# ON THE PITFALLS OF WEIGHT DECAY DECOUPLING

**Haoming Liu, Xinhao Liu**


## ABSTRACT

In deep learning, the effectiveness of optimization algorithms is paramount for training neural networks efficiently. While adaptive gradient methods such as Adam have been proven beneficial in handling sparse gradients and non-stationary objectives, they often underperform in generalization compared to Stochastic Gradient Descent (SGD) with momentum. This project is inspired by AdamW, an optimizer that decouples weight decay from loss function, thereby achieving a better generalization performance compared to Adam. The main objective of this project is to extend this understanding by providing a comprehensive analysis of the two key components, weight decay and hyper-parameter decoupling. We first discuss the effect of different weight decay formulations in the context of adaptive gradient methods. Then, we investigate the true performance of hyper-parameter decoupling through heatmap visualizations. Notably, we present some observations and insights that are inconsistent with the ones claimed in prior works, which can be helpful in further research in this field. The code is publicly available at: `https://github.com/Gaaaavin/CS-GY-6763-AMLDS`.


## 1 INTRODUCTION

The field of machine learning, particularly deep learning, relies heavily on the efficiency of optimization algorithms to train neural networks. Adaptive gradient methods such as AdaGrad (Duchi et al., 2011), RMSProp Hinton et al. (2012), Adam (Kingma & Ba, 2014), and AMSGrad (De et al., 2019) have become widely used owing to their ability to handle sparse gradients and non-stationary objectives. However, these methods, especially Adam, are often criticized for their lackluster generalization performance compared to Stochastic Gradient Descent (SGD) with momentum. The generalization gap between adaptive methods and SGD with momentum in certain applications, such as image classification, has spurred research into enhancing the capabilities of these methods to match or surpass the benchmark set by SGD.

Our project is inspired by the seminal work AdamW (Loshchilov & Hutter, 2019), which revisits the role of weight decay regularization in adaptive gradient methods. The study distinguishes the often conflated concepts of L2 regularization and weight decay, particularly underlining their divergence in adaptive methods like Adam. This led to the development of AdamW, an iteration of the Adam optimizer that decouples weight decay from the loss function optimization process.

This project aims to critically analyze the decoupled weight decay proposed in AdamW, uncovering the misleading pitfalls in prior works in terms of the two key mechanisms, weight decay and hyper-parameter decoupling. Specifically, we extend the comparison of weight decay formulations under learning rate annealing from vanilla SGD to adaptive gradient methods with learning rate annealing. Then, we investigate the true effect of weight decay decoupling by replicating the experiments in the AdamW paper. We also compare the grid search results for the first moment estimate parameter $\beta_1$ to investigate potential of hyper-parameter decoupling in general.

The insights gained from this study are expected to contribute significantly to the field of neural network optimization, as many of them are diverged from the ones claim in prior works. By dissecting and analyzing the mechanics of a widely-utilized optimizer, this research seeks to provide practical value to practitioners in the field, particularly in rethinking the effect of weight decay and hyper-parameter decoupling for designing novel adaptive gradient based optimizers.

## 2 Related Work

**Adaptive Gradient Methods.** The landscape of optimization in deep learning has been significantly shaped by adaptive gradient methods. AdaGrad, introduced by Duchi et al. (Duchi et al., 2011), was pioneering in its approach to adjusting learning rates based on the history of gradients. RMSProp, an unpublished but widely cited work by Hinton et al. Hinton et al. (2012), further refined this approach by introducing a moving average of squared gradients. Adam (Kingma & Ba, 2014) combined the benefits of AdaGrad and RMSProp and proposed an adaptive gradient algorithm that does both first and second moment estimate. This was further refined by Reddi et al. (De et al., 2019) in AMSGrad, addressing some of the convergence issues in Adam.

