A ORGANIZATION OF THE APPENDIX

- We describe the training procedure and datasets used throughout the paper in detail in appendix C.
- In appendix D, we reproduce the extreme memorization phenomenon from Figure 2 on CIFAR-100 and SVHN.
- In Section 3.2, Fig 4 shows how changing the scale of activation leads to a drop in generalization performance in the case of ReLU activation and softmax cross-entropy loss. In appendix E, we include results with multi-class hinge and squared losses.
- We include results when using linear activation with softmax cross-entropy loss in appendix F.
- In appendix G, we discuss how Sigmoid function, when used as activation, responds to scaling of initialization.
- Appendix H includes details on the exact architecture and hyperparameters used in Section 5.

B PROOF OF THEOREM 1

In this section we provide the missing proof of Theorem 1, restated below:

Theorem 1. *Suppose each entry of W_1 is initialized via a Gaussian with mean 0 and variance σ^2. Then for any x and x', we have*

$$\left| \mathop{\mathbb{E}}_{W_1} [\langle \phi(W_1 x), \phi(W_1 x') \rangle] \right| \leq h \exp\left(-\frac{\sigma^2 \|x - x'\|^2}{2} \right)$$

Proof. Since each individual row of W_1 is independent, it suffices to prove the statement for $h = 1$. If $x = 0$ the statement is trivially true, so suppose $x \neq 0$. Let $c = \frac{\langle x', x \rangle}{\|x\|^2}$ and let $\Delta = x' - cx$. Notice that $\langle \Delta, x \rangle = 0$ and $\|\Delta\| \leq \|x - x'\|$. We also have

$$W_1 x' = c W_1 x + W_1 \Delta$$

Notice that $W_1 x$ is normally distributed with mean 0 and variance $\sigma^2 \|x\|^2$. Further, $W_1 \Delta$ is normally distributed with mean 0 and variance $\sigma^2 \|\Delta\|^2$. Let A be a mean 0 random variable with variance $\sigma^2 \|x\|^2$ and B be a mean 0 random variable with variance $\sigma^2 \|\Delta\|^2$. Notice that since $\langle \Delta, x \rangle = 0$, the joint distribution $(W_1 x, W_1 x')$ is the same as that of $(A, cA + B)$. Therefore we have:

$$\begin{aligned}
\left| \mathop{\mathbb{E}}_{W_1} [\langle \phi(W_1 x), \phi(W_1 x') \rangle] \right| &= \left| \mathop{\mathbb{E}}_{A,B} [\sin(A) \sin(cA + B)] \right| \\
&= |\mathbb{E}[\sin(A) \sin(cA) \cos(B) + \sin(A) \cos(cA) \sin(B)]| \\
&= |\mathbb{E}[\sin(A) \sin(cA) \cos(B)]| \\
&= |\mathbb{E}[\sin(A) \sin(cA)] \, \mathbb{E}[\cos(B)]| \\
&\leq |\mathbb{E}[\cos(B)]| \leq \exp\left(-\frac{\sigma^2 \|\Delta\|^2}{2} \right)
\end{aligned}$$

□

C DESCRIPTION OF THE TRAINING PROCEDURE AND DATASETS

We use the Tensorflow framework for conducting our empirical study and all of our code is included as part of supplementary material. In every experiment, we train using SGD, without momentum, with a constant learning rate of 0.01 and batch size of 256. We employ a p100 single-instance GPU for each training run. For most of the experiments, the model is trained until it obtains perfect accuracy on the training set, with only a few exceptions which are either unavoidable or requires extravagant training iterations. For example, in the experiments involving linear activation, since none of the datasets we use are completely linearly separable, we do not expect the net to get 100% accuracy on

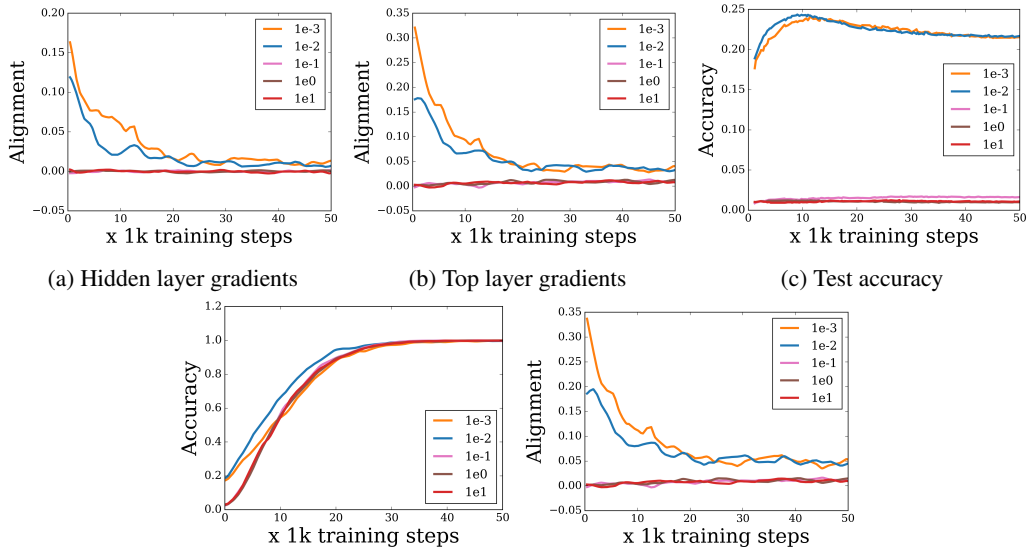
the training set. Another interesting case is the Sigmoid activation, for which the gradients starts to saturate as the scale of the input to the Sigmoid function increases. Thus, we stop the training at a point when at least one of the model in the study achieves perfect accuracy on the training set.

