Enhancing Language Model Hypernetworks with Restart: A Study on Optimization

Yihan Zhang[1], **Jie Fu**[2], **Rongrong Ji**[3], **Jie Chen**[1,4*]
[1] School of Electronic and Computer Engineering, Peking University, Shenzhen, China
[2] Shanghai AI Lab [3] Xiamen University
[4] Pengcheng Laboratory, Shenzhen, China
ariszhang@stu.pku.edu.cn, fujie@pjlab.org.cn,
rrji@xmu.edu.cn, chenj@pcl.ac.cn

Abstract

Hypernetworks are a class of meta-networks that generate weights for main neural networks. Their unique parameter spaces necessitate exploring suitable optimization strategies to enhance performance, especially for language models. However, a comprehensive investigation into optimization strategies for hypernetworks remains absent. To address this gap, we analyze the loss landscape of hypernetworks and propose that restart optimization strategies can improve their performance for language models. We find that hypernetworks have inherently more complicated loss landscapes compared to conventional networks due to their distinct parameter spaces. Consequently, a restart strategy that periodically resets the learning rate can facilitate better convergence for hypernetworks. Through experiments on instruction tuning and multi-task training, we demonstrate that the restart strategy consistently enhances the performance of hypernetworks for language models, often more effectively than for conventional deep neural networks. Our findings highlight the importance of tailored optimization techniques to unlock the full potential of hypernetworks in natural language processing tasks[1].

1 Introduction

In recent research, large language models (LLMs) have been proven to be useful tools for learning complicated representations. However, LLMs have limitations regarding their weights and architecture, typically fixed after training. Hypernetworks, proposed by Ha et al. (2022), use one small neural network to generate the weights for the main network, which attracts great attention for their potential to address this gap. Hypernetworks create a new parameter space distinct from the one created by the main network. Given the significance of model training strategies in the overall training process, it is highly advantageous to investigate optimization techniques within this new parameter space. The existing hypernetworks usually adopt an optimization strategy similar to the standard training of a general LLMs (Ye and Ren, 2021; Ivison and Peters, 2022). This means that they do not specifically conduct comparative research on different optimizers and schedulers during training, which we believe finally prevents hypernetworks from being used to their full potential.

To address the question above, we conduct a study on the optimization of hypernetworks for language models (LMs). When considering the optimization of the neural networks, their loss landscapes depict the direction of optimization. This makes loss landscape important when we try to bridge this gap between conventional LMs and hypernetworks. The restart strategy (Loshchilov and Hutter, 2022), by cyclically resetting the learning rate to the initial state during training, can effectively navigate the challenges of ill-conditioned loss functions and saddle points. Particularly for hypernetworks with unique parameter spaces, these simple meta networks' loss landscapes tend to be

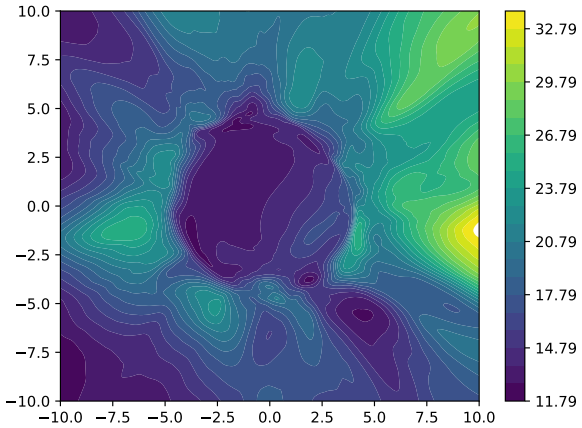


Figure 1: Rugged loss landscapes of Hypernetwork

*Corresponding Author.
[1]Code is publicly available for research purposes: https://github.com/Aris-z/HyperRestart

Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 7826–7838

more rugged compared to the main network, with more local minima and saddle points, as shown in Figure 1. This makes traditional optimization methods prone to getting stuck in these areas and unable to proceed further. The restart strategy introduces a certain perturbation to the model and allows it to escape from the blocked regions and more easily reach the global optimum.

In this work, we conduct a series of experiments to investigate the influence of the restart strategy in hypernetworks for language models. Our investigation also includes an analysis of minima and saddle points, which provides a foundation for the application of the restart strategy in this context. We conduct a series of experiments to assess the robustness and generalizability of the restart strategy when applied to hypernetworks for LMs. As a result, we reveal that the restart strategy enhances the performance of hypernetworks when used for LMs. Notably, this effect is often more obvious in hypernetworks than in conventional LMs.

In summary, our contributions are as follows:

- We conduct an empirical study of the loss landscape of hypernetworks and conventional LLMs, concentrating on their structural differences and point out the similarities and differences between hypernetwork and conventional network.
- We explore the impact of restart phases in the training of hypernetworks, with a special emphasis on their role in instruction tuning for LLMs.
- We provide a comprehensive set of experimental results across diverse datasets and hypernetwork architectures, demonstrating the effectiveness and generalization of the restart strategy in enhancing hypernetwork performance.

2 Related Work

2.1 Restart in Optimization

To enhance optimization in deep neural networks, restart techniques are increasingly used to navigate complicated loss landscapes and avoid local minima. Adaptive warm restart (O'donoghue and Candes, 2015) initiates restarts based on the objective function's behavior or the gradient's orientation. The learning rate's role is pivotal in this context, as it dictates the convergence speed towards the global minimum (Mishra and Sarawadekar, 2019).