**AdamW.** AdamW, developed by Loshchilov and Hutter (Loshchilov & Hutter, 2019), marks a significant development in the landscape of adaptive optimization methods. It addresses a key limitation of the Adam optimizer by decoupling weight decay from the adaptive learning rate updates. In traditional adaptive methods like Adam, the weight decay component is intertwined with the adaptive learning rate, which can diminish the intended regularization effect. AdamW corrects this by applying weight decay directly to the weights, independent of the learning rate adaptation. This not only preserves the regularization benefits of weight decay but also enhances the generalization performance of the model, particularly in scenarios where Adam falls short. Though the success of AdamW has been widely acknowledged by the vast community, we also found some overlooked pitfalls through the experiment results presented in later sections.

**Applications.** The impact of adaptive gradient based optimizers like Adam and AdamW is evident in their widespread use in state-of-the-art deep learning models. For instance, the GPT series by Brown et al. (Brown et al., 2020) and Radford et al. (Radford et al., 2019) utilized Adam for training large-scale language models. Similarly, Vision Transformers (ViT) by Dosovitskiy et al. (Dosovitskiy et al., 2020) and CLIP by Radford et al. (Radford et al., 2021) demonstrated the effectiveness of these optimizers in vision-related tasks. He et al.'s ResNet (He et al., 2016), another groundbreaking model in image recognition, also benefits from these advanced optimization techniques.

## 3 Revisiting AdamW

**Overview.** The AdamW optimizer (Loshchilov & Hutter, 2019), presents an innovative approach to optimization in neural networks. This section revisits the central claims and theoretical foundations laid out in their paper, along with the pseudo-code that elucidates the implementation of AdamW.

**Propositions in AdamW.** The AdamW paper posits three key propositions:

1. *Weight Decay as L2 Regularization in Standard SGD*: The first proposition establishes that for standard SGD, the inclusion of weight decay is equivalent to L2 regularization. This equivalence means that executing SGD with a base learning rate $\alpha$ on a batch loss function $f_t(\theta)$ with weight decay $\lambda$ is the same as optimizing the regularized batch loss function $f_t^{reg}(\theta)$, with a modified weight decay factor $\lambda' = \frac{\lambda}{\alpha}$.
2. *Distinction between Weight Decay and L2 Regularization for Adaptive Gradients*: The second proposition highlights a divergence between weight decay and L2 regularization when using adaptive gradients. It states that for adaptive gradient methods, no L2 coefficient $\lambda$ can replicate the effect of running the optimizer on a loss function with weight decay. This is due to the non-identity preconditioner matrix $M_t$ altering the update rule, making it fundamentally different from L2 regularization.
3. *Scale-Adjusted L2 Regularization for Adaptive Gradient Algorithms*: The third proposition introduces the concept of scale-adjusted L2 regularization for adaptive gradient algorithms with a fixed preconditioner matrix. It asserts that using the proposed algorithm (see algorithm 1) with base learning rate $\alpha$ to optimize a batch loss function with weight decay $\lambda$ is equivalent to optimizing the scale-adjusted regularized batch loss function without weight decay. This equivalence is represented by the modified batch loss function $f_t^{sreg}(\theta)$, incorporating element-wise multiplication and square root transformations applied to the preconditioner matrix. However, this cannot be applied to practical adaptive gradient algorithms, as the preconditioner matrix is modified at every step.

