang Lu, Weixin Liu, Xi Wang, Yangfan Bai, Qiuliang Chen, Li Zhao, Shiyong Li, Peng Sun, Dianhai Yu, Yanjun Ma, Hao Tian, Hua Wu, Tian Wu, Wei Zeng, Ge Li, Wen Gao, and Haifeng Wang. 2021. Ernie 3.0 titan: Exploring larger-scale knowledge enhanced pre-training for language understanding and generation. *ArXiv*, abs/2112.12731.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *NAACL*.

Zelda B Zabinsky et al. 2009. Random search algorithms. *Department of Industrial and Systems Engineering, University of Washington, USA*.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav Artzi. 2021. Revisiting few-sample bert fine-tuning. *ArXiv*, abs/2006.05987.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *ArXiv*, abs/1509.01626.

# A Appendices

## A.1 Pesudocode

Algorithm 2 presents the pseudocode to get fixed subnetworks. It is simple to apply the clipping operation in the place of dropout layer in PLMs with few lines.

**Algorithm 2** Fixed clipping to get a single deterministic subnetwork.

1: Define a unique string and convert to unique integer seed $s$ using string hash algorithm, *i.e.*, SHA-1 algorithm, for each clipping layer;
2: Generate a unique clipping mask according to Eq. (1);
3: Apply the clipping operation as Eq. (2) to obtain the unique "thinned" subnetwork;
4: Store the unique string $s$ (rather than subnetwork weights) for reuse and memory-saving.

## A.2 Data statistics

Table 3 summarizes the statistics of natural language understanding benchmark datasets for evaluation. We randomly sample 16 samples for the training and evaluation set to construct the true few-shot training and evaluation set, respectively. Following (Zhang et al., 2021; Gao et al., 2021a; Gu et al., 2021; Sun et al., 2022), we employ original evaluation set as the test set. We use the test set for those without a development set to evaluate the performance.

| Data | NLU Task | #train | #test | #class |
|---|---|---|---|---|
| SST-2 | Sentiment analysis | 67k | 0.9k | 2 |
| Yelp polarity | Sentiment analysis | 560k | 38k | 2 |
| AG's News | Topic classification | 120k | 7.6k | 4 |
| DBPedia | Topic classification | 560k | 70k | 14 |
| MRPC | Paraphrase classification | 3.7k | 0.4k | 2 |
| SNLI | Language inference | 549k | 9.8k | 3 |
| RTE | Language inference | 2.5k | 0.3k | 2 |

Table 3: Summary of data statistics.

## A.3 Training details

We summarize the training settings of baseline models as follows: (1) Prompt Tuning. Following (Lester et al., 2021), only the soft prompts prepended to the input sequences are trained while freezing the PLM model weights. The Adam optimizer with a learning rate of 5e-4, and a batch size of 16 for 1,000 epochs are adopted. (2) Model Tuning. It uses the Adam optimizer with a learning rate of 1e-5 and batch size 16 for 200 epochs. (3) Feature-based Tuning (Peters et al., 2019). The weights of PLMs are frozen, and only the BiLSTM or MLP layer on top of `[CLS]` are trained. Both settings apply the Adam optimizer with a learning rate of 3e-4, batch size of 16, and 1,000 training epochs on few-shot data. (4) Adapter tuning (Houlsby et al., 2019). The weights of PLMs are frozen, Adapter injects the parameter-efficient adapter module after each transformer sub-layer, but before the skip connection. In our experiment with RoBERTa$_{\text{large}}$, it only tunes around 2.4M extra parameters in total. (5) Manual prompt. The following four manually designed prompts are used for zero-shot evaluation:

1. "<sentence>. It was [MASK]."
2. "[MASK] news: <sentence>"
3. "[Category: [MASK]] <sentence>"
4. "<hypothesis>? [MASK], <premise>"

In which SST-2 and Yelp Polarity datasets adopt template 1, AG's News uses template 2, DBPedia uses template 3, and all two-sentence classification datasets including MRPC, RTE, and SNLI apply template 4.

(6) In-context learning (Brown et al., 2020). Randomly selected 32 training samples are concatenated with the input text for GPT-3 like in-context learning. (7) Random Search. We apply the random search method on the intrinsic prompt embedding $\mathbf{z}_k \in \mathbb{R}^d (d = 500)$ for 1,000 steps. It samples a large number of search points and selects the optimal one based on their fitness. (8) Black-Box Tuning. We follow the original experimental settings in (Sun et al., 2022) which adopt the CMA-ES algorithm for black-box search.