In our 2-layer MLP model, in almost all cases we use 1024 units for the hidden layer with exceptions of 1) experiments with Sigmoid activation and 2) ReLU activation with squared loss. In both of these cases, we increase the number of hidden units to 2048 in order to increase their training speed. Number of units for the softmax layer depends on the number of output classes, which is 10 for CIFAR-10 / SVHN, and 100 for CIFAR-100. The details of the ConvNet architecture are included in appendix H. For any layer that doesn't involve changing the initialization scale, for instance the top layer in all our models, defaults to using Glorot uniform initializer (Glorot & Bengio, 2010). For experiments corresponding to Sections 3 and 3.2, we refrain from employing bias variables in order to match the setup exactly. For experiments in Section 5, all biases are initialized to zero.

We employ 3 image classification datasets each having 32x32 pixels color image as input. CIFAR-10 dataset (Krizhevsky, 2009) consists of 60000 images with 10 classes. Classes are balanced with 6000 images per class. Training set consists of 50000 images and 10000 test images. CIFAR-100 (Krizhevsky et al.) is very similar to CIFAR-10 except that it has 100 classes with 600 images per class. Finally, The Street View House Numbers (SVHN) Dataset (Netzer et al., 2011) has images of digits from house numbers obtained from Google Street View with a total of 10 classes. Training set contains 73257 images and 26032 test images.

D SIN ACTIVATION

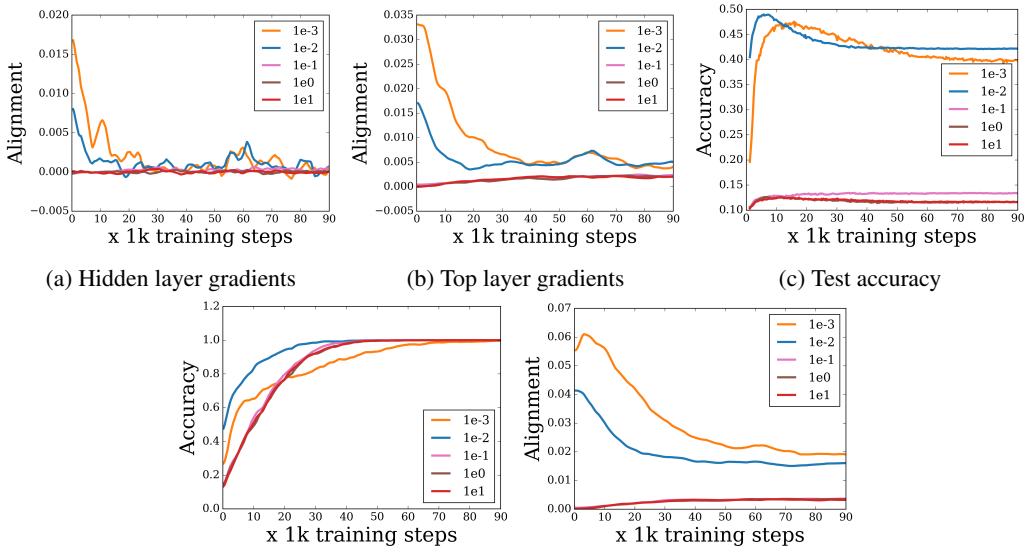
Figure 2 shows that increasing the scale of initialization for hidden layer weights W_1 in a 2-layer MLP model leads to extreme memorization on CIFAR-10 dataset. Keeping everything else the same, we reproduce the same phenomenon on two other datasets, namely, CIFAR-100 and SVHN respectively.



(a) Hidden layer gradients (b) Top layer gradients (c) Test accuracy

(d) Train accuracy (e) Hidden layer activations

Figure 6: Results when using sin activation function on CIFAR-100 dataset.



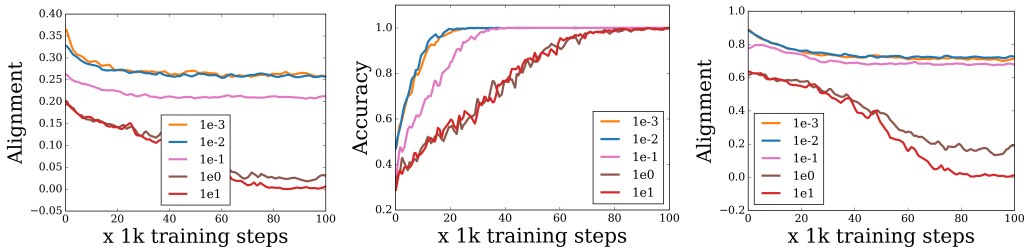
(a) Hidden layer gradients (b) Top layer gradients (c) Test accuracy

(d) Train accuracy (e) Hidden layer activations

Figure 7: Results when using sin activation function on SVHN dataset.