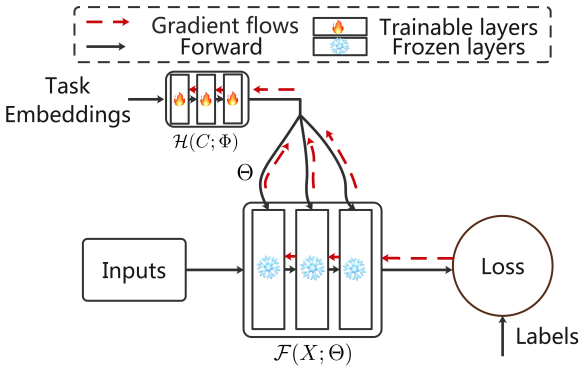


Figure 2: Illustration of general hypernetworks. $\mathcal{F}(X;\Theta)$ is the main network that generates the outputs corresponding to different inputs X. $\mathcal{H}(C;\Phi)$ is the hypernetwork whose outputs Θ are the weights or other parameters of the main network. C is a learnable task embeddings and Φ represents the hypernetworks' parameters.

Smith (2017) introduced the cyclical learning rate (CLR) policy, which varies the learning rate within defined boundaries. Loshchilov and Hutter (2022) adapted restart methods to stochastic gradient descent (SGD), implementing a cosine annealing schedule to periodically reset the learning rate.

The main obstacle in loss function optimization is often saddle points characterized by small gradients and can decelerate learning (Dauphin et al., 2014; Choromanska et al., 2015). Restart strategies have proven effective in traversing these regions more quickly (Dauphin et al., 2015). Large language models (LLMs) employ learning rate schedules that include warmup and decay phases during pre-training (Zhao et al., 2023), and recent approaches suggest that resetting the learning rate during continual pre-training can further enhance performance (Gupta et al., 2023). These strategies are crucial for optimizing the learning process in complicated loss landscapes.

2.2 Hypernetworks

Hypernetworks (Ha et al., 2022) are a novel class of neural networks that generate the parameters for a main deep neural network (DNN). Instead of learning parameters of a main DNN directly, the hypernetworks produce these weights (Chauhan et al., 2023a). This paradigm has given rise to hyperDNNs, which encompass DNNs utilizing hypernetworks for various applications (Mahabadi et al., 2021; Ivison and Peters, 2022; Volk et al., 2022; Xiao et al., 2023; Ye and Ren, 2021).

Hypernetworks are characterized by their ability to facilitate soft weight sharing across multiple tasks, enhancing multi-task learning and transfer learning capabilities (Ye and Ren, 2021; von Oswald et al., 2020; Chauhan et al., 2023b). They also enable the creation of data-adaptive DNNs, where the hypernetwork tailors the main network's weights to the specific data at hand (Ye and Ren, 2021). Furthermore, hypernetworks often have fewer weights than traditional DNNs, offering a form of parameter efficiency that is particularly advantageous in resource-constrained scenarios (Zhao et al., 2020; Ivison and Peters, 2022).

Recent advances in parameter-efficient tuning for NLP models, despite their effectiveness, are often limited to low data scenarios such as zero-shot or few-shot. Ivison et al. (2023) address this by introducing Hypernetworks for Instruction Tuning (HINT). In parallel, He et al. (2022) propose HyperPrompt that leverages a hypernetwork as a global memory for query attention. These hypernetwork-based approaches promise to deliver new task adaptability with controlled computational and parameter overhead. Because of the general application of hypernetworks, investigating how to unlock the potential of hypernetworks is important. Considering that the structure of hypernetworks may be different from that of ordinary DNNs, we hope to find better optimization methods from their differences.

3 Preliminaries

In the following part, we refer to the hypernetwork and the corresponding main network whose parameters are not generated by the hypernetwork but are learnable, when we analyze the difference in loss landscape between the hypernetwork and the main network.

3.1 Landscape of Hypernetworks

For a pair of input data (X, Y), we denote the common language model as $\mathcal{F}$, where the model forward is represented by $\mathcal{F}(X;\Theta) = Y$, and Θ is the set of weights updated via backpropagation. However, in hypernetwork language models (hyperLMs), there is a hypernetwork $\mathcal{H}(C;\Phi)$ to generate the weights Θ of the main network $\mathcal{F}$, where C is the input to the hypernetwork. The optimization problems for the hyperLM can be simply written as given below:

$$\begin{aligned} &\min_{\Theta} \mathcal{L}(\mathcal{F}(X;\Theta), Y) \\ \rightarrow &\min_{\Phi} \mathcal{L}(\mathcal{F}(X;\mathcal{H}(C;\Phi)), Y), \end{aligned} \tag{1}$$

where $\mathcal{L}$ is the loss function. Following the formula above, the general gradient update strategy is formulated as: $\theta_m^n[t] = \theta_m^n[t-1] - \eta \frac{\partial \mathcal{L}}{\partial \theta_m^n[t-1]}$. In hypernetworks, main network weights Θ are the output of hypernetwork $\mathcal{H}(C;\Phi)$. Then, the gradient of hypernetwork weights is: $\frac{\partial \mathcal{L}}{\partial \phi_i^j} = \sum_{m,n} \frac{\partial \mathcal{L}}{\partial \theta_m^n} \frac{\partial \theta_m^n}{\partial \phi_i^j}$, $\frac{\partial \mathcal{L}}{\partial \theta_m^n}$ is the main network's weight gradient, θ_m^n is the nth parameter in m layer of main network and ϕ_i^j is the jth parameter in i layer of hypernetwork. Now we suppose there is a set: $\mathbb{Y} = \{\Theta_1, \ldots, \Theta_i, \ldots\}$. The element Θ_i in $\mathbb{Y}$ are the parameters such that gradients of the main network are equal to 0. The existence of set $\mathbb{Y}$ is obvious. It is equivalent to the existence of minima (both local and global) and saddle points in neural networks.

One part of our motivation is based on the fact that the main problem of high-dimensional non-convex optimization is saddle points (Dauphin et al., 2014). Given that most neural networks satisfy this condition, we argue that the generalizability of this assumption is valid.

