LINEAR MODE CONNECTIVITY AND TRANSFORMERS

James Song
University of Michigan
shxjames@umich.edu

ABSTRACT

Despite the highly complex loss landscapes that often occur in neural network training, there has been interesting phenomena and patterns that are shown empirically. Linear Mode Connectivity (LMC) is one of those phenomena due to the observation that different solutions reached by Stochastic Gradient Descent can be connected with a linear path. In this work, we study whether LMC occurs in Transformers. We provide empirical evidence across decoder-only transformers with varying sizes and model configurations, which follows GPT-2 (Radford et al., 2019; Brown et al., 2020). We demonstrate empirically that transformers are stable during training and unstable during initialization, which means linear mode connectivity occurs when we train or fine-tune a pretrained model. The code is available at `https://github.com/james-hx-song/lmc-transformers`.

1 INTRODUCTION AND RELATED WORKS

When training a neural network with mini-batch stochastic gradient descent (SGD), batches of training samples are used by the network in a random order within an epoch. This randomness can be seen as SGD noise that alters the trajectory of the optimization process. Frankle et al. (2019) studies this process through **Instability Analysis**, which determine whether the outcome of optimizing neural nets is *stable* to SGD noise. There are two main ways they do so:

- **Instability Analysis at Initialization:** Train 2 copies of a randomly initialized network with different orders of SGD batches. In other words, the 2 networks are trained with different samples of SGD noise. Then, linearly interpolate the minima found by the 2 copies and observe the train and test errors.
- **Instability Analysis during Training:** Train a randomly initialized network, then make 2 copies and train them with different orders of SGD batches.

By linear interpolating the networks at the end of training, Frankle et al. (2019) assesses a linear form of *mode connectivity*, where the 2 solutions found are connected by a path of non-increasing error.

In this work, we mainly study instability and specifically linear mode connectivity (LMC) in both settings, and we study LMC in the context of transformers, specifically decoder-only transformers.

2 SETUP

Frankle et al. (2019) uses linear interpolation to determine whether a network is stable to SGD noise, hence exhibits LMC. We do the same. Formally, we denote $\mathbf{Err}(W)$ to be the train or test error of a network with weights W. Let $\mathbf{Err}_{\sup}(W_A, W_B) = \sup_\alpha(\mathbf{Err}(\alpha W_A + (1-\alpha)W_B)$. This quantity denotes the largest error when interpolating the two networks A, B. Finally, let $\mathbf{Err}_{\text{mean}}(W_A, W_B) = \frac{\mathbf{Err}(W_A)+\mathbf{Err}(W_B)}{2}$. We are interested the difference $\mathbf{Err}_{\sup}(W_A, W_B) - \mathbf{Err}_{\text{mean}}(W_A, W_B)$. If the difference is close to 0, the two networks are linearly mode connected (Frankle et al., 2019).

In more detail, we define the notion of **Linear Mode Connectivity.**

Definition 2.1 (Linear Mode Connectivity)**.** Given a dataset S and two networks with parameters W_A and W_B after training such that $\mathbf{Err}_S(W_A) \approx \mathbf{Err}_S(W_B)$, we say W_A and W_B are linearly connected if they satisfy

$$\mathbf{Err}_S(\alpha W_A + (1-\alpha)W_B \approx \mathbf{Err}_S(W_A)$$

Empirically, Frankle et al. (2019) consider instability $< \ 2\%$ to be stable. They use 0-1 loss as an evaluation of their model's performance. In this work, we use cross-entropy loss as our **Err** function on both train and test dataset.

3 Experiment Details

3.1 Training Details

We train 4 transformer language models with varying configurations on the tiny shakespeare dataset (Karpathy, 2015) due to a limitation in compute. The dataset has The architecture that we implement in this work follows the GPT-2 architecture described by Radford et al. (2019) and their inference code at their github. We follow settings described by Brown et al. (2020) for certain design choices in optimization: we use the Adam Optimizer with $\beta = (0.9, 0.95)$ and we use a learning rate between $6 \cdot 10^{-4}$ and $6 \cdot 10^{-5}$, with a linear warmup schedule and a cosine decay afterwards. However, we did not implement the weight decay or the initialization schemes specified by Radford et al. (2019), as the point of this work is to study the optimization behavior of transformers.