**Algorithm 1** Adam (Kingma & Ba, 2014) and AdamW Optimizer (Loshchilov & Hutter, 2019)

**given** initial learning rate $\alpha \in \mathbb{R}$, exponential decay rates for moment estimates $\beta_1, \beta_2 \in [0, 1)$, decoupled weight decay factor $\lambda \in \mathbb{R}$, $\epsilon = 10^{-8}$
**initialize** time step $t \leftarrow 0$, parameter vector $\boldsymbol{\theta}_{t=0} \in \mathbb{R}^n$, first moment vector $\boldsymbol{m}_{t=0} \leftarrow \boldsymbol{0}$, second moment vector $\boldsymbol{v}_{t=0} \leftarrow \boldsymbol{0}$, schedule multiplier $\eta_{t=0} \in \mathbb{R}$
**repeat**
  $t \leftarrow t + 1$
  $\nabla f_t(\boldsymbol{\theta}_{t-1}) \leftarrow \text{SelectBatch}(\boldsymbol{\theta}_{t-1})$ ▷ select batch and return the corresponding gradient
  $\boldsymbol{g}_t \leftarrow \nabla f_t(\boldsymbol{\theta}_{t-1}) + \lambda \boldsymbol{\theta}_{t-1}$ ▷ compute gradients for the selected mini-batch
  $\boldsymbol{m}_t \leftarrow \beta_1 \boldsymbol{m}_{t-1} + (1 - \beta_1)\boldsymbol{g}_t$ ▷ first moment estimate
  $\boldsymbol{v}_t \leftarrow \beta_2 \boldsymbol{v}_{t-1} + (1 - \beta_2)\boldsymbol{g}_t^2$ ▷ second moment estimate
  $\hat{\boldsymbol{m}}_t \leftarrow \boldsymbol{m}_t / (1 - \beta_1^t)$ ▷ bias correction for first moment estimate
  $\hat{\boldsymbol{v}}_t \leftarrow \boldsymbol{v}_t / (1 - \beta_2^t)$ ▷ bias correction for second moment estimate
  $\eta_t \leftarrow \text{SetScheduleMultiplier}(t)$ ▷ learning rate annealing
  $\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} - \eta_t \left( \alpha \hat{\boldsymbol{m}}_t / (\sqrt{\hat{\boldsymbol{v}}_t} + \epsilon) + \lambda \boldsymbol{\theta}_{t-1} \right)$ ▷ update parameters
**until** *stopping criterion is met*
**return** optimized parameters $\boldsymbol{\theta}_t$

**AdamW optimizer.** The algorithm 1 outlines the AdamW optimization algorithm, which essentially modifies the Adam optimizer to improve the way it handles weight decay. In standard Adam, we use a L2 regularization term that is coupled in the loss function, which may potentially lead to inconsistent application of weight decay across parameters and, consequently, an unintended regularization effect. AdamW addresses this by applying weight decay directly while updating the parameters. This ensures a more consistent regularization effect across all parameters, regardless of their gradient magnitudes, resulting in a better generalization ability.

## 4 Experiment Result

### 4.1 When Decoupled Weight Decay Meets Learning Rate Annealing

As noted by Loshchilov and Hutter (Loshchilov & Hutter, 2019) in the AdamW paper, there are actually two types of "weight decay": L2 regularization and weight decay. The former is coupled in the loss function as a regularization term, as shown by the pseudocode of Adam in algorithm 1. In contrast, the latter regularizes the norm of parameters directly while updating the parameters. The initial form of weight decay was proposed by Hanson and Pratt (Hanson & Pratt, 1988):

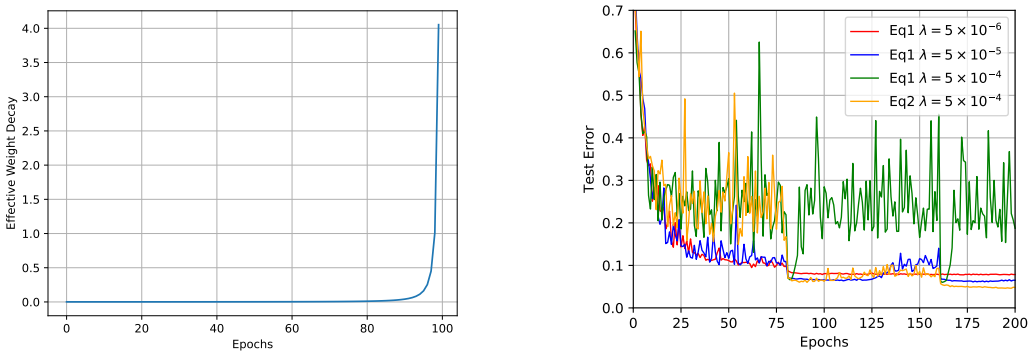
$$\boldsymbol{\theta}_t = (1 - \lambda')\boldsymbol{\theta}_{t-1} - \eta_t \alpha \tilde{\boldsymbol{g}}_{t-1}, \tag{1}$$