When the parameter θ_m^n is generated by hypernetwork $\mathcal{H}(C;\Phi)$, the gradient of hypernetwork parameter Φ is:

$$\begin{aligned} &\frac{\partial \mathcal{L}}{\partial \phi_i^j} = \sum_{m,n} \frac{\partial \mathcal{L}}{\partial \theta_m^n} \frac{\partial \theta_m^n}{\partial \phi_i^j} = 0. \\ &\text{s.t.} \quad \forall i, j, \phi_i^j \in \Phi \end{aligned} \tag{2}$$

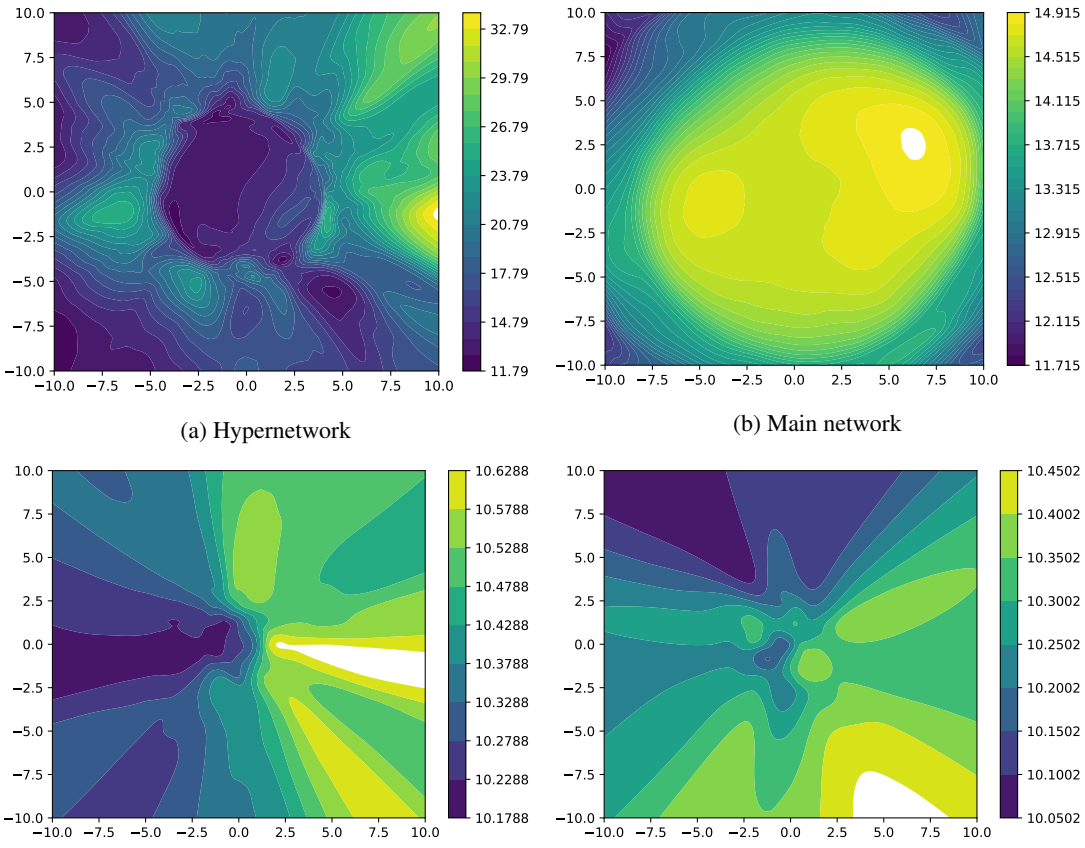
This means if there is a set $\mathbb{X} = \{\Phi_1, \ldots, \Phi_i, \ldots\}$, whose elements are the parameters such that gradients of hypernetwork equal to 0. We note $\mathcal{H}^{-1}$ as the inverse transformation of $\mathcal{H}$, then

$$\begin{aligned} &\mathcal{H}^{-1}(C;\mathbb{Y}) \subseteq \mathbb{X} \\ \text{s.t.} \quad &\mathcal{H}^{-1}(C;\mathbb{Y}) = \bigcup \mathcal{H}^{-1}(C;\Theta_i). \quad \Theta_i \in \mathbb{Y} \end{aligned} \tag{3}$$

In other words, the saddle points or minimum points of the main network are also those of the hypernetwork. Especially $\frac{\partial \mathcal{L}}{\partial \phi_i^j}$ is sum of main network's gradients $\frac{\partial \mathcal{L}}{\partial \theta_m^n}$, which can be close to zero while each part $\frac{\partial \mathcal{L}}{\partial \theta_m^n} \frac{\partial \theta_m^n}{\partial \phi_i^j}$ is not.

3.2 Alignment with Main Networks

In section 3.1, we find hypernetwork has a cumulative number of saddle points and minima that is at



(a) Hypernetwork

(b) Main network

(c) Initialized hypernetwork

(d) Initialized main network

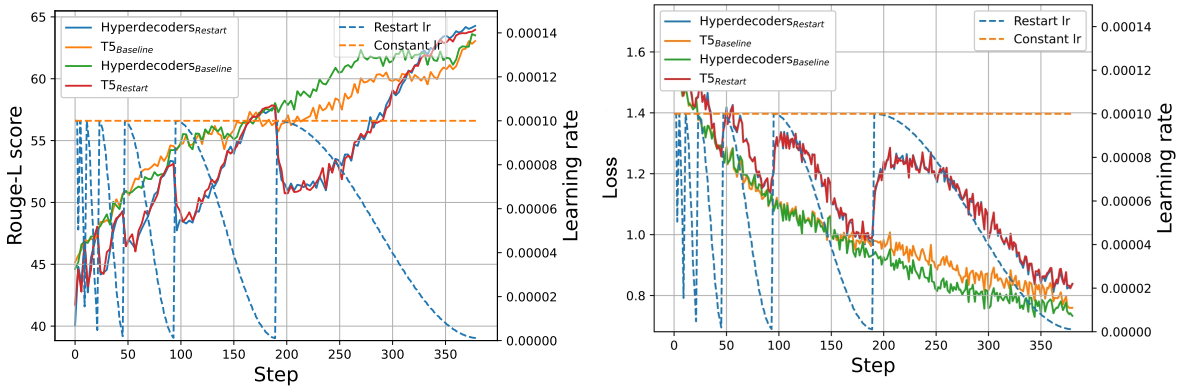
Figure 3: (a) and (b) illustrate that after training, the hypernetwork maintains a complicated loss landscape, in contrast to the main network, which exhibits a simplified landscape. And the hypernetwork's convergence to its minimum coincides with the main network reaching a saddle point. This corresponds to our results in section 3.2, which means the zero gradient points (minimum or saddle point) are corresponding. But it doesn't mean that the saddle points of the main network are also the saddle points of the hypernetwork. In this figure, it is the saddle point of the main network corresponding to the minimum of the hypernetwork. (c) and (d) demonstrate that, prior to training, both networks exhibit comparably intricate loss landscapes, reflecting the anticipated similarity in their initial optimization topography. This also corresponds to our results in section 3.2, which means the optimization route and difficulty of the two models in the very early stage of training will be similar.