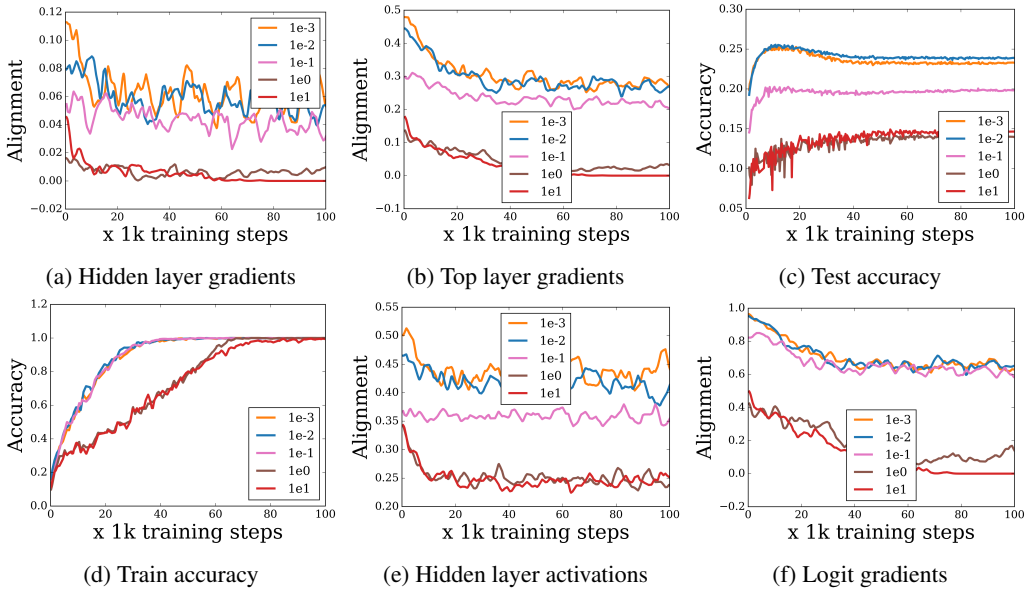
E RELU ACTIVATION

E.1 SOFTMAX CROSS ENTROPY



(a) Top layer gradients (b) Train accuracy (c) Logit gradients

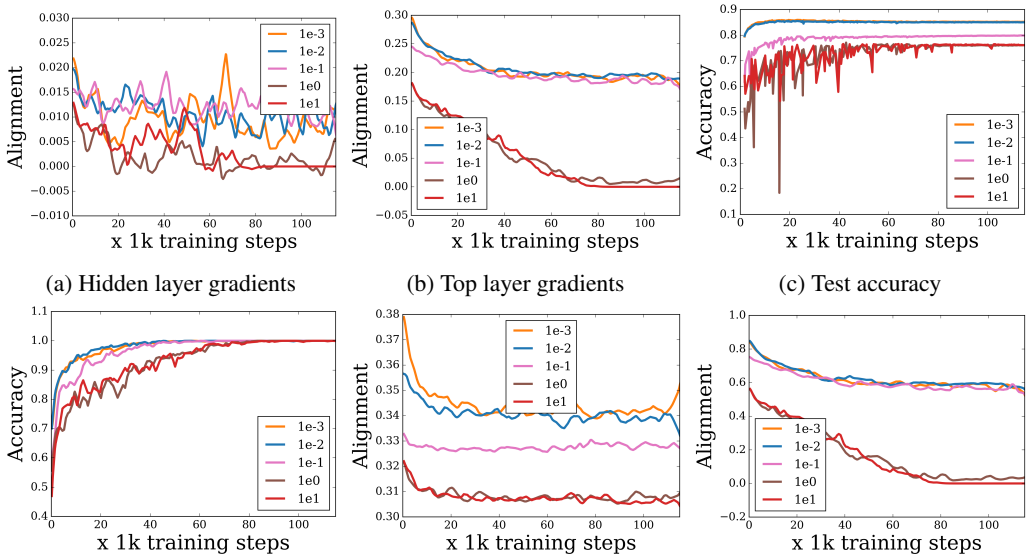
Figure 8: Additional plots when using ReLU activation function with softmax cross-entropy on CIFAR-10 dataset.



(a) Hidden layer gradients (b) Top layer gradients (c) Test accuracy

(d) Train accuracy (e) Hidden layer activations (f) Logit gradients

Figure 9: Results when using ReLU activation function with softmax cross-entropy on CIFAR-100 dataset.

