Understanding Deep Learning

Chapter 20: Why does deep learning work?

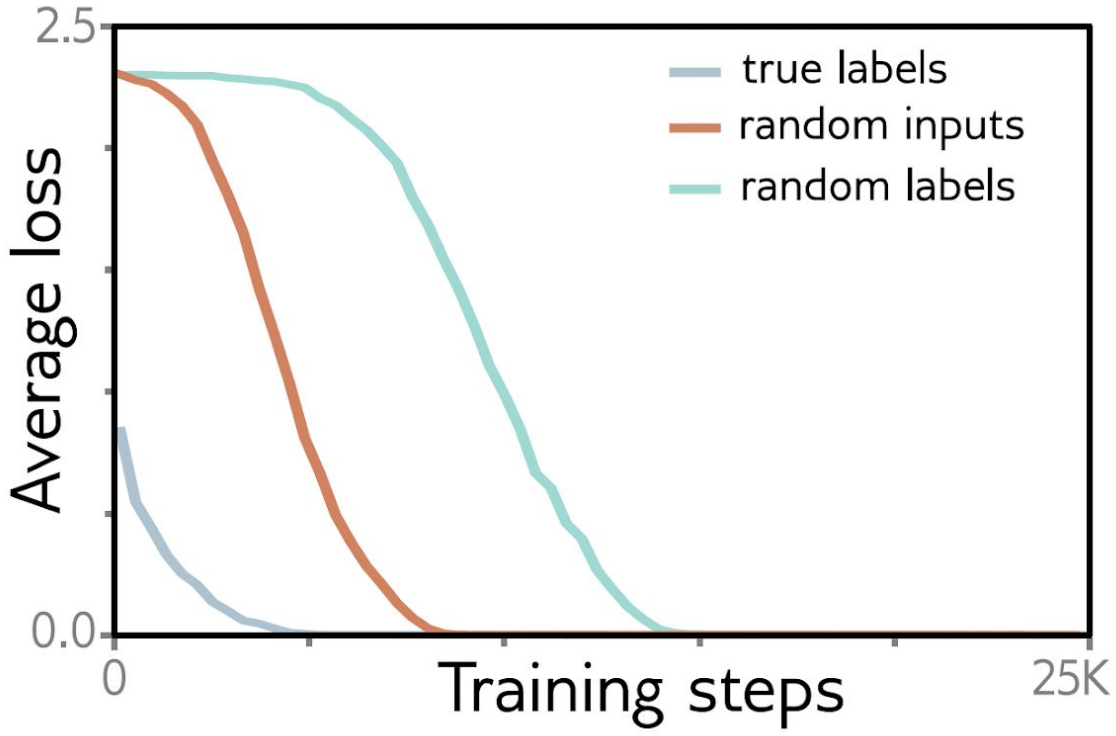
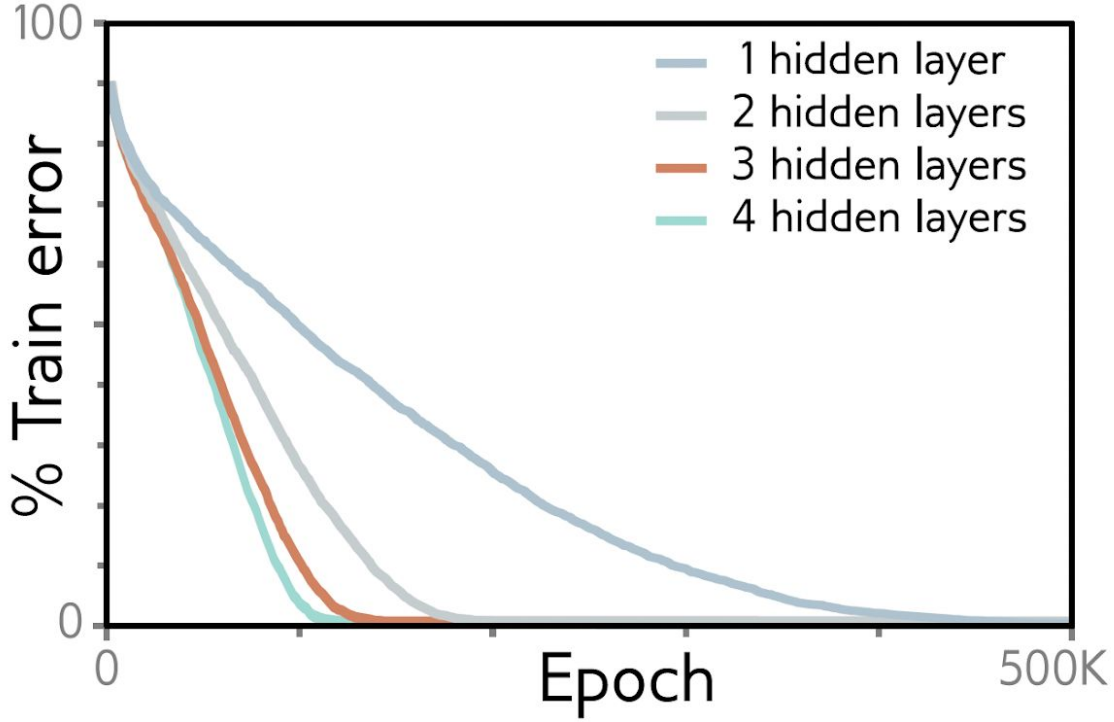