least as great as the main network's. It is natural for us to figure out if there is a guarantee that the saddle points or minima of the main network align with those of the hypernetwork. Because the geometry of loss landscapes of models is characterized by the Hessian matrix of the loss function, we focus on analyzing the Hessian of hypernetworks. We denote the Hessian matrix of hypernetwork as $H_\phi = \frac{\partial^2 \mathcal{L}}{\partial \phi_i^j \partial \phi_k^l}$ where $i, j, k, l \in \{1, \ldots, |\Phi|\}$ and the Hessian matrix of main network as $H_\theta = \frac{\partial^2 \mathcal{L}}{\partial \theta_i^j \partial \theta_k^l}$ where $i, j, k, l \in \{1, \ldots, |\Theta|\}$. Then we can find that

$$H_\phi = J_\phi^T H_\theta J_\phi, \tag{4}$$

where J_ϕ is the Jacobian matrix of main network. The detailed proof can be found in Appendix A. Note that the Jacobi matrix usually becomes degenerate at the end of training because of instrisic dimension(Hu et al.). But at the beginning of training J_ϕ is not as degenerate as it is at the end of training. This suggests that the two Hessian matrices likely share the same positive and negative inertia coefficients.

To further validate that, we conduct experiments in Figure 3. It provides a visual representation of the hypernetwork and the corresponding network, both in their trained and untrained states. Our aim is to investigate the alignment of the loss landscape between hypernetworks and main networks. This visualization was generated using the methodology proposed by Li et al. (2018). The figure was plotted with an x-axis and y-axis range of [-10,



(a) Rouge-L Fmeasure of models

(b) Loss of models

Figure 4: Rouge-L Fmeasure and loss of models on P3 training split. The legend on the left corresponds to the Rouge-L score and loss. The legend on the right is the corresponding learning rate. The solid line corresponds to the left legend, and the dashed line corresponds to the right legend.

10] at a resolution of 256 steps. During the visualization process, specific parameters were fixed, and the Xavier initialization technique (Glorot and Bengio, 2010) was applied. Figure 3 (a) and (b) illustrate that after training, the hypernetwork maintains a complicated loss landscape, in contrast to the main network, which exhibits a simplified landscape. Notably, the hypernetwork's convergence to its minimum coincides with the main network reaching a saddle point, corroborating our prior analysis. Conversely, (c) and (d) demonstrate that, prior to training, both networks exhibit comparably intricate loss landscapes, reflecting the anticipated similarity in their initial optimization topography.

This ensures that, for hypernetworks, using some optimization methods from conventional LMs will not cause any performance loss at the beginning, but the results show that at the end of the optimization, the hypernetwork still maintains a relatively complex loss landscape while general LMs tend to flatten out. This requires special methods such as restart to help the model achieve the similar convergence effect as conventional LMs. The restart strategy's dynamic learning rate adjustments facilitate the model's ability to navigate complicated loss landscapes more effectively. By periodically resetting the learning rate, the strategy helps the model avoid local minima and encourages exploration, which can lead to better generalization on unseen data. It simulates a new warm-restarted run once T_i epochs are performed, where i is the index of the run. Its learning rate can be formulated as:

$$\eta_t = \eta_{min} + \frac{1}{2}(\eta_{max} - \eta_{min})(1 + cos\frac{T_{cur}}{T_i}\pi) \quad (5)$$

, where η_{max} is set to the initial learning rate, T_{cur} is the number of epochs since the last restart and T_i is the number of epochs between two warm restarts.

To enhance model convergence, we train the model for a total of $\sum_i T_i$ epochs, where T_i is the number of epochs between two restarts. This approach entails halting the training process when the learning rate reaches its minimum at a given epoch.

4 Experiments

Our theoretical derivations and methodologies aim to reveal the problem of hypernetwork optimization and investigate the corresponding solution. In the following part, we mainly use two hypernetworks, hyperdecoders and hyperformer. Hyperdecoders is a type of hypernetwork that generates the parameters of adapters in decoders and the adapters and FFNs are connected in parallel. Hyperformer is another type of hypernetwork that generates the parameters of layer normalization and FFNs in adapters and the adapters and FFNs are connected in series.

4.1 Instruction Tuning for Hypernetwork Restart

Due to the popularity of instruction tuning and the success of hypernetworks on instruction tuning, we first investigate instruction tuning on hypernet-

Table 1: Result of models on GLUE datasets. For MRPC, QQP and STS-B, we report the F1 score and for other tasks, we report accuracy. The green scores indicate improved performance on these datasets with the restart, while red scores indicate reduced performance. The $*$ means the result of our reproduction.