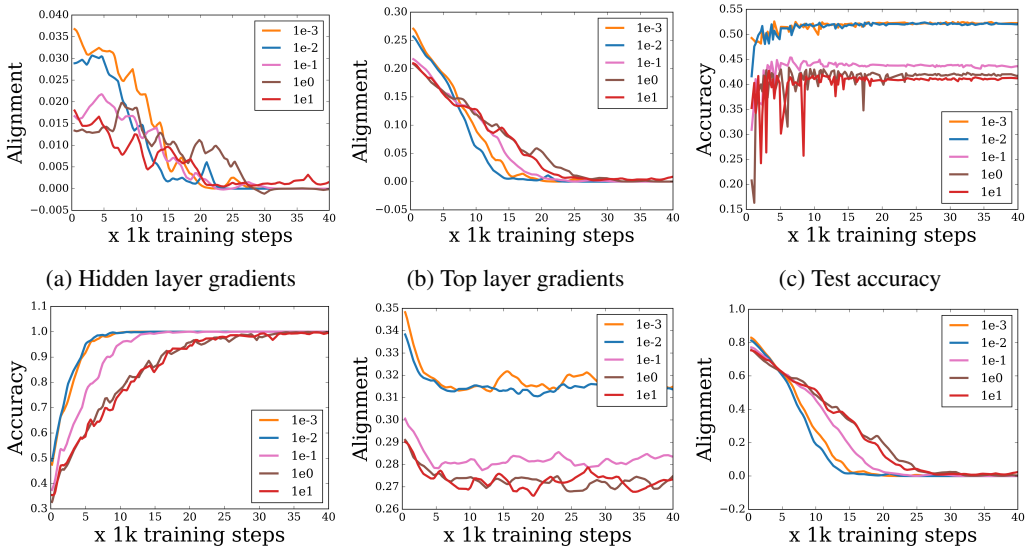


(a) Hidden layer gradients (b) Top layer gradients (c) Test accuracy

(d) Train accuracy (e) Hidden layer activations (f) Logit gradients

Figure 10: Results when using ReLU activation function with softmax cross-entropy on SVHN dataset.

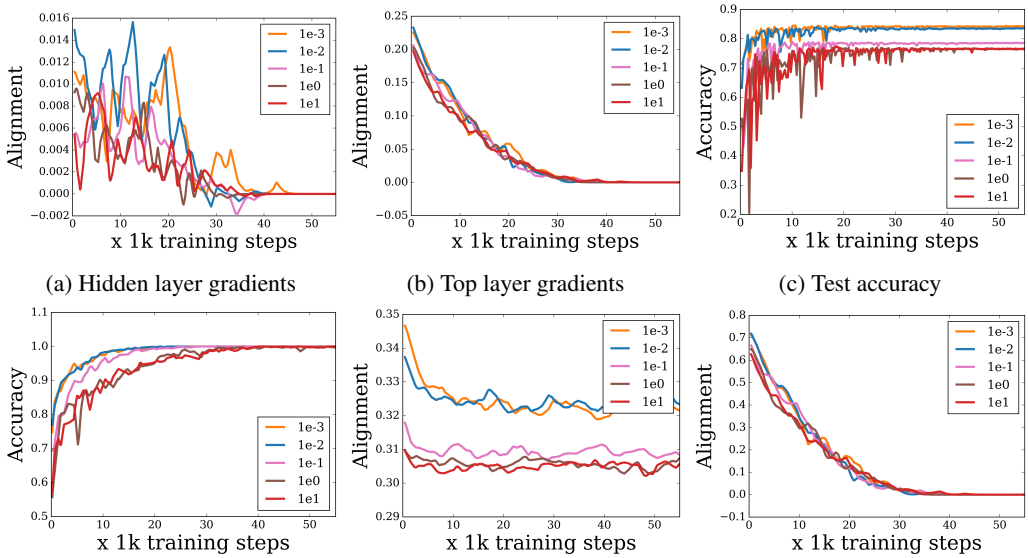
E.2 HINGE LOSS



(a) Hidden layer gradients (b) Top layer gradients (c) Test accuracy

(d) Train accuracy (e) Hidden layer activations (f) Logit gradients

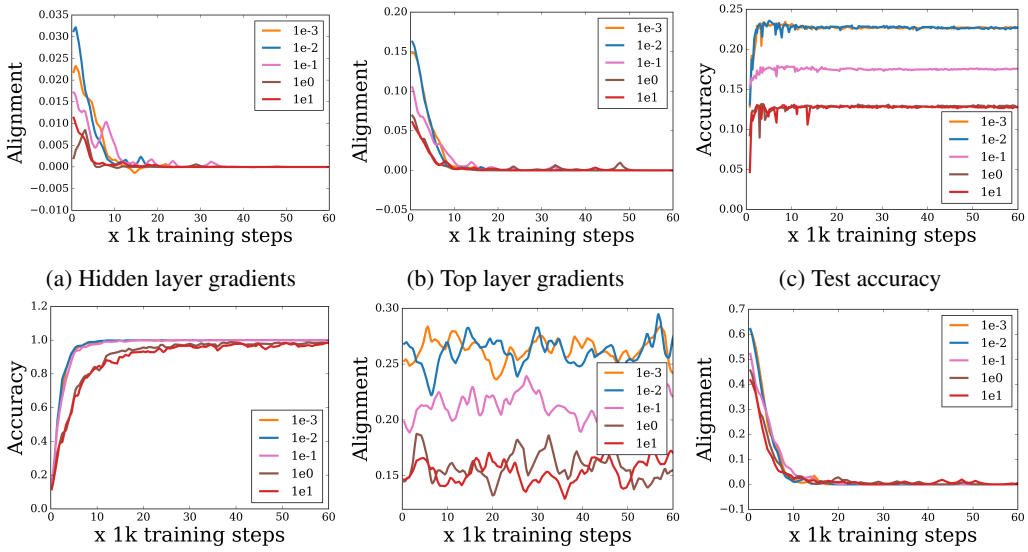
Figure 11: Results when using ReLU activation function with hinge loss on CIFAR-10 dataset.