Figure 20.1 Fitting random data. Losses for AlexNet architecture trained on CIFAR-10 dataset with SGD. When the pixels are drawn from Gaussian random distribution with the same mean and variance as the original data, the model can still be fit (albeit more slowly). When the labels are randomized, the model can still be fit (albeit even more slowly). Adapted from Zhang et al. (2017a).

Figure 20.2 MNIST-1D training. Four fully connected networks were fit to 4000 MNIST-1D examples with random labels using full batch gradient descent, He initialization, no momentum or regularization, and learning rate 0.0025. Models with 1,2,3,4 layers had 298, 100, 75, and 63 hidden units per layer and 15208, 15210, 15235, and 15139 parameters, respectively. All models train successfully, but deeper models require fewer epochs.



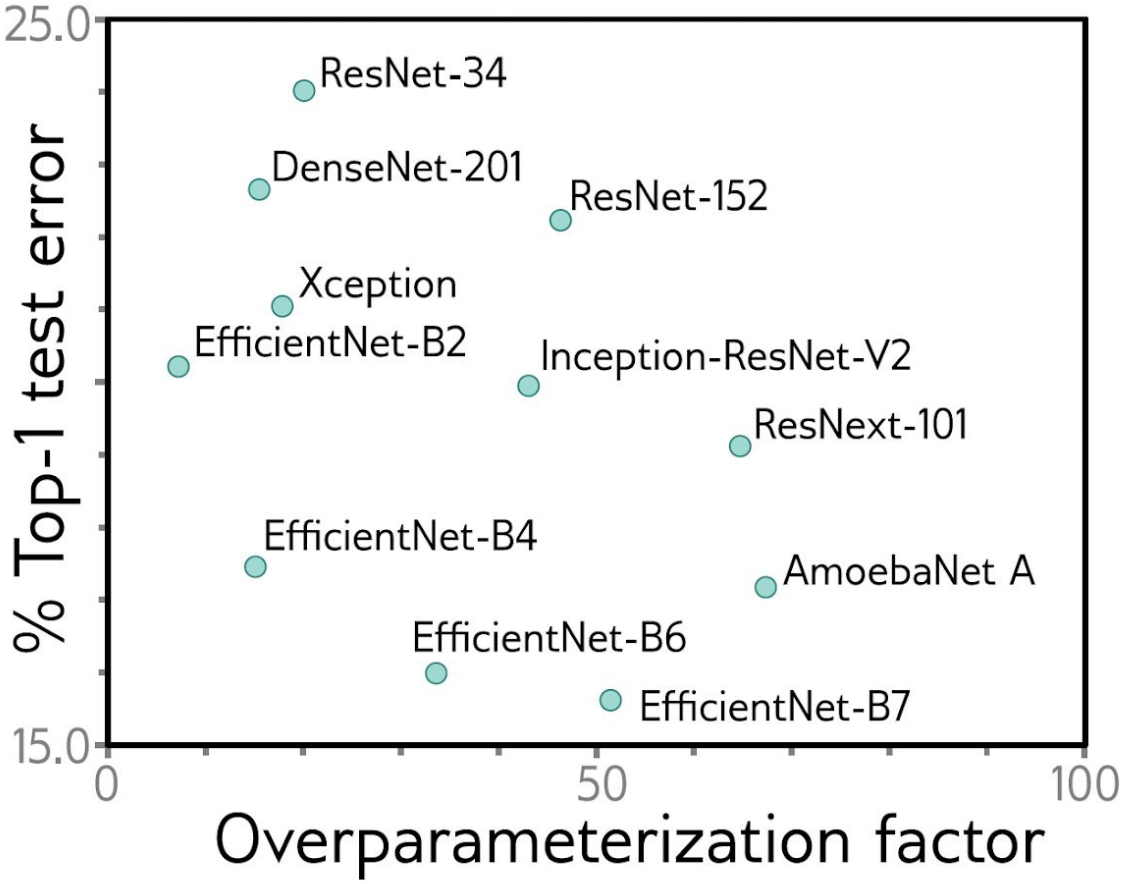
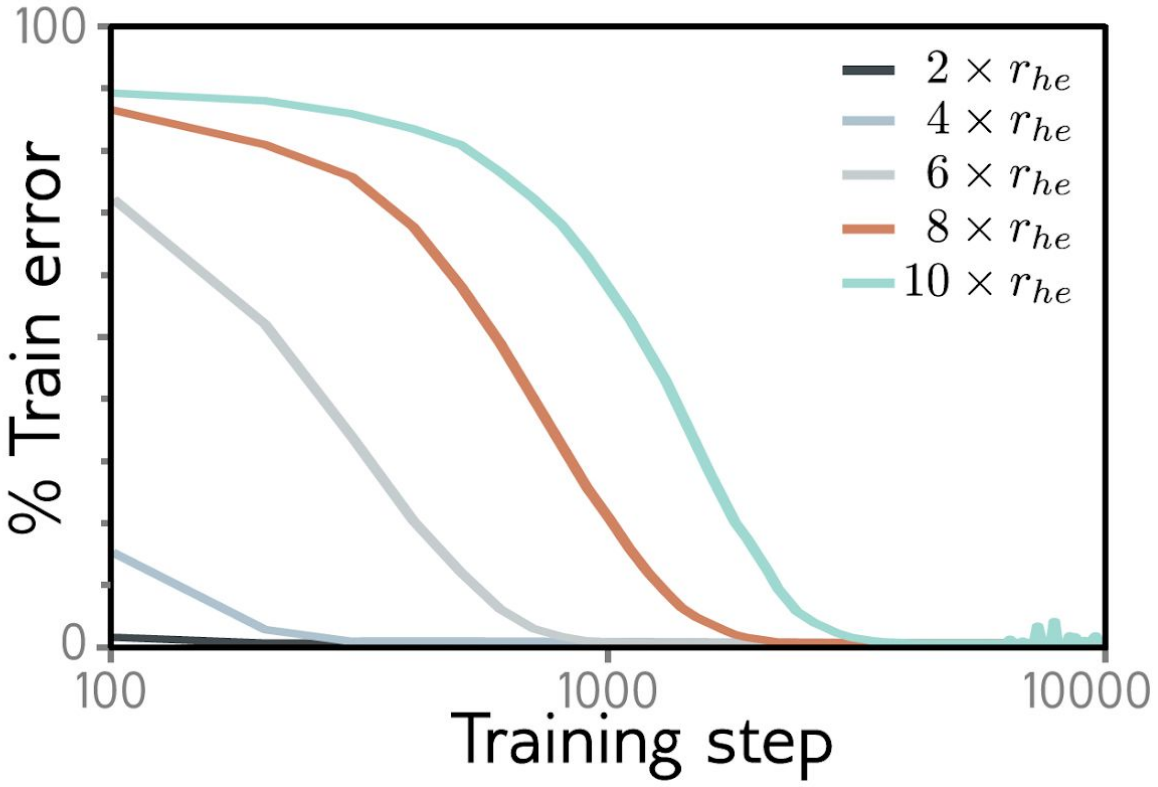


Figure 20.3 Overparameterization. ImageNet performance for convolutional nets as a function of overparameterization (in multiples of dataset size). Most models have 10–100 times more parameters than there were training examples. Models compared are ResNet (He et al., 2016a,b), DenseNet (Huang et al., 2017b), Xception (Chollet, 2017), EfficientNet (Tan & Le, 2019), Inception (Szegedy et al., 2017), ResNeXt (Xie et al., 2017), and AmoebaNet (Cubuk et al., 2019).

Figure 20.4 Initialization and fitting. A three-layer fully connected network with 200 hidden units per layer was trained on 1000 MNIST examples with AdamW using one-hot targets and mean-squared error loss. It takes longer to fit networks when larger multiples of He initialization are used, but this doesn't change the outcome. This may simply reflect the extra distance that the weights must move. Adapted from Liu et al. (2023c).