For a fair comparison, all prompt-based methods apply the same prompt length, manual template, and initialized prompt embeddings $P_0$. All experiments are run over multiple random seeds on 4 Nvidia A100 chips. It is worth noting that different subnetworks will not retard the training time since we parallelly evaluate different subnetworks.

For Clip-Tuning methods, we adopt the following hyperparameters setting as shown in Table 4. We adopt the same CMA-ES algorithm settings in Sun et al. (2022), *i.e.*, intrinsic prompt dimension $d = 500$, evolutionary population size of 20. The margin of hinge loss $\mathcal{L}$ is set to 2.

| Hyperparameter | Value |
|---|---|
| Intrinsic dimension ($\|z_k\|$) | 500 |
| Prompt length | 50 |
| Clipping ratio ($p_{\text{clip}}$) | 0.1 |
| Number of subnetworks ($N$) | 5 |
| Search algorithm ($\mathcal{A}$) | CMA-ES |
| Search iteration $k$ | 10k, except 20k for DBPedia |

Table 4: Hyperparameter settings of our experiments.

### A.4 Choices of fitness evaluation function $\mathcal{L}$

To verify the effect of the fitness evaluation function, we design experiments with a cross-entropy function on four types of NLU tasks, one dataset for each task (see Table 3 for statistics). Table 5 shows that Clip-Tuning can also achieve performance gains with cross-entropy functions on SNLI tasks but degrades on other three datasets. However, the improvement consistency is worse than hinge loss. This may be because that hinge loss is less prone to outliers in training samples, making the optimization process more robust.

| Fitness function $\mathcal{L}$ | SST-2 | AG's News | MRPC | SNLI |
|---|---|---|---|---|
| ce | 88.87 | 83.49 | 73.72 | 80.38 |
| ce + ours | 88.20 | 82.82 | 70.60 | 83.08↑ |
| hinge | 87.04 | 81.73 | 72.18 | 80.84 |
| hinge+ours | 90.44↑ | 82.81↑ | 72.46↑ | 83.16↑ |

Table 5: Fitness evaluation function comparison. Cross-entropy (ce) vs. hinge loss. Experiments are run over multiple random seeds.

### A.5 Random vs. pre-trained prompt initialization ($P_0$)

To testify the generalized effect of our method using various prompt initialization on sentence-pair tasks, we conduct further experiments as shown in Figure 4. It is observed that our method confers amenable benefits using both initialization strategies on three benchmarks. Meanwhile, our method achieves more considerable performance gain using randomly initialized prompts than pre-trained prompts.

### A.6 Does the number of class affect?

We find that derivative-free prompt learning outperforms derivative-based prompt tuning when the class number of datasets becomes small. In contrast, gradient-based methods achieve better when classifying many categories. We conjecture that this is because datasets with a large number of categories, such as DBPedia (14 classes), can give rise

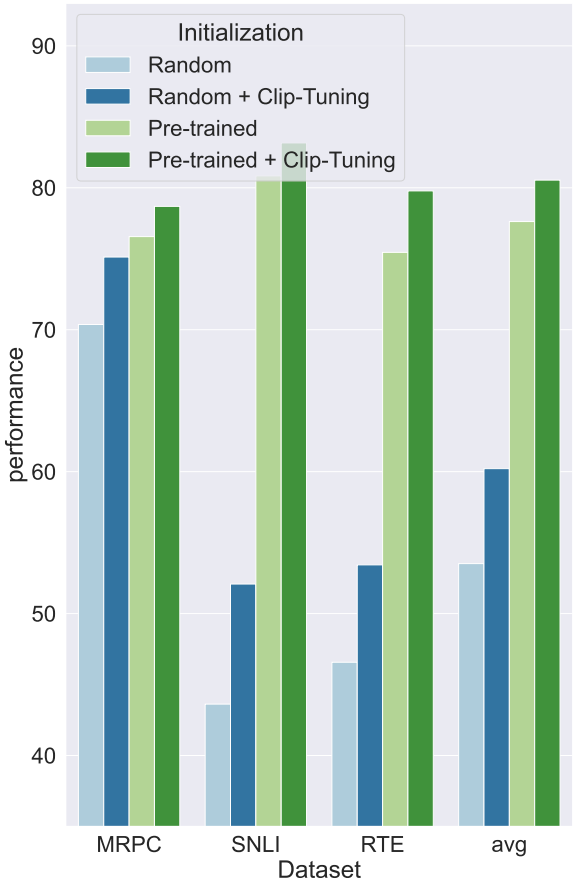


Figure 4: Different prompt initialization vs. performance (SST-2).

to a higher dimensional search space compared to the counterparts of only two classes, enlarging the range of search space. This still remains an open question for future work.