(a) Hidden layer gradients (b) Top layer gradients (c) Test accuracy

(d) Train accuracy (e) Hidden layer activations (f) Logit gradients

Figure 12: Results when using ReLU activation function with hinge loss on SVHN dataset.



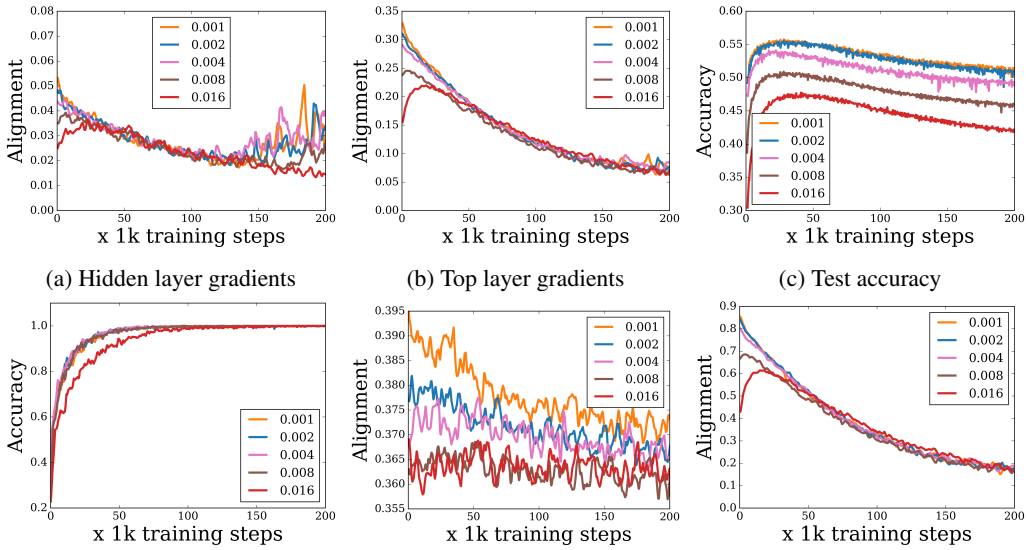
(a) Hidden layer gradients (b) Top layer gradients (c) Test accuracy

(d) Train accuracy (e) Hidden layer activations (f) Logit gradients

Figure 13: Results when using ReLU activation function with hinge loss on CIFAR-100 dataset.

E.3 SQUARED LOSS

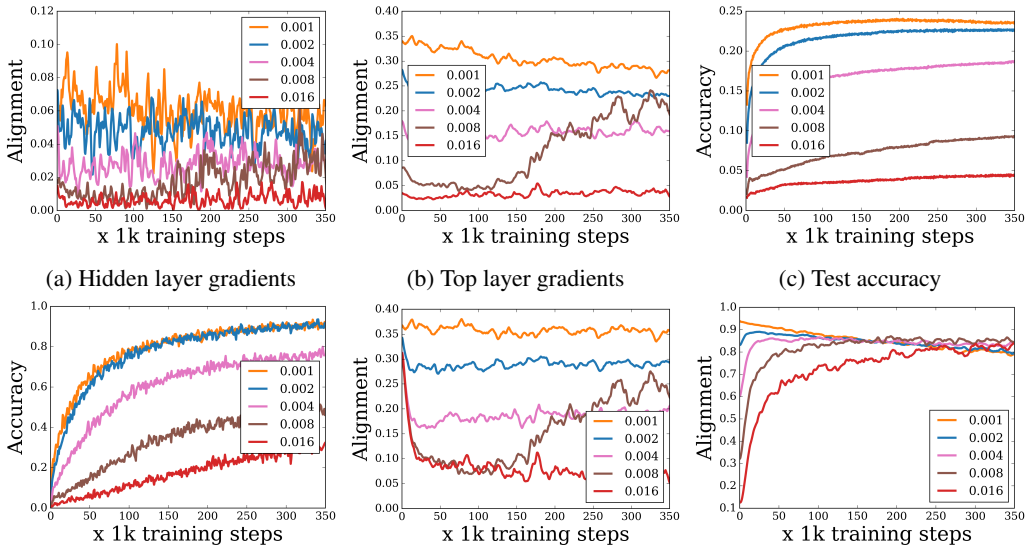
Note that when employing squared loss, we increase the number of hidden units from the usual 1024 units to 2048 in order to compensate for very low training speed. Also, we observed that increasing the scale of initialization for W_1 beyond a certain scale leads to divergence in training after a few iterations. Thus, we recover the phenomenon of interest with much less aggresive increase in scale of initialization i.e. we double the standard deviation instead of increasing it by ten times as done in other experiments.