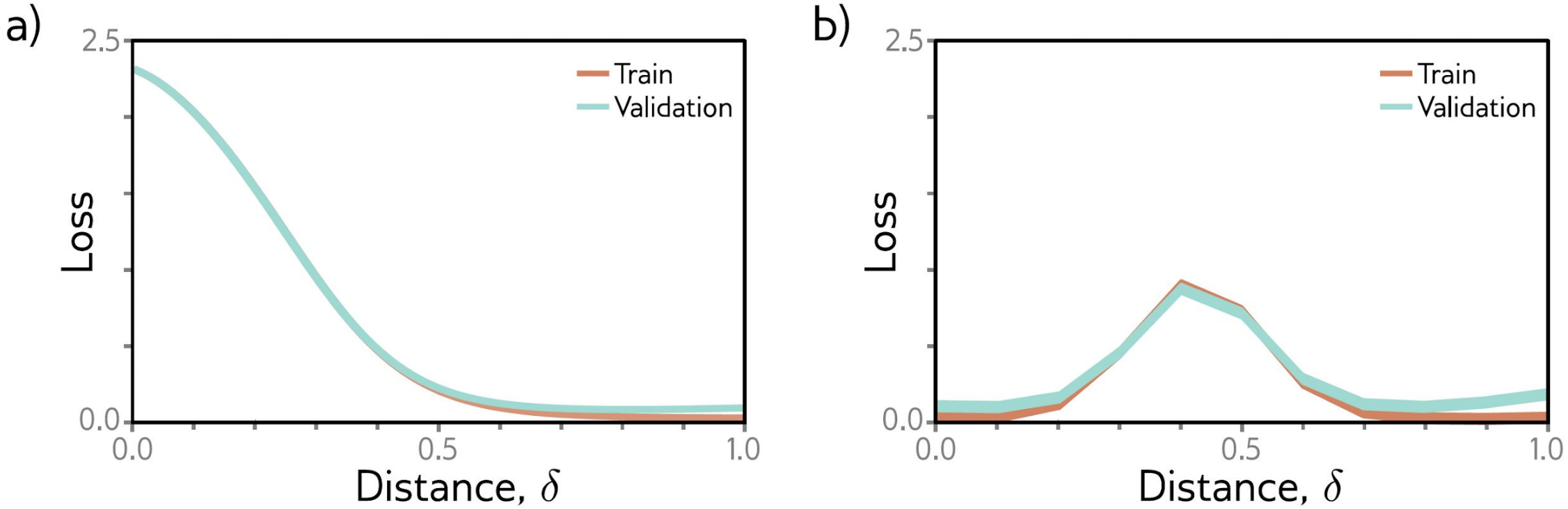
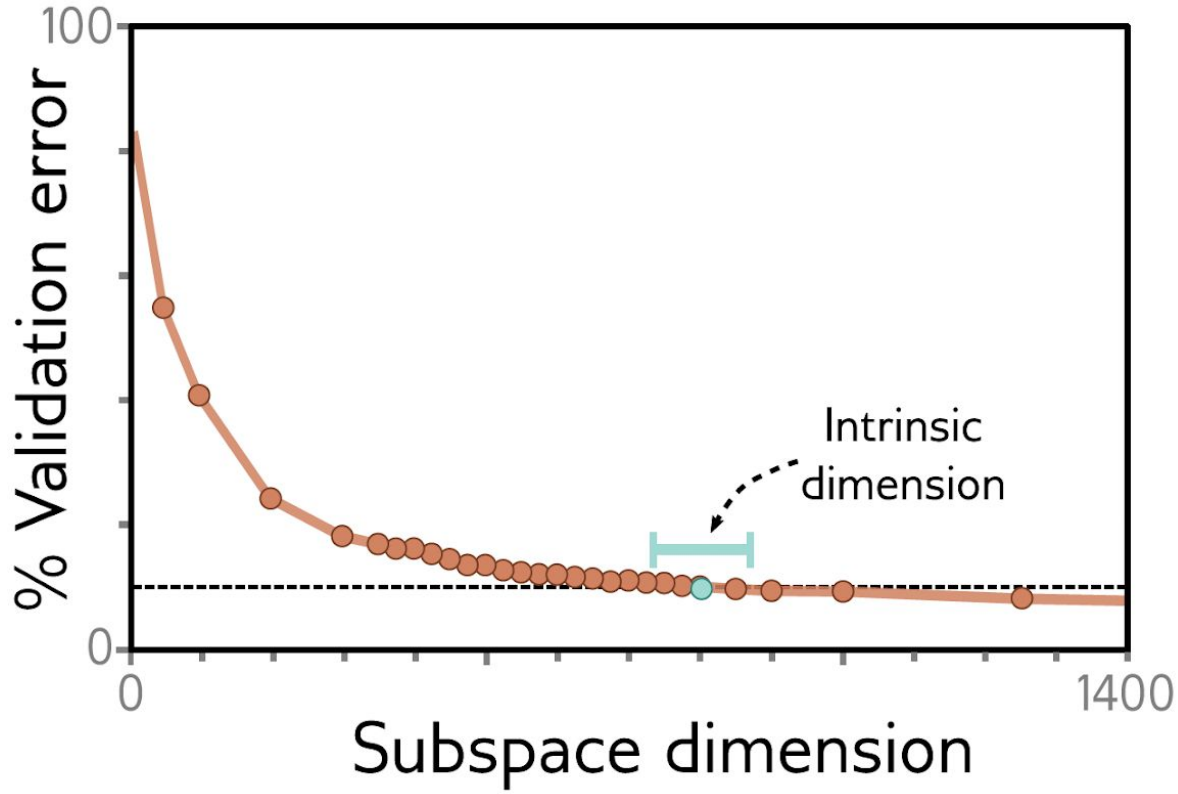