where $\lambda'$ is a fixed weight decay factor, $\alpha$ is the initial learning rate, $\eta_t$ is the learning rate scaling factor at step $t$, and $\tilde{\boldsymbol{g}}_{t-1}$ is the processed gradient for the current mini-batch. Hence, we have $\tilde{\boldsymbol{g}}_{t-1} = \nabla f_t(\boldsymbol{\theta}_{t-1})$ for vanilla SGD and $\tilde{\boldsymbol{g}}_{t-1} = \hat{\boldsymbol{m}}_t / (\sqrt{\hat{\boldsymbol{v}}_t} + \epsilon)$ for AdamW. In comparison, the Decoupled Weight Decay proposed in the AdamW paper can be written as follows:

$$\boldsymbol{\theta}_t = (1 - \eta_t \alpha \lambda)\boldsymbol{\theta}_{t-1} - \eta_t \alpha \tilde{\boldsymbol{g}}_{t-1}, \tag{2}$$

where $\lambda$ is the initial weight decay factor. In general, the formulation shown in Eq. (2) is a more popular implementation adopted by deep learning libraries, such as PyTorch and TensorFlow. By the first proposition in the AdamW paper, the two formulations are exactly the same if we do not apply learning rate scheduling (i.e., $\eta_t = 1$) and let $\lambda' = \alpha\lambda$ for a vanilla SGD optimizer. However, this is not the case when the learning rate scheduler come into play, which should be a common setup while training a deep neural networks nowadays.

One key observation here is that though the formulation in Eq. (1) seems to be simpler and allows more flexibility for hyper-parameter tuning, it is also potentially problematic. If we adopt learning rate annealing for easier convergence, such a formulation would lead to the explosion of the weight decay factor, as the adjusted learning rate gradually gets smaller along with training. Figure 1a demonstrates this phenomenon with a cosine learning annealing. Figure 1b is borrowed from Xie et al. (Xie et al., 2023), where they compare the two formulations using a vanilla SGD optimizer. We can see that the formulation in Eq. (2) indeed outperforms the one in Eq. (1). On top of that,




(a) Illustration of the exploded weight decay.

(b) Comparison using vanilla SGD.

Figure 1: Potential drawback of using the fixed weight decay formulation in Eq. (1).

they argued that the formulation in Eq. (2) is theoretically better than Eq. (1) [1]. We rephrase their non-convergence theorem as below.

**Theorem 1.** *(Non-convergence due to fixed weight decay and learning rate annealing). Suppose learning dynamics is governed by GD with fixed weight decay (Eq. (1)) and the learning rate* $\tilde{\eta}_t := \eta_t \alpha \in (0, +\infty)$ *holds. If* $\exists\, \delta$ *such that* $0 < \delta \leq |\tilde{\eta}_t - \tilde{\eta}_{t+1}|$ *for any* $t > 0$*, then the learning dynamics cannot converge to any non-zero stationary point satisfying the condition:*

$$\max(||\nabla f_t(\boldsymbol{\theta}_t))||^2, ||\nabla f_t(\boldsymbol{\theta}_{t+1}))||^2) \geq \frac{\lambda'^2 \delta^2 ||\theta^*||^2}{\tilde{\eta}_t \tilde{\eta}_{t+1}} > 0,$$

*where* $f_t(\boldsymbol{\theta}) = \ell_{error}(\boldsymbol{\theta}) + \frac{\lambda'}{2\tilde{\eta}_t}||\boldsymbol{\theta}||^2$ *is the regularized loss function.*

We can first notice that the loss function is time-dependent, where the scaling factor for the regularization term gradually grows larger. Broadly speaking, Theorem 1 claims that the fixed formulation of weight weight decay (Eq. (1)) reduces the stability of the stationary points. Accordingly, the gradient norms will not converge to zero. On top of that, Xie et al. (Xie et al., 2023) further show that even with proper convergence guarantees and stabler minima offered by decoupled weight decay formulation (Eq. (2)), the gradient norm upper bound at convergence still monotonically increases with the weight decay factor $\lambda$. To tackle this problem, they proposed to adaptively adjust the weight decay factor by averaging the gradient norms (i.e., second moment estimate $\hat{\boldsymbol{v}}_{\boldsymbol{t}}$).