Network	#Param	Tokenizer	Train Steps	Learning Rate	Schedule
TinyGPT	204K	Unique Char Ordering	20k	[6e-5, 6e-4]	Warmup: 2k
ToyGPT	1.65M	GPT-2 tokenizer	15k	[6e-5, 6e-4]	Warmup: 1.5k
CompactGPT	12M	GPT-2 tokenizer	10k	[6e-5, 6e-4]	Warmup: 1k
MinGPT	30M	GPT-2 tokenizer	3k	[6e-5, 6e-4]	Warmup: 300

Table 1: Training configurations for different GPT models. Cosine decay is used immediately after the warmup until training ends for each network. Train steps are somewhat arbitrary; the networks are trained multiple times to see approximately when the error converges. A batch size of 32 is used for all models. For TinyGPT, the tokenizer used is a naive one: each token represents the index of the character in an ordered list of unique characters in the dataset. For more details about configurations, please visit the github repo.

We also finetune a 124 M pretrained GPT-2 on the shakespeare dataset to investigate whether transformers are stable during training or not. Due to limitations in compute, we use a batch size of 8 and finetune for 300 iterations.

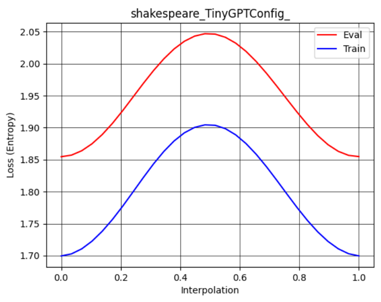
3.2 Dataset

The tiny shakespeare dataset has 338025 tokens with the GPT-2 Tokenizer, and 1.1 million characters in total. We split the dataset into train and test with a 9:1 split. Because the dataset is an order of magnitude smaller than compactGPT and minGPT, they are overparameterized, which means their test loss is high. However, since this work focuses on optimization behavior of SGD, we don't add any regularization such as dropout, which is used by Radford et al. (2019).

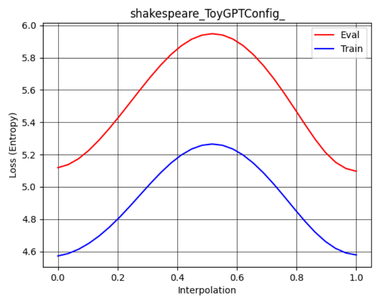
4 Results

4.1 Training from Initialization

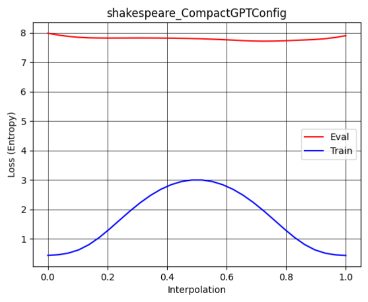
We randomly initialize a transformer and make two copies of the network, then train the two copies with Stochastic Gradient Descent. We find that although none of the networks are stable to SGD noise at initialization, the network is stable when evaluated on the test dataset if the network is heavily overparameterized with respect to the dataset.



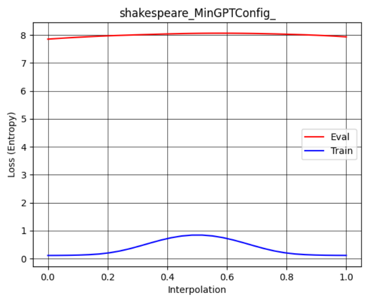
(a) TinyGPT (204K)



(b) ToyGPT (1.65M)



(c) CompactGPT (12M)



(d) MinGPT (30M)

Figure 1: Error when linearly interpolating transformers trained from the same initialization with different SGD noise. Trained networks are at 0.0 and 1.0. Referencing from definition 2.1, we use 30 equally spaced α values, same as the settings by Frankle et al. (2019).