Figure 20.5 Linear slices through loss function. a) A two-layer fully connected ReLU network is trained on MNIST. The loss along a straight line starting at the initial parameters (δ=0) and finishing at the trained parameters (δ=1) descends monotonically. b) However, in this two-layer fully connected MaxOut network on MNIST, there is an increase in the loss along a straight line between one solution (δ=0) and another (δ=1). Adapted from Goodfellow et al. (2015b).

Figure 20.6 Subspace training. A fully connected network with two hidden layers, each with 200 units was trained on MNIST. Parameters were initialized using a standard method but then constrained to lie within a random subspace. Performance reaches 90% of the unconstrained level when this subspace is 750D (termed the *intrinsic dimension*), which is 0.4% of the original parameters. Adapted from Li et al. (2018a).



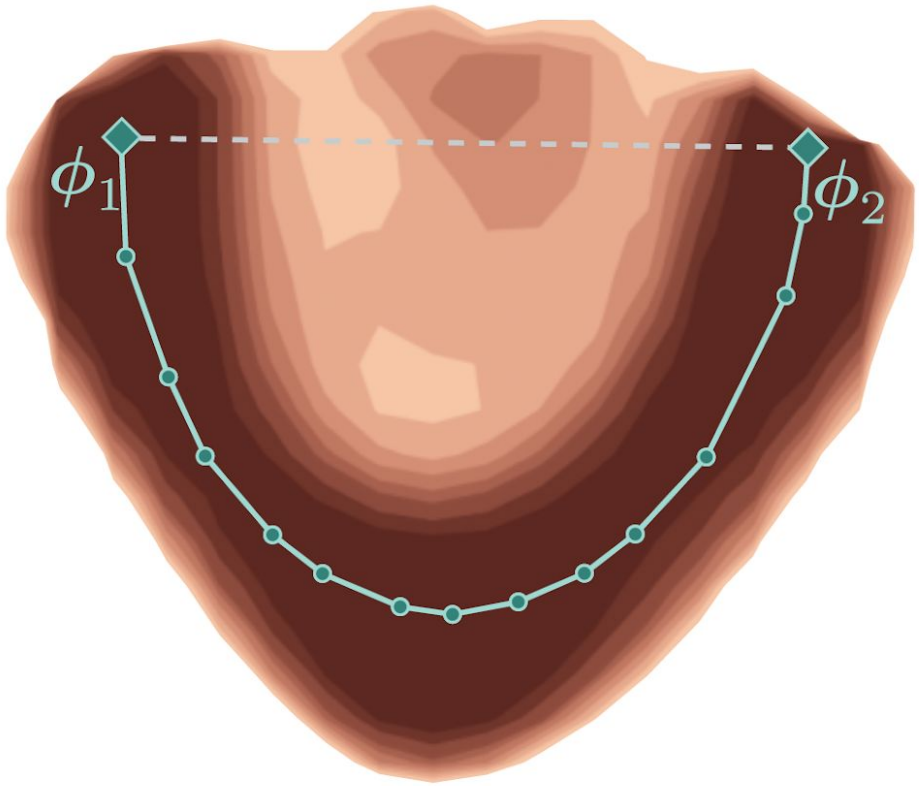


Figure 20.7 Connections between minima. A slice through the loss function of DenseNet on CIFAR-10. Parameters ϕ_1 and ϕ_2 are two independently discovered minima. Linear interpolation between these parameters reveals an energy barrier (dashed line). However, for sufficiently deep and wide networks, it is possible to find a curved path of low energy between two minima (cyan line). Adapted from Draxler et al. (2018).