The non-convergence claim mentioned earlier is empirically supported by their experiments using vanilla SGD, as shown in Figure 1b. However, results using adaptive gradient methods, which form the basis of their method, have not been included. This observation leads us to conduct complementary experiments to explore the impact of various weight decay formulations on Adam and AdamW variants. Our hypothesis is that the impact of weight decay factor explosion can be minor, since the learning rate can be quite small and has negligible impact in the final training phase.

To verify this hypothesis, we conduct experiments on the commonly used CIFAR-10 (Krizhevsky, 2009) dataset using ResNet-18 (He et al., 2016). More specifically, we evaluate Adam and AdamW with two different weight decay formulations (Eq. (1) and Eq. (2)). We use a learning rate of `1e-3` for all weight decay factor within the search space {`1e-1`, `1e-2`, `1e-3`, `1e-4`, `1e-5`, `1e-6`}. We run each experiment 5 times using different random seed with cosine learning rate scheduling. We plot the average, maximum, and minimum test errors in Figure 2. By taking a closer look, we can draw the following conclusions: First, weight decay (i.e., AdamW variants) ensures a smoother convergence compared to L2 regularization (i.e., Adam), especially when the weight decay factor is larger. Second, all the three optimizers give similar performance when the weight decay factor is small. Third, AdamW variants is not necessarily better than Adam. With proper hyper-parameters, it can even converge to better results. Last, unlike the performance gap shown in Figure 1b, both

[1] We formulate the weight decay in a more general way, but the core idea is essentially the same.

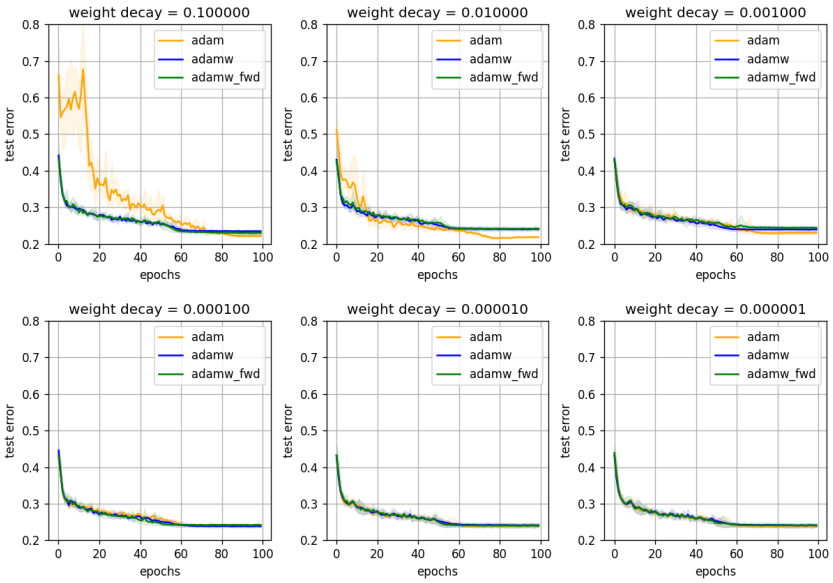



Figure 2: Comparison of Adam and AdamW variants with different weight decay formulations, where 'adamw_fwd' corresponds to Eq. (1) and 'adamw' corresponds to Eq. (2).

formulations of weight decay exhibit extremely close performance based on adaptive gradient methods. We attribute this distinction to two aspects. On the one hand, the scheduled learning rate in the final training phase can be quite small; on the other hand, the large hyper-parameters for first and second moment estimate (i.e., $(\beta_1, \beta_2) = (0.9, 0.999)$) help alleviate the impact of the explosion of weight decay. Hence, we can reach an interesting conclusion: **the impact of different weight decay formulations is minor when learning rate annealing is applied**.