(a) Hidden layer gradients (b) Top layer gradients (c) Test accuracy

(d) Train accuracy (e) Hidden layer activations (f) Logit gradients

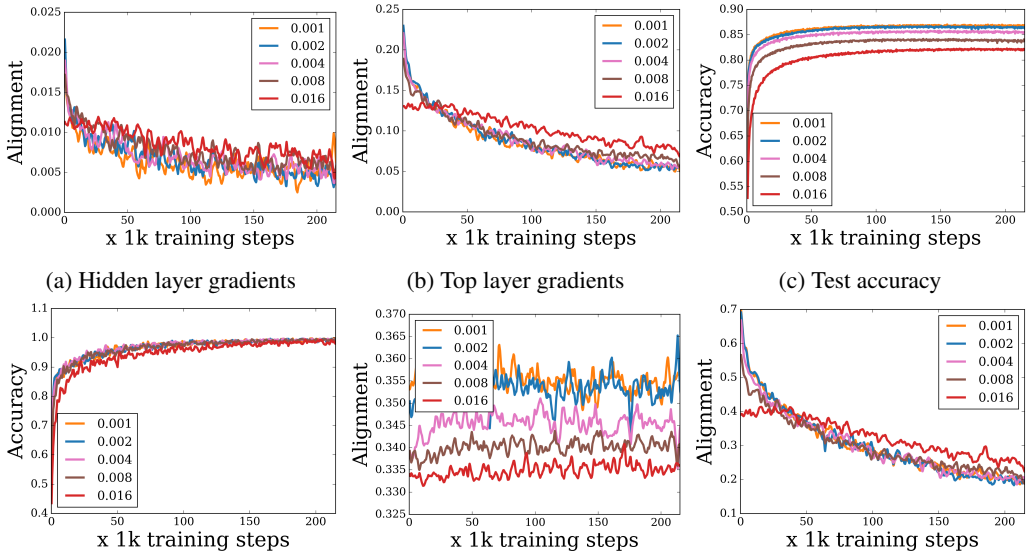
Figure 14: Results when using ReLU activation function with squared loss on CIFAR-10 dataset.



(a) Hidden layer gradients (b) Top layer gradients (c) Test accuracy

(d) Train accuracy (e) Hidden layer activations (f) Logit gradients

Figure 15: Results when using ReLU activation function with squared loss on CIFAR-100 dataset.

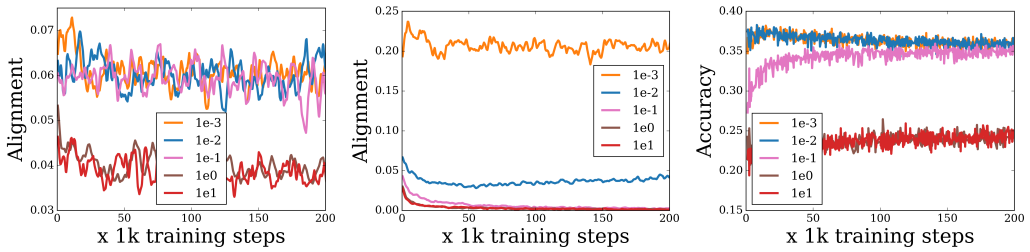


(a) Hidden layer gradients (b) Top layer gradients (c) Test accuracy

(d) Train accuracy (e) Hidden layer activations (f) Logit gradients

Figure 16: Results when using ReLU activation function with squared loss on SVHN dataset.

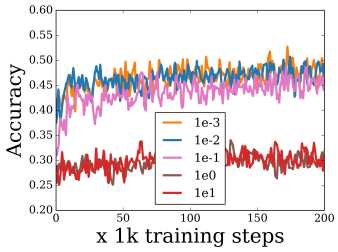
F LINEAR ACTIVATION



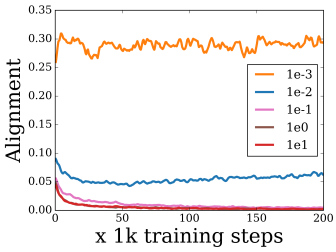
(a) Hidden layer gradients

(b) Top layer gradients

(c) Test accuracy

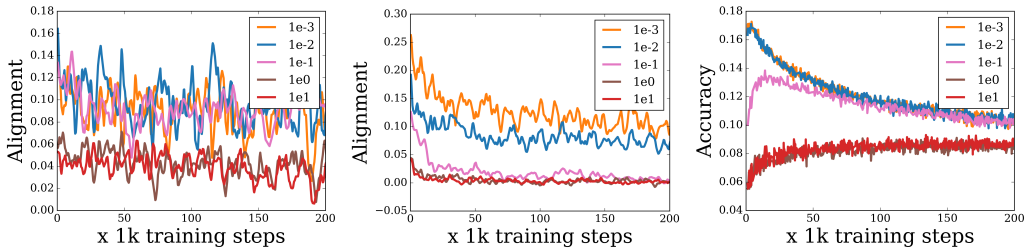


(d) Train accuracy



(e) Hidden layer activations

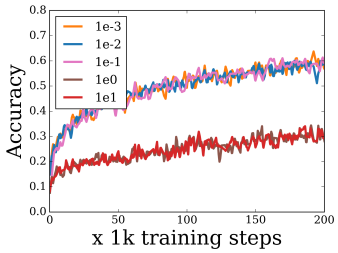
Figure 17: Results when using linear activation function with softmax cross entropy loss on CIFAR-10 dataset.