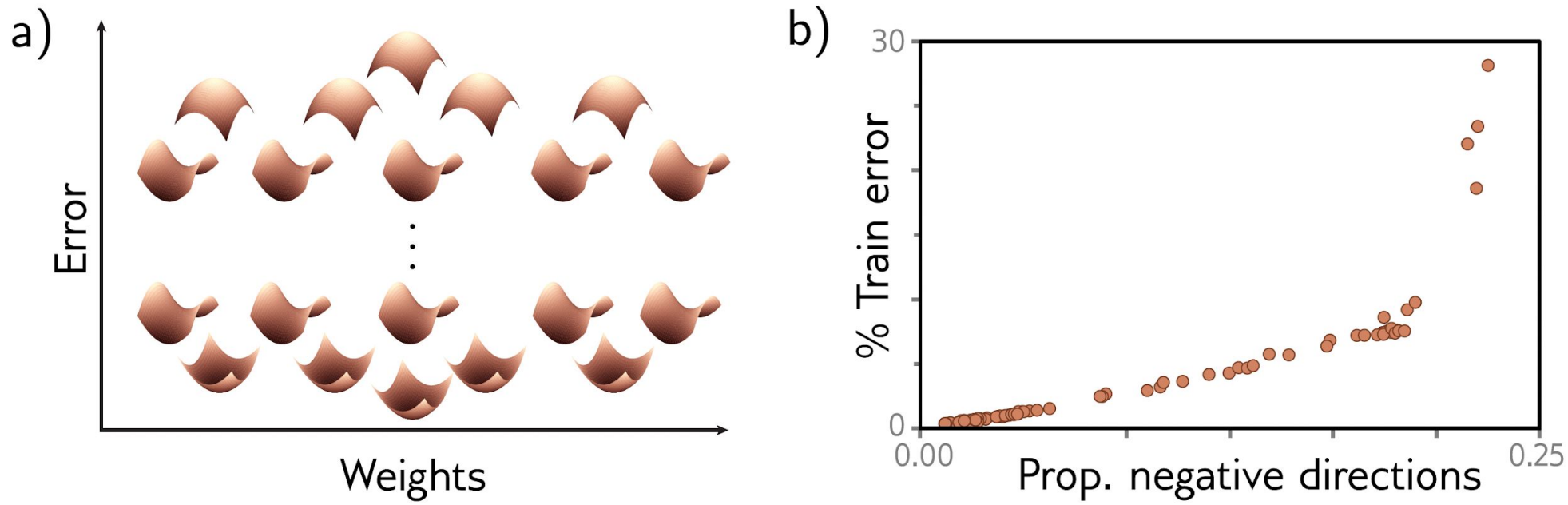


Figure 20.8 Critical points vs. loss. a) In random Gaussian functions, the number of directions in which the function curves down at points with zero gradient decreases with the height of the function, so minima all appear at lower function values. b) Dauphin et al. (2014) found critical points on a neural network loss surface (i.e., points with zero gradient). They showed that the proportion of negative eigenvalues (directions that point down) decreases with the loss. The implication is that all minima (points with zero gradient where no directions point down) have low losses. Adapted from Dauphin et al. (2014) and Bahri et al. (2020).