### 4.2 Replicating the Heatmaps in AdamW

From the previous section, we have observed that AdamW variants does not necessarily perform better than Adam, and Adam may even gives better performance. As a result, we want to further investigate the true performance gain of AdamW by replicating the heatmap experiments.

To reproduce the heatmaps, we conduct hyper-parameter grid search on the CIFAR-10 (Krizhevsky, 2009) dataset using ResNet-18 (He et al., 2016). Specifically, we search the learning rate within the log-uniform space of (`1e-11`, `1e-1`) and the weight decay factor within the log-uniform space of (`1e-13`, `1e-3`). We still apply cosine learning rate scheduling and run each experiment 5 times with different random seed to average the final test errors. The final heatmap is obtained through interpolation all grid points, as shown in the first row of Figure 3. Besides, we also attach the grid search results before interpolation in the Appendix (Figure 4).

As claimed by Loshchilov and Hutter (Loshchilov & Hutter, 2019) in the AdamW paper, one of the key advantages of weight decay decoupling lies in hyper-parameter search. When using the Adam optimizer, the best configurations tend to exhibit diagonal-like patterns on the heatmap, indicating the correlations between the learning rate $\alpha$ and the weight decay factor $\lambda$; whereas the decoupled weight decay formulation allows a direct regularization and thus alleviates such interference to some extent. Accordingly, one can fix one hyper-parameter and find the best value for the other easily. We attach the heatmaps presented by Loshchilov and Hutter in the Appendix (Figure 5).

However, we find some inconsistent results in our replicated experiments, whose search space is a super-set of the search space used in the AdamW paper. Noticing that the test error space in the first row of Figure 3 is much smoother than the ones presented in the original paper. Moreover, we can see that the region with lower test errors are largely resemble to each other, and one can