Figure 1 shows the train (blue) and test (red) errors when linearly interpolating between the minimas found by the copies. Looking at the figures, none of the networks are stable at initialization. Interestingly, even though TinyGPT has much fewer parameters compared to the other models, when trained to full capacity, its evaluation cross-entropy loss (Fig. 1(a)) is much lower than that of all the other models. However, we see that the evaluation loss of TinyGPT and ToyGPT display a bell-like curve, which denies linear mode connectivity between the two trained networks. Although the evaluation loss suggests that the solutions for CompactGPT and MinGPT are linearly connected, the train loss does not support this evidence, and instead displays a similar "bell-shaped" curve compared to fig. 1(a) and fig. 1(b). Therefore, we see that all of the transformer networks do not have linear mode connectivity when trained from the same initialized networks.

To further understand the behavior of the evaluation loss on overparameterized networks, we also interpolated the weights of MinGPT early on in training to observe its behavior. As seen in fig. 2, there is clear evidence of instability in the network. The network is not trained to its full extent with respect to its training loss, but even with respect to its evaluation loss, we see that the networks are not linearly connected.

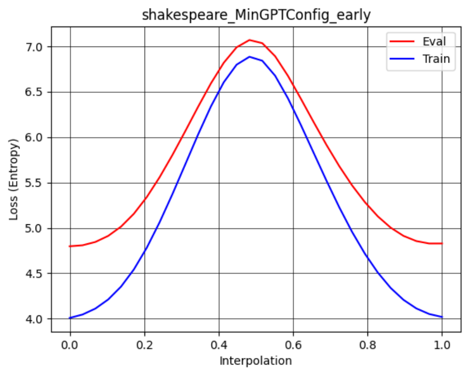


Figure 2: MinGPT Interpolation at Iteration 500

4.2 Finetuning with Pretrained Model

To investigate whether this phenomenon persists for pretrained model, we fine-tune the pretrained model of GPT-2 (124 M) loaded from huggingface Radford et al. (2019).

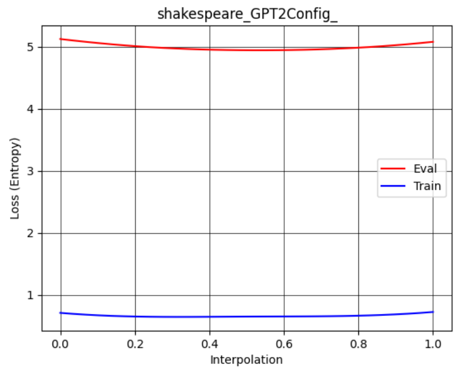


Figure 3: Interpolation for the Finetuned 124 M GPT-2 Model after 300 iterations of training.

After training for 300 iterations, we find that the 2 solutions found by SGD are linearly connected due to the constant loss in the interpolation graph as shown above. Using the terminology from Frankle et al. (2019), we see that transformers (or GPT-2 in this specific scenario) are stable during training, and in particular during the finetuning process.

5 Conclusion

In this work, we explored Linear Mode Connectivity (LMC) in transformers architecture, specifically the decoder-only models like GPT-2. We conducted experiments to understand the stability of these networks with respect to SGD noise at initialization and after pretraining. We find that while transformers are unstable at initialization, they exhibit LMC when fine-tuned from a pretrained model. Additionally, overparameterized networks, though seemingly unstable during the early stages of training and with respect to train error, have nuanced behaviors in their loss profiles: their evaluation loss is nearly linearly connected.

References

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, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey 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. Language models are few-shot learners. *CoRR*, abs/2005.14165, 2020. URL `https://arxiv.org/abs/2005.14165`.

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M. Roy, and Michael Carbin. Linear mode connectivity and the lottery ticket hypothesis. *CoRR*, abs/1912.05671, 2019. URL `http://arxiv.org/abs/1912.05671`.

Andrej Karpathy. char-rnn. `https://github.com/karpathy/char-rnn`, 2015.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.