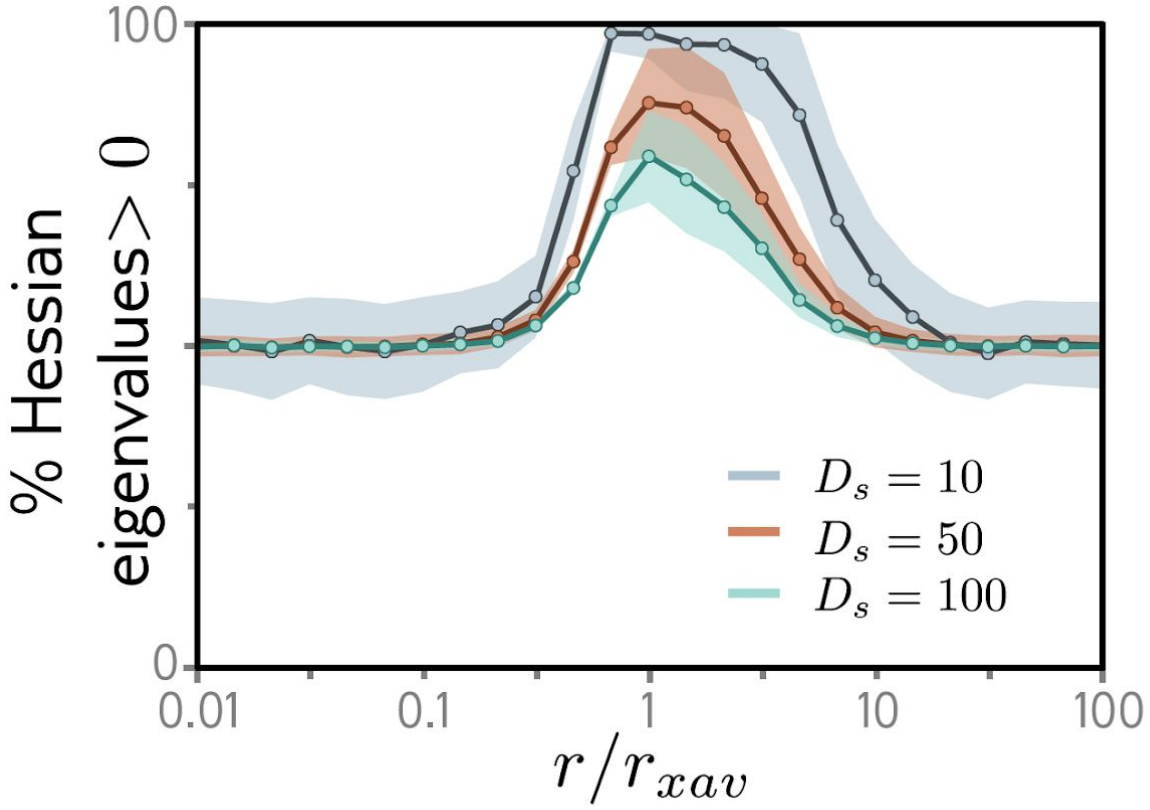
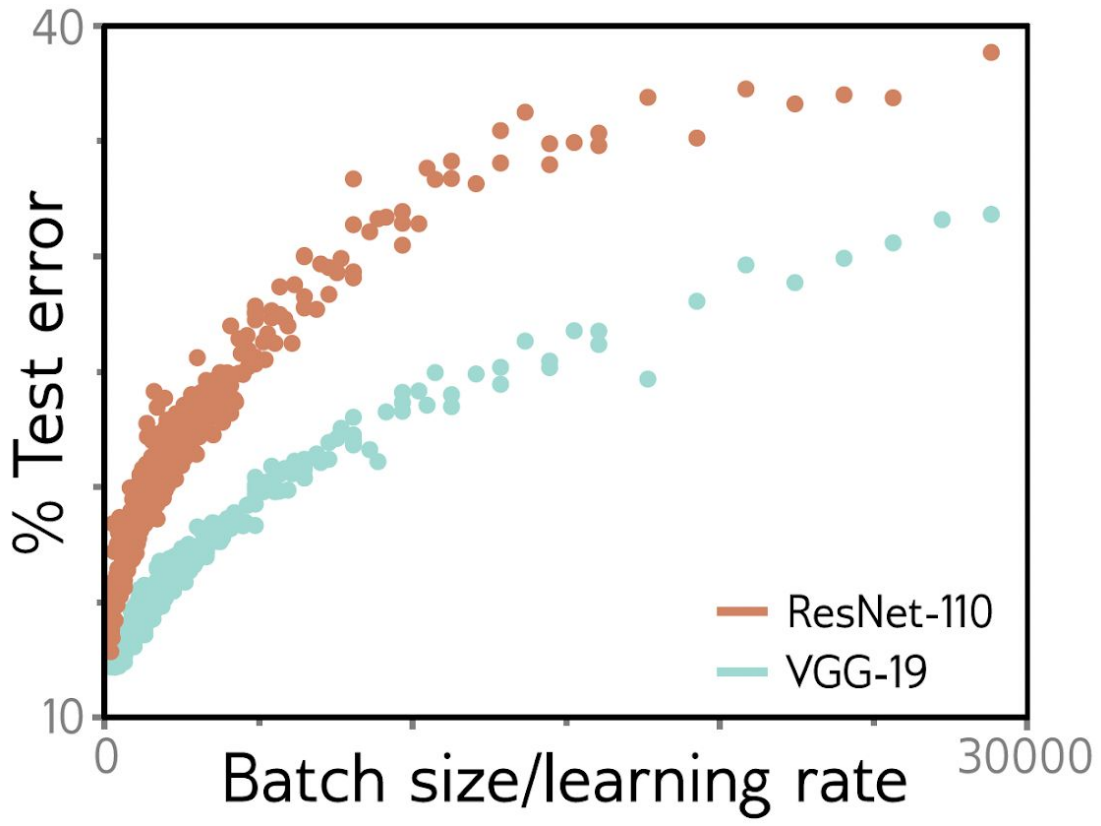