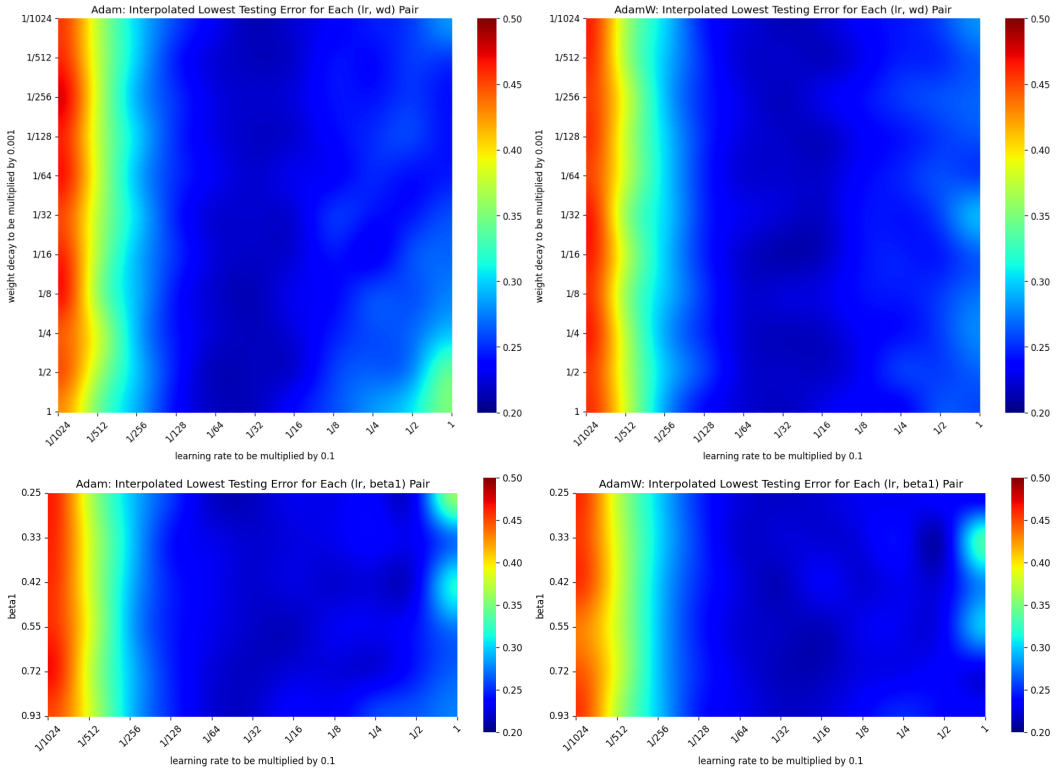



Figure 3: Replicated heatmaps using grid search.

easily find the best learning rate after fixing some weight decay factors using any of them. The only advantage of AdamW shown in our experiment is that its best configuration indeed converges to a slightly lower test error compared to that of Adam. We note that such inconsistent results might be caused by the models and/or metrics, as we used a different ResNet version for experiments and the range of our test error we got does not match with the AdamW paper. Nonetheless, it still is natural to question the true effect of weight decay decoupling, as our experiments have shown that **the performance gap between Adam and AdamW is quite minor**.

### 4.3 Potential of Decoupling Other Hyper-parameters

After noticing that weight decay decoupling have minor improvement, one follow-up question to ask is thus: **Does decoupling work on other hyper-parameters?** In this section, we explore the potential of decoupling the first moment estimate factor $\beta_1$ and the learning rate. To do so, we also conduct grid search over the first moment estimate factor $\beta_1$ and the initial learning rate $\alpha$ to see if we can reproduce the correlated patterns shown in Figure 5 as a starting point.

We follow the same experiment setting as in Section 4.2 and use $\beta_1$ within the log-uniform space of $(0.25, 0.93)$. We present the heatmaps in the second row of Figure 3. From the figure, we can see that the heatmaps are still smooth and highly resemble as in the grid search of learning rate versus weight decay, and the best configuration in AdamW also converges to a slightly lower test error compared to that of Adam. As a result, we do not proceed in formulating the decoupled update rule and running further experiments. Instead, we argue that **the expected improvement of decoupling hyper-parameters in the update step may have been over-estimated**.

## 5 Conclusion

We summarize our findings as follows: 1) different weight decay formulations do not have a significant impact when applying learning rate annealing; 2) the performance gap between AdamW and Adam is lower than expected; 3) the true effect of hyper-parameter decoupling is questionable.

