General Comments. We thank the reviewers for their valuable feedback. As the reviewers point out, the deep equilibrium model offers a new perspective on deep networks. Instead of actually creating a deep stack of layers, the central idea of this paper is to develop an alternative view of deep learning where we directly optimize for and backpropagate through an equilibrium state of the network (which, to the best of our knowledge, no deep approaches have explored or targeted to date, and similar ideas such as the Neural ODE differ significantly in their formulation). The way DEQ "ignores" depth and solves for the equilibrium suggests a different view of output modeling and further leads to certain interesting properties beyond the obvious reductions in memory footprint (cf. Theorem 1 and 2).

Importantly, compared to prior implicit-depth approaches such as Neural ODEs, in this work we also demonstrate the potential power and applicability of such models on practical, large-scale and high-dimensional datasets. In fact, we are able to get 24.0 ppl using a slightly larger DEQ-Transformer than in Table 3, which outperforms the current SOTA result that can run on GPUs (these results will be reflected in the revision). We believe that the equilibrium view of deep learning could lead to many directions of research, in both designing better sequence models (e.g., via better-designed f_θ, see Theorem 2) and studying the properties of the equilibrium optimization.

We also agree with the reviewers that the runtime discussion should be moved into the main text. We briefly include some of our observations below and will have a more thorough analysis of the relationship between threshold ε, training/inference speeds, and modeling accuracy in the experiment section of the revision. We now address specific questions/comments raised by each reviewer.

Review #1. We thank reviewer #1 for the valuable feedback. As we highlighted in the general comments above, the DEQ approach is very different from techniques like gradient checkpointing (GC). In essence, GC enables training an L-layer network using $O(\sqrt{L})$ memory, without actually affecting the computations themselves (GC only recomputes certain blocks). It is an implementation-based methodology that is practical on almost any layer-based network. On the other hand, continuous/implicit-depth models such as Neural ODE and DEQ reduce memory requirements by *formulation* rather than implementation, as these models usually come from certain black-box solvers and analytical backpropagation.

Quantitatively, we have followed the reviewer's suggestion and compared GC and DEQ using a 70-layer TrellisNet (w/ aux. loss, etc.) on WT103. We find that GC works best when we checkpoint after every 9 layers, and record a 5.2GB memory footprint at training time under these conditions. This is 57% more than the DEQ memory footprint (see Table 3). The training speed of GC is approximately 1.6× slower than original training, while DEQ can be up to 2.4× slower (this is an updated result, see our response to reviewer #3 for more details). More fundamentally, though, we should emphasize GC offers $O(\sqrt{L})$ memory consumption while DEQ is $O(1)$. (And recall we are dealing with $L \to \infty$ ...)

Review #3. We thank reviewer #3 for the comments, and for taking the time to check our proof and read our code. We also feel that DEQ provides a new and exciting direction for further designing better implicit-depth models as well as exploring the properties of equilibrium training.

We originally picked the values of ε just to ensure that we get a fixed point that is as accurate as possible under the superlinear convergence of quasi-Newton methods. However, since the submission we have further observed that the conclusion from Figure 4 also holds in training. By using larger ε or a smaller iteration limit, we find that the model can be trained much faster with only a small degradation in performance (e.g., we get 24.3 ppl on WT103 with DEQ-Transformer by limiting the max # of Broyden iterations to 35; the same model can yield 24.0 ppl using a smaller ε). We generally find that $\varepsilon < 0.01$ is sufficient. With that observation, the DEQ training time is now around 2-2.4× that of the original networks (see Table 4) without materially affecting accuracy. We are running more settings and will provide a detailed discussion of this (and some caveats) in the revision.

Regarding initialization, we find that most commonly used initialization schemes with small values (around 0) should suffice. It is important to ensure that the model starts with a small operator norm in the weight matrices. DEQ is not sensitive to any specific initialization scheme because non-linearities such as σ/tanh and LayerNorm help make f_θ contractive (and stable). We are happy to discuss this in the revision.

Review #4. We thank reviewer #4 for the comments. DEQ essentially provides a way to model a deep network at its infinite limit. Deep learning research indicates that more layers frequently lead to better results, and DEQ provides a way to explore the limits of layer stacking without paying an exorbitant price in memory or computation. As highlighted in our response to reviewer #3, we can also further reduce the training/inference cost by less accurate (but still good enough) fixed-point estimation using larger ε, which can be an interesting topic for further research. In addition, while we did not observe any "unstable" fixed points empirically, we believe it is important to ensure that the transformation f_θ itself is stable and contractive (e.g., ideally, having J_{f_θ} operator norm less than 1 would be a sufficient condition).