We would like to thank the reviewers for their helpful feedback; we will use it to significantly improve our manuscript.

Reviewer 2

We would like to give a high level summary of our paper for clarity. Neural networks are hard to train and techniques, like skip connections and batch normalization, have been developed to make training easier, but only work in some cases and add complexity to the architecture. Simultaneously, there has been recent (and not so recent) theoretical work showing that properly scaling the weights in a network at initialization can significantly improve learning. Unfortunately the "correct" weight scale is highly architecture dependent. The main goal of this work is to automate this process.

1. *"Q2. I don't understand the protocols you used in experiments. [...] Is it natural to compare random init (or other known initialization method) vs. meta-learned init?"*: Since we are trying to automate the process of choosing a good initialization, it is natural and important to compare MetaInit to handcrafted initializations. In addition, we also compare with, to our knowledge, the best automated initialization method called LSUV (Mishkin et al, 2015) in Table 1. We outperform LSUV by at least 2% on CIFAR-10 for residual networks.
2. *"Why do you have to remove skip connections and batch normalization layers?"*: An important question is whether or not the aforementioned architectural tricks are necessary for training models that perform well or can we get by with good initialization alone? Previously, this had been difficult to study since, as shown in Table 1, performance on vanilla architectures using traditional initialization schemes does not succeed. However, using MetaInit we are able to substantially close the gap and compare architectural features directly. We believe being able to do this efficiently is necessary to make machine learning more rigorous.
3. *"Q1. [...] Even if you have no theoretical proof, there should be an experimental support"'*: We would like to point out that there is a large body of work relating initialization to trainability dating back at least to (Glorot and Bengio, 2010) and an even longer history relating Hessian conditioning to learning (Nocedal et al, 1998).

   We do provide strong experimental support that the gradient deviation is a good metric. Table 1 and 2 show that minimizing the gradient deviation can improve the performance up to 60% for some models, even on Imagenet. However, we agree with the referee that can be even more explicit in showing that gradient deviation is strongly correlated with test-time performance. To that end we will add gradient deviation measurements along with test error to all of our tables. For example,

Model	Method	Test Error (%)	Gradient Deviation
WideResnet 204-4	DeltaOrthogonal	6.7	2.29
	MetaInit	3.4	0.50

Reviewer 3

Thanks for your feedback, we will add the requested experiments.

1. *"However, why not also show results for non-random data? The paper also mention it operates in 'data-agnoistic' fashion, what advantages does it bring to be data agnostic?"*: We confirm that the algorithm also works well with non-random data and we will add an experiment to that effect. Non-random labelled data is more costly than random data so it is useful when a method can work without it (i.e. for few-shot learning).
2. *"equation 2 is [..] not necessarily [looking at] the curvature"*: Equation 4 shows how the gradient deviation is related to the curvature (eigenvalues) of the Hessian. The gradient deviation is upper-bounded by a weighted average of the eigenvalues, with higher weights in the dimension where the gradient is higher. Note, that the gradient deviation does not go to zero if the magnitude of the gradient vanishes.
3. "What happens if metainit is combined with batchnorm?": We find that MetaInit still works with batchnorm, we will add these experiments.
4. *"Can you show a training error plot as function of update iterations?"*: We will add these curves, which show it usually converges much faster than the baseline.

Reviewer 1

1. *"Derive the analytical scaling factors [...] to compare with classic initializations (Xavier, Kaiming)"*: Figure 2 compares the scaling factors found by MetaInit with those found by Xavier and Fixup initialization. These experiments show that the MetaInit scaling factors match those of Xavier for architectures where Xavier initialization is appropriate. We will include a figure comparing with Kaiming in the appendix.
2. *"The related work section seems a bit thin"*: We plan to use part of the additional space allotted to expand the related work to include MAML, hyper-gradients and traditional initializations.
3. *"Compute the criterion using finite-difference instead of Taylor expansion and see which is better."*: Thank you for the suggestion, we have started comparing the two approaches and will include a discussion.