# APPENDIX

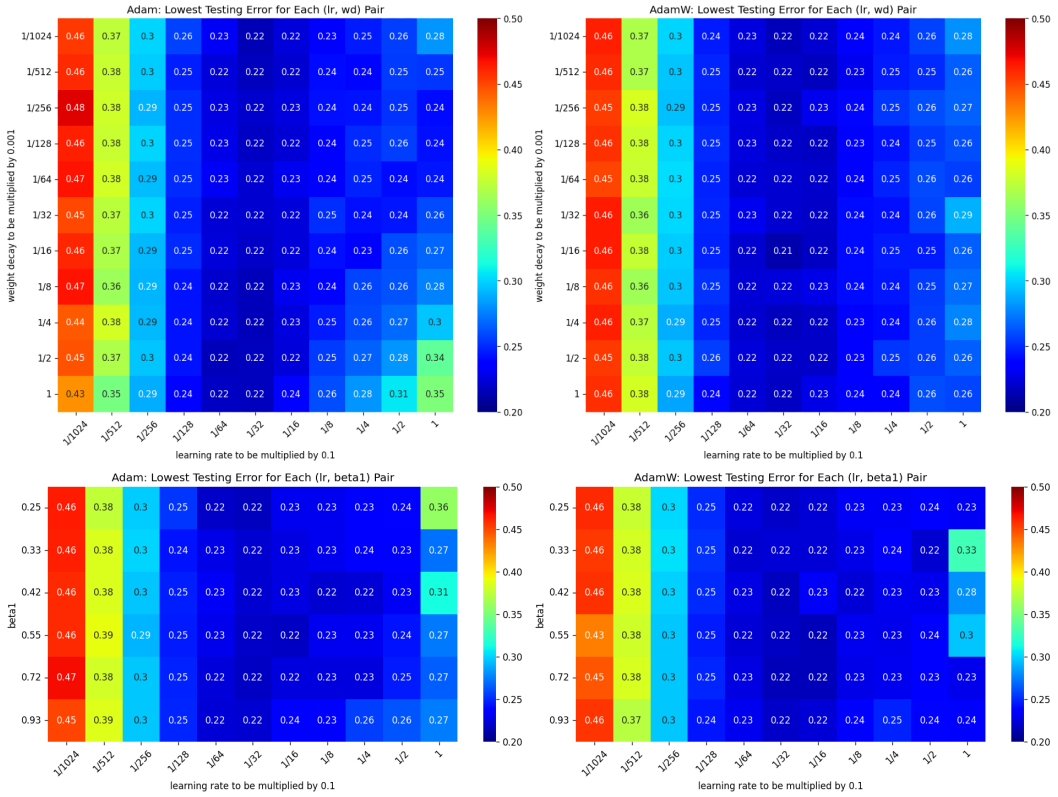



Figure 4: Grid search results before interpolation.

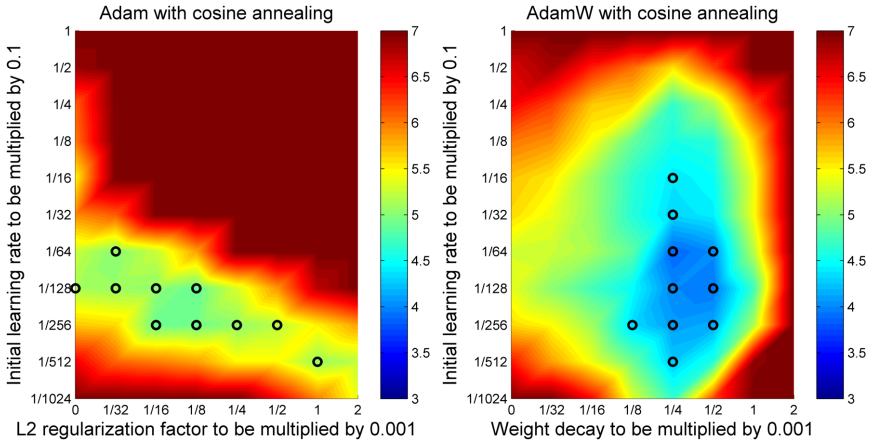



Figure 5: Heatmaps presented in the AdamW paper (Loshchilov & Hutter, 2019).

## LIMITATION

We only conducted experiments on the image classification task using a single dataset and a single model, so the conclusions we drawn may not be generalizable to other tasks, datasets, and models. A more comprehensive evaluation can be left as future work.

## ACKNOWLEDGEMENT

This work is supported in part by NYU High Performance Computing resources and services.

## REFERENCES

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

Soham De, Anirbit Mukherjee, and Enayat Ullah. Convergence guarantees for rmsprop and adam in non-convex optimization and an empirical comparison to nesterov acceleration. In *ICLR*, 2019.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR*, 12(7), 2011.

Stephen Hanson and Lorien Pratt. Comparing biases for minimal network construction with back-propagation. In D. Touretzky (ed.), *Advances in Neural Information Processing Systems*, volume 1. Morgan-Kaufmann, 1988.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. *Cited on*, 14(8):2, 2012.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021.

Zeke Xie, Issei Sato, and Masashi Sugiyama. On the overlooked pitfalls of weight decay and how to mitigate them: A gradient-norm perspective. In *Advances in Neural Information Processing Systems*, 2023.