Model		CoLA	SST-2	MRPC	QQP	STS-B	MNLI	QNLI	RTE	Avg
Baseline	Finetuning*	47.43	99.30	90.85	92.31	83.50	84.04	98.60	78.83	84.35
	Hyperformer*	48.18	97.48	92.14	91.51	88.93	85.63	94.59	79.56	84.75
	Hyperformer++*	48.59	97.40	91.10	91.94	89.31	85.71	94.79	84.98	85.47
+ restart	Finetuning	42.27 (-5.16)	99.10 (-0.20)	90.07 (-0.78)	92.14 (-0.17)	87.90 (+4.40)	84.77 (+0.73)	98.80 (+0.20)	81.02 (+2.19)	84.50 (+0.15)
	Hyperformer	51.41 (+3.23)	98.00 (+0.52)	92.08 (-0.06)	91.53 (+0.02)	88.88 (-0.05)	86.24 (+0.61)	96.00 (+1.41)	83.21 (+3.65)	85.91 (+1.16)
	Hyperformer++	49.12 (+0.53)	97.80 (+0.40)	94.29 (+3.19)	92.07 (+0.13)	89.35 (+0.04)	86.02 (+0.31)	94.89 (+0.10)	81.75 (+0.73)	85.66 (+0.19)

works. In this section, we will try to transfer hypernetwork to instruction datasets to evaluate the effectiveness of restart for instruction tuning.

Table 2: Accuracy of models on P3 validation split.

Model	P3
FLAN-T5	63.051 ± 0.296
+ restart	$64.159 \pm 0.334 \uparrow$
Hyperdecoders	63.507 ± 0.201
+ restart	$\mathbf{64.904 \pm 0.302} \uparrow$

Setup In this experiment, we have opted to utilize Hyperdecoders based on the FLAN-T5 model and FLAN-T5 (Chung et al., 2022) baseline to compare the convergence of hyperLMs and conventional LMs, both with and without the use of restarts. P3 is a collection of prompted English datasets covering a diverse set of NLP tasks including question answering, dialogue, text generation, text editing, reasoning, etc., providing rich instruction examples for the model. We train these models on P3 (Public Pool of Prompts) T0 split (Sanh et al.), which is a multi-task instruction following dataset including 193 tasks. We use the AdamW optimizer in all subsequent experiments due to its proven effectiveness with transformer architectures and hypernetworks. Following the setup of (Ivison et al., 2023) that uses RougeL as the evaluation metric, we hope the performance of these models can be properly measured.

Our baseline in this scenario is the Hyperdecoders and FLAN-T5 models without restart. The FLAN-T5 and Hyperdecoders model is simply trained using the P3 dataset above. Because FLAN-T5 has been well-trained on instruction datasets, it is more reasonable to choose it as a baseline if we want to review the performance of hypernetworks on instruction tuning. For cosine annealing restart, we use a learning rate of $1e-4$, $\eta_{min} = 1e-6$, $T_0 = 1$ and $T_{mul} = 2$.

Experiments The result is shown in Figure 4 and we also show the final validation score in Table 2. In line with our prediction, though each time the result occurs the loss and the accuracy become worse, the results at the end of two restart intervals become better and finally outperform the model without restart. Notably, we set the restart learning rate as the same as without restart. From the perspective of the mean, this means we use a smaller learning rate in restart than in no restart, which results in a slightly higher loss. However, despite training the same number of steps with a smaller learning rate, the restart strategy still achieves better accuracy, demonstrating the value of this method.

Note that the data used for training is less than one epoch, which means the restart strategy effectively helps models (not only including hypernetwork models) converge fast. Moreover, at the end of the final training phase, the performance of models with the restart is much better than models with no restart, but finally, it slows down the convergence speed and reduces the gap between them. This may be because η_{min} of the restart strategy is small, and the learning rate in the end is too small.

To verify this hypothesis, we carried out a series of subsequent experiments, each with a distinct value of η_{min}. The impact of varying η_{min} on the outcome is depicted in Figure 5. Our results indicate that an excessively small η_{min} (specifically, when set to 0 in our experiment) leads to suboptimal performance compared to other values. However, it is also noteworthy that a large η_{min} does not necessarily guarantee improved results. This suggests a non-linear relationship between η_{min} and the performance of our model, emphasizing

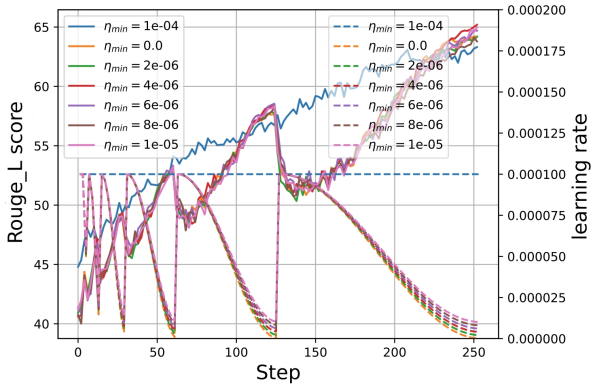


Figure 5: The rouge-L score of different restart η_{min} of models on GLUE dataset. The legend on the left corresponds to the Rouge-L score and the legend on the right is the corresponding learning rate. The solid line corresponds to the left legend, and the dashed line corresponds to the right legend

the importance of carefully selecting this parameter. Overall, we find restart strategy shows better results on the benchmark and outperforms models without restart, and this conclusion still holds for hypernetworks.

4.2 Comparison to Existing SOTA Hypernetworks

Setup To validate the effectiveness of the state-of-the-art hypernetwork model, we hope to employ the HINT. The authors demonstrated exceptional performance by training the HINT model on the T0 P3 dataset using TPUs and Google Bucket. However, due to the unavailability of these resources, we instead utilize Hyperdecoders as mentioned earlier, and apply them to the MRQA dataset (Fisch et al., 2019) to assess both in-domain and out-of-domain effectiveness. The MRQA dataset comprises 6 datasets for training and evaluation, namely HotpotQA (Yang et al., 2018), Natural Questions (Kwiatkowski et al., 2019), NewsQA (Trischler et al., 2017), SQuAD (Rajpurkar et al., 2016), SearchQA (Dunn et al., 2017), and TriviaQA (Joshi et al., 2017), as well as 6 additional datasets for out-of-domain evaluation, which include BioASQ (Tsatsaronis et al., 2015), DROP (Dua et al., 2019), DuoRC (Saha et al., 2018), RACE (Lai et al., 2017), RelationExtraction (Levy et al., 2017), and TextbookQA (Kembhavi et al., 2017). We conduct a full fine-tuning of these datasets using a global batch size of 256 and a learning rate of $3e-4$.