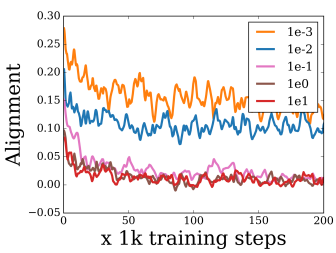
(a) Hidden layer gradients

(b) Top layer gradients

(c) Test accuracy

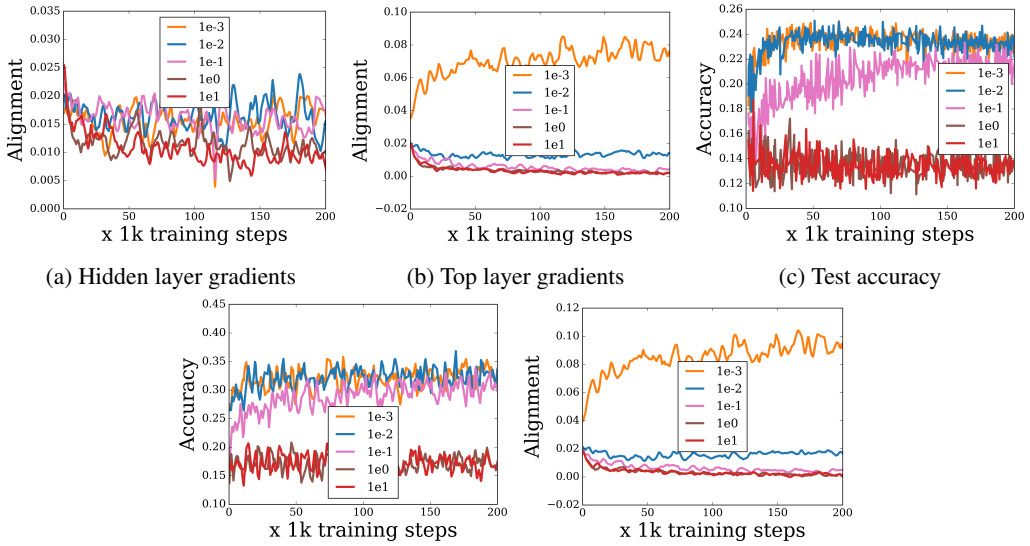


(d) Train accuracy



(e) Hidden layer activations

Figure 18: Results when using linear activation function with softmax cross entropy loss on CIFAR-100 dataset.



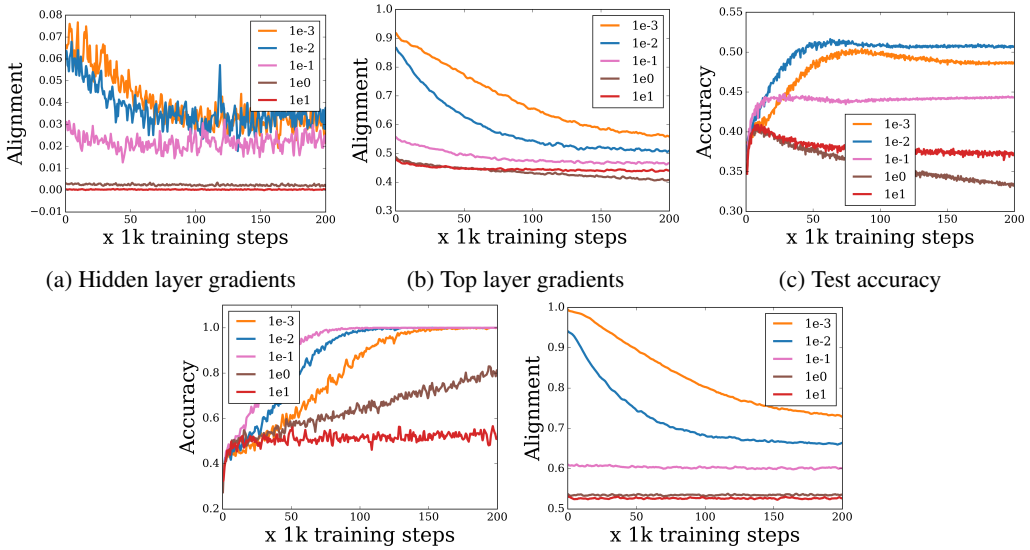
(a) Hidden layer gradients (b) Top layer gradients (c) Test accuracy

(d) Train accuracy (e) Hidden layer activations

Figure 19: Results when using linear activation function with softmax cross entropy loss on SVHN dataset.