Experiments Table 3 and Table 4 show the result on MRQA dataset. In the in-domain MRQA validation set, models employing restart techniques demonstrate a marked improvement over the baseline Hyperdecoders. This suggests that these techniques may facilitate faster convergence and thus enhance performance on tasks within the model's training domain. However, the effectiveness of the restart technique is less consistent in the out-of-domain MRQA validation set. While it outperforms the baseline in some datasets, it does not in others. This indicates that the benefits of the restart approach may not generalize across all tasks, particularly those that lie outside the model's initial training domain.

4.3 Comparison to Other Hypernetworks

Setup We follow the set and model of (Mahabadi et al., 2021) and evaluate the performance on the GLUE benchmark (Wang et al., 2018). We select this model due to its unique hypernetwork architecture that generates adapters in both the encoder and decoder. This distinguishes it from our previously discussed model. For simplicity, we only conduct the multi-task training because of its popularity in recent works. This benchmark we use covers paraphrase detection (MRPC, QQP), sentiment classification (SST-2), natural language inference (MNLI, RTE, QNLI), and linguistic acceptability (CoLA). We fine-tune the models with a learning rate of 3e-4 (for restart strategy we make the $\eta_{max} = 3e-4$). The other hyperparameters are selected similarly as Mahabadi et al. (2021). For better comparison, we choose the finetuning model and the Hyperformer model without restart as a baseline.

Experiments To further study the generalization of this method, we conduct the following experiments to verify this strategy works on most hypernetworks. As evidenced by our experimental results presented in Table 1, substantial enhancements can be realized on standard datasets. With restart, hypernetwork can universally increase performance on nearly every task. The benefits of implementing restarts are especially noticeable in the MRPC, CoLA, and RTE tasks, where the hypernetwork model experiences a 3-point improvement. In turn, this contributes to a significant elevation in the average performance across these tasks. Fascinatingly, the restart strategy does not confer the same degree of improvement on the standard T5 model as it does on the hypernetwork. In fact, in the CoLA task, the application of the restart strategy results in a 5-point performance decrease compared to the model without restarts. This suggests that the

Table 3: Accuracy and F1 score of models on in-domain MRQA validation split. The green scores indicate improved performance on these datasets with the restart, while red scores indicate reduced performance.

Model	SQuAD	HotpotQA	TriviaQA	NewsQA	SearchQA	NaturalQs	Avg
Hyperdecoders	84.61/91.22	62.63/78.43	68.82/73.73	53.09/67.63	77.31/82.29	64.99/77.19	68.58/78.41
+ restart	84.38/91.48 (-0.23)/(+0.26)	63.77/79.36 (+1.14)/(+0.93)	70.31/74.83 (+1.49)/(+1.10)	54.30/68.74 (+1.21)/(+1.11)	77.78/82.79 (+0.47)/(+0.50)	66.79/78.96 (+1.80)/(+1.77)	70.00/79.36 (+1.42)/(+0.95)

Table 4: Accuracy and F1 score of models on out-domain MRQA validation split. The green scores indicate improved performance on these datasets with the restart, while red scores indicate reduced performance.

Model	BioASQ	DROP	DuoRC	RACE	Relation Ext.	TextbookQA	Avg
Hyperdecoders	54.65/68.82	36.26/45.33	48.90/58.74	32.94/46.35	74.14/85.44	47.04/55.66	48.99/60.04
+ restart	51.93/66.24 (-2.72)/(-2.58)	37.46/45.83 (+1.20)/(+0.50)	47.17/57.38 (-1.73)/(-1.36)	34.12/47.98 (+1.16)/(+1.63)	72.92/84.60 (-1.22)/(-0.84)	46.17/56.26 (-0.87)/(+0.60)	48.30/59.72 (-0.69)/(-0.32)

efficacy of the restart strategy may be different on the hyperLMs and conventional LMs, underscoring the need for careful consideration when applying such strategies. Moreover, our results indicate that the restart strategy generally leads to improved performance across most tasks when applied to hypernetwork models. Notably, the Hyperformer model with restarts outperforms all other models in terms of average performance, demonstrating the potential of this approach.

4.4 Comparison to Other Learning Rate Schedulers

Table 5: Accuracy of different strategies for Hyperdecoders on GLUE datasets.

Scheduler	GLUE
Constant	83.23 ± 0.29
Cosine	83.27 ± 0.33
Linear	84.18 ± 0.20
Polynomial	83.43 ± 0.27
Restart	$\mathbf{84.88 \pm 0.33}$

To evaluate the effectiveness of the restart strategy against other learning rate schedulers, we conducted experiments using the Hyperdecoders on the GLUE benchmark. Our comparison includes a variety of learning rate schedules: constant, linear with warmup, cosine with warmup, and polynomial decay with warmup. The results, presented in Table 5, demonstrate that the restart strategy surpasses its counterparts. This strategy consistently achieved the highest scores, indicating its robustness in handling the diverse set of tasks.

The restart strategy's success can be largely attributed to its dynamic learning rate adjustments, which facilitate the model's ability to navigate complicated loss landscapes more effectively. By periodically resetting the learning rate, the strategy helps the model avoid local minima and encourages exploration, which can lead to better generalization on unseen data. Moreover, the periodic nature of the restart strategy acts as a form of implicit regularization. This can potentially reduce the risk of overfitting(Loshchilov and Hutter, 2022).

5 Limitations

As previously discussed, the hypernetwork introduces a novel parameter space, making it more suitable for incorporating a new restart strategy. We have demonstrated the effectiveness of this approach and hope to explore these directions further in future work. The hypernetwork architecture is not only applicable to natural language processing but also raises the question of whether it can be effectively applied to other modalities, such as images or multimodal data. Additionally, due to the efficiency challenges associated with hypernetworks, we have not yet extended our experiments to larger-scale language models. Exploring solutions to these challenges remains an important direction for future research.

6 Conclusions

To address the problems due to the introduction of novel parameter spaces, we analyze the loss landscape of hypernetworks and main networks. We propose that the restart strategy for the learning rate can improve the performance and conver-

gence speed of hypernetworks. We conduct our experiments on three different tasks and across two different hypernetwork models: Hyperformer and Hyperdecoders. We anticipate that our work will stimulate further research into hypernetworks optimization, leading to models that are more closely aligned with conventional LMs. Additionally, we hope our findings will inspire more research into improving hypernetworks optimization strategies.

Acknowledgments

This work was supported in part by the National Key R&D Program of China (No. 2022ZD0118201), the Shenzhen Medical Research Funds in China (No. B2302037), Natural Science Foundation of China (No. 61972217, 32071459, 62176249, 62006133, 62271465), and AI for Science (AI4S)-Preferred Program, Peking University Shenzhen Graduate School, China.

References

Vinod Kumar Chauhan, Jiandong Zhou, Ping Lu, Soheila Molaei, and David A Clifton. 2023a. A brief review of hypernetworks in deep learning. *arXiv preprint arXiv:2306.06955*.

Vinod Kumar Chauhan, Jiandong Zhou, Soheila Molaei, Ghadeer Ghosheh, and David A Clifton. 2023b. Dynamic inter-treatment information sharing for heterogeneous treatment effects estimation. *arXiv preprint arXiv:2305.15984*.

Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. 2015. The loss surfaces of multilayer networks. In *Artificial intelligence and statistics*, pages 192–204. PMLR.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Yann Dauphin, Harm De Vries, and Yoshua Bengio. 2015. Equilibrated adaptive learning rates for non-convex optimization. *Advances in neural information processing systems*, 28.

Yann N Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. 2014. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *Advances in neural information processing systems*, 27.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378.

Matthew Dunn, Levent Sagun, Mike Higgins, V Ugur Guney, Volkan Cirik, and Kyunghyun Cho. 2017. Searchqa: A new q&a dataset augmented with context from a search engine. *arXiv preprint arXiv:1704.05179*.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. Mrqa 2019 shared task: Evaluating generalization in reading comprehension. In *2nd Workshop on Machine Reading for Question Answering, MRQA@ EMNLP 2019*, pages 1–13. Association for Computational Linguistics (ACL).

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings.

Kshitij Gupta, Benjamin Thérien, Adam Ibrahim, Mats L Richter, Quentin Anthony, Eugene Belilovsky, Irina Rish, and Timothée Lesort. 2023. Continual pretraining of large language models: How to (re) warm your model? *arXiv preprint arXiv:2308.04014*.

David Ha, Andrew M Dai, and Quoc V Le. 2022. Hypernetworks. In *International Conference on Learning Representations*.

Yun He, Steven Zheng, Yi Tay, Jai Gupta, Yu Du, Vamsi Aribandi, Zhe Zhao, YaGuang Li, Zhao Chen, Donald Metzler, et al. 2022. Hyperprompt: Prompt-based task-conditioning of transformers. In *International Conference on Machine Learning*, pages 8678–8690. PMLR.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Hamish Ivison, Akshita Bhagia, Yizhong Wang, Hannaneh Hajishirzi, and Matthew Peters. 2023. Hint: Hypernetwork instruction tuning for efficient zero- and few-shot generalisation. In *The 61st Annual Meeting Of The Association For Computational Linguistics*.

Hamish Ivison and Matthew E Peters. 2022. Hyperdecoders: Instance-specific decoders for multi-task nlp. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1715–1730.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly

supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611.

Aniruddha Kembhavi, Minjoon Seo, Dustin Schwenk, Jonghyun Choi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Are you smarter than a sixth grader? textbook question answering for multimodal machine comprehension. In *Proceedings of the IEEE Conference on Computer Vision and Pattern recognition*, pages 4999–5007.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. Race: Large-scale reading comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. In *21st Conference on Computational Natural Language Learning, CoNLL 2017*, pages 333–342. Association for Computational Linguistics (ACL).

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. 2018. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems*, 31.

Ilya Loshchilov and Frank Hutter. 2022. Sgdr: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*.

Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. 2021. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. In *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)*, pages 565–576.

Purnendu Mishra and Kishor Sarawadekar. 2019. Polynomial learning rate policy with warm restart for deep neural network. In *TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON)*, pages 2087–2092.

Brendan O'donoghue and Emmanuel Candes. 2015. Adaptive restart for accelerated gradient schemes. *Foundations of computational mathematics*, 15:715–732.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Amrita Saha, Rahul Aralikatte, Mitesh M Khapra, and Karthik Sankaranarayanan. 2018. Duorc: Towards complex language understanding with paraphrased reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1683–1693.

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, et al. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*.

Leslie N Smith. 2017. Cyclical learning rates for training neural networks. In *2017 IEEE winter conference on applications of computer vision (WACV)*, pages 464–472. IEEE.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. Newsqa: A machine comprehension dataset. *ACL 2017*, page 191.

George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, et al. 2015. An overview of the bioasq large-scale biomedical semantic indexing and question answering competition. *BMC bioinformatics*, 16(1):1–28.

Tomer Volk, Eyal Ben-David, Ohad Amosy, Gal Chechik, and Roi Reichart. 2022. Example-based hypernetworks for out-of-distribution generalization. *arXiv preprint arXiv:2203.14276*.

Johannes von Oswald, Christian Henning, Benjamin F Grewe, and João Sacramento. 2020. Continual learning with hypernetworks. In *8th International Conference on Learning Representations (ICLR 2020)(virtual)*. International Conference on Learning Representations.