G SIGMOID ACTIVATION

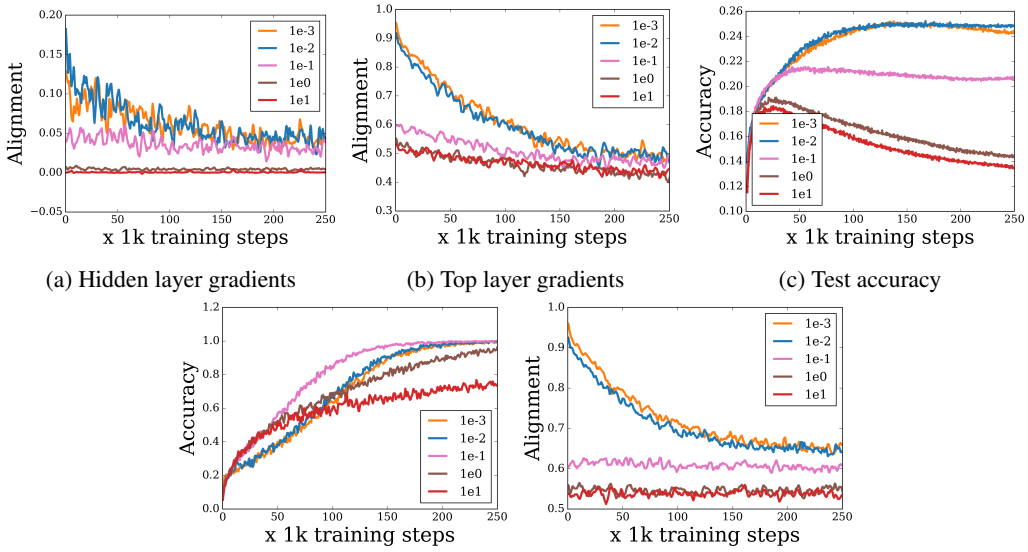
Note that when employing Sigmoid activation, after a certain scale the hidden layer gradients start to vanish. We try to compensate for this by increasing the number of hidden units to 2048.



(a) Hidden layer gradients (b) Top layer gradients (c) Test accuracy

(d) Train accuracy (e) Hidden layer activations

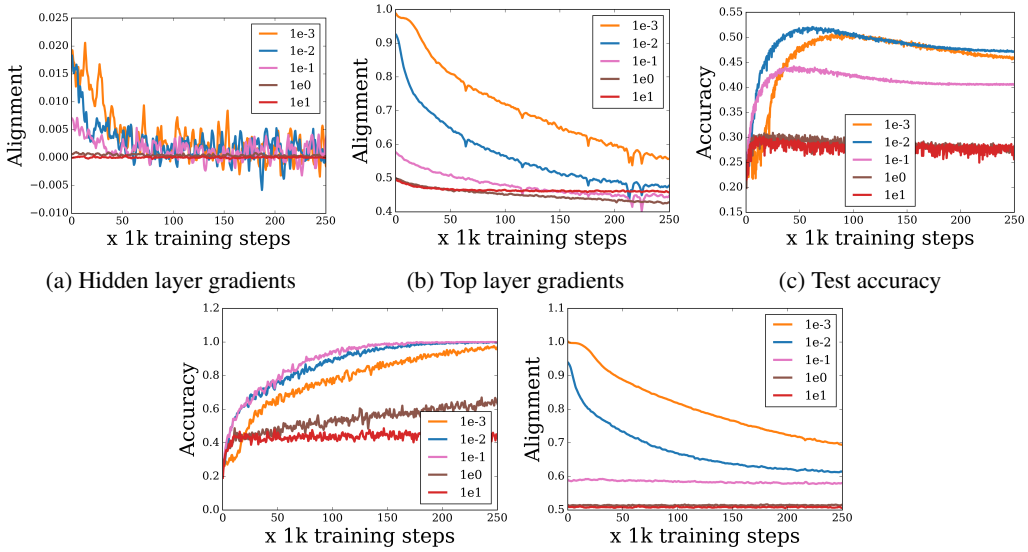
Figure 20: Results when using Sigmoid activation function on CIFAR10 dataset.



(a) Hidden layer gradients (b) Top layer gradients (c) Test accuracy

(d) Train accuracy (e) Hidden layer activations

Figure 21: Results when using Sigmoid activation function on CIFAR100 dataset.



(a) Hidden layer gradients (b) Top layer gradients (c) Test accuracy

(d) Train accuracy (e) Hidden layer activations

Figure 22: Results when using Sigmoid activation function on SVHN dataset.

H CONVNET ARCHITECTURE

Our goal is to recover the phenomenon that convolution and pooling operation leads to more aligned representations. In order to do this, we construct a simple ConvNet architecture. We start with two consecutive convolution and max pool operations, followed by a fully-connected and softmax layers. Note that the last two layers fully connected and softmax are operationally the same as our 2-layer MLP.

We keep all the other hyper parameters the same between training runs for all the architectures. All the models are trained with SGD without momentum with learning rate set to 0.01 and batch size to 256.

- Convolution layer 1: 32 filters with 5x5 kernel size followed by ReLU activation.
- Max pooling layer 1: pool size 3x3 with stride of 2x2
- Convolution layer 2: 64 filters with 5x5 kernel size followed by ReLU activation.
- Max pooling layer 2: pool size 3x3 with stride of 2x2
- Fully connected layer with 1024 hidden units followed by ReLU activation.
- Softmax layer with 10 output logits.