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 preprint arXiv:1804.07461*.

Zedian Xiao, William Held, Yanchen Liu, and Diyi Yang. 2023. Task-agnostic low-rank adapters for unseen english dialects. *arXiv preprint arXiv:2311.00915*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380.

Qinyuan Ye and Xiang Ren. 2021. Learning to generate task-specific adapters from task description. In

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), pages 646–653.

Dominic Zhao, Seijin Kobayashi, João Sacramento, and Johannes von Oswald. 2020. Meta-learning via hypernetworks. In *4th Workshop on Meta-Learning at NeurIPS 2020 (MetaLearn 2020)*. NeurIPS.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

A Proof of Hessian.

We denote loss function $\mathcal{L} = \mathcal{F}(\theta_1, \ldots, \theta_n)$, $\theta_i = \mathcal{H}(\phi_1, \ldots, \phi_n)$. From the chain rule, we can derive that:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \phi_i^j} =& \frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}} \frac{\partial \boldsymbol{\theta}}{\partial \phi_i^j} \\ =& \nabla_{\boldsymbol{\theta}} \mathcal{L} \cdot \mathbf{X}_{ij}, \end{aligned} \tag{6}$$

where $\nabla_{\boldsymbol{\theta}} \mathcal{L} = \begin{bmatrix} \frac{\partial \mathcal{L}}{\partial \theta_1} & \frac{\partial \mathcal{L}}{\partial \theta_2} & \ldots & \frac{\partial \mathcal{L}}{\partial \theta_n} \end{bmatrix}$ and $\mathbf{X}_{ij} = \begin{bmatrix} \frac{\partial \theta_1}{\partial \phi_i^j} & \frac{\partial \theta_2}{\partial \phi_i^j} & \cdots & \frac{\partial \theta_n}{\partial \phi_i^j} \end{bmatrix}^T$. Then we estimate their loss landscape Hessian matrix by calculating the second-order derivative,

$$\begin{aligned} \frac{\partial^2 \mathcal{L}}{\partial \phi_i^j \partial \phi_k^l} &= \mathbf{X}_{ij}^T \mathbf{H}_\theta \mathbf{X}_{kl} + \nabla_{\boldsymbol{\theta}} \mathcal{L} \cdot \mathbf{X}_{ij,kl}, \\ \mathbf{X}_{ij,kl} &= \begin{bmatrix} \frac{\partial^2 \theta_1}{\partial \phi_i^j \partial \phi_k^l} & \frac{\partial^2 \theta_2}{\partial \phi_i^j \partial \phi_k^l} & \cdots & \frac{\partial^2 \theta_n}{\partial \phi_i^j \partial \phi_k^l} \end{bmatrix}^T. \end{aligned} \tag{7}$$

Note that $\frac{\partial^2 \mathcal{L}}{\partial \phi_i^j \partial \phi_k^l}$ is the items of matrix H_ϕ. We can further obtain

$$\begin{aligned} \mathbf{H}_\phi =& \begin{bmatrix} \mathbf{X}_1^T \cdot \mathbf{H}_\theta \cdot \mathbf{X}_1 & \mathbf{X}_1^T \cdot \mathbf{H}_\theta \cdot \mathbf{X}_2 & \ldots & \mathbf{X}_1^T \cdot \mathbf{H}_\theta \cdot \mathbf{X}_m \\ \mathbf{X}_2^T \cdot \mathbf{H}_\theta \cdot \mathbf{X}_1 & \mathbf{X}_2^T \cdot \mathbf{H}_\theta \cdot \mathbf{X}_2 & \ldots & \mathbf{X}_2^T \cdot \mathbf{H}_\theta \cdot \mathbf{X}_m \\ \ldots & \ldots & \ddots & \ldots \\ \mathbf{X}_m^T \cdot \mathbf{H}_\theta \cdot \mathbf{X}_1 & \mathbf{X}_m^T \cdot \mathbf{H}_\theta \cdot \mathbf{X}_2 & \ldots & \mathbf{X}_m^T \cdot \mathbf{H}_\theta \cdot \mathbf{X}_m \end{bmatrix} + \mathbf{R} \\ =& \mathbf{J}_\phi^T \mathbf{H}_\theta \mathbf{J}_\phi + \mathbf{R}, \end{aligned} \tag{8}$$

where J_ϕ is the Jacobi matrix of main network and $R_{ij} = \sum \frac{\partial \mathcal{L}}{\partial \theta_i} \frac{\partial^2 \theta_i}{\partial \phi_i^j \partial \phi_k^l}$.

In usual LMs, we can suppose the second derivative in R is small enough to be ignored. Then we find that

$$H_\phi = J_\phi^T H_\theta J_\phi. \tag{9}$$

B Dataset Statistics

Tables 6, 7 and 8 provide some summary statistics of each dataset used and split sizes.

Table 6: Summary statistics of splits used when evaluating GLUE

Dataset	Train Split Size	Validation Split Size	Test Split Size
CoLA	8551	521	522
SST-2	66349	1000	872
STS-B	5749	750	750
MRPC	3668	204	204
QQP	362846	1000	40430
MNLI	392702	9832	9815
QNLI	103743	1000	5463
RTE	2490	138	139

Table 7: Summary statistics of splits used when evaluating MRQA

Dataset	Train Split Size	Validation Split Size
SQuAD	86588	10507
HotpotQA	72928	5901
TriviaQA	61688	7785
NewsQA	74160	4212
SearchQA	117384	16980
Natural Qs	104071	12836
BioASQ	-	1504
DROP	-	1503
DuoRC	-	1501
RACE	-	674
Relation Ext.	-	2948
TextbookQA	-	1503

Table 8: Summary statistics of splits used when evaluating P3 split

Dataset	Train Split Size	Validation Split Size	Test Split Size
P3-split	3301000	3301	3301