Figure 20.9 Goldilocks zone. The proportion of eigenvalues of the Hessian that are greater than zero (a measure of positive curvature/convexity) within a random subspace of dimension D_s in a two-layer fully connected network with ReLU functions applied to MNIST as a function of the squared radius r^2 of the point relative to Xavier initialization. There is a pronounced region of positive curvature known as the *Goldilocks zone*. Adapted from Fort & Scherlis (2019).

Figure 20.10 Batch size to learning rate ratio. Generalization of two models on the CIFAR-10 database depends on the ratio of batch size to the learning rate. As the batch size increases, generalization decreases. As the learning rate increases, generalization increases. Adapted from He et al. (2019).



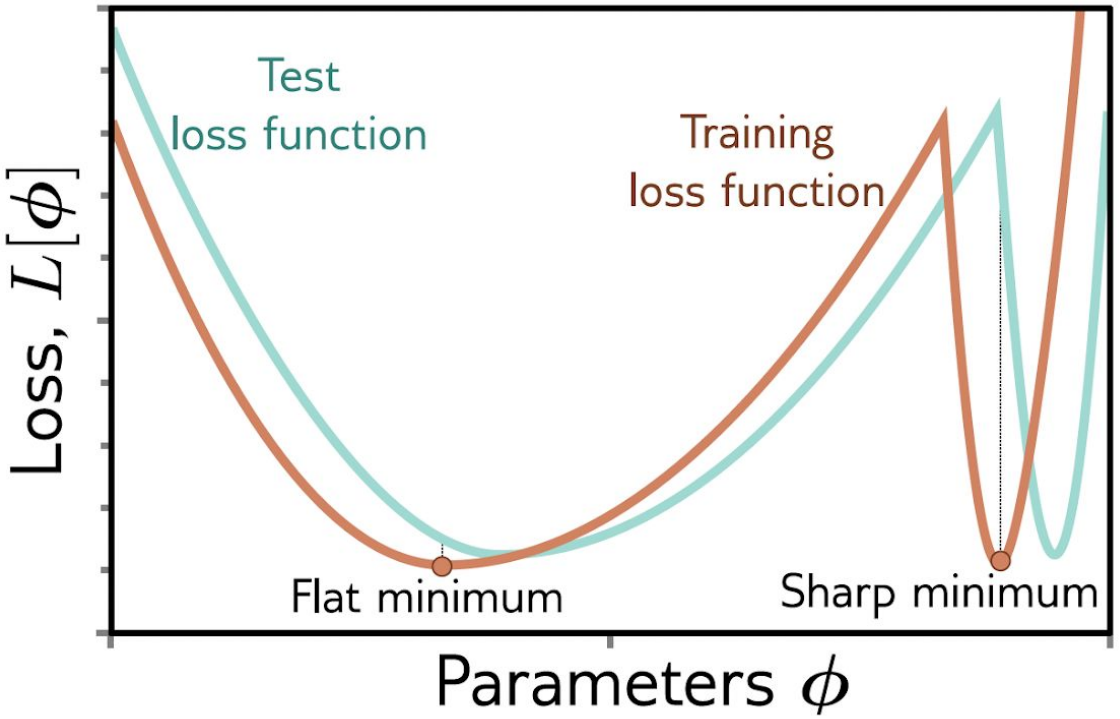


Figure 20.11 Flat vs. sharp minima. Flat minima are expected to generalize better. Small errors in estimating the parameters or in the alignment of the train and test loss functions are less problematic in flat regions. Adapted from Keskar et al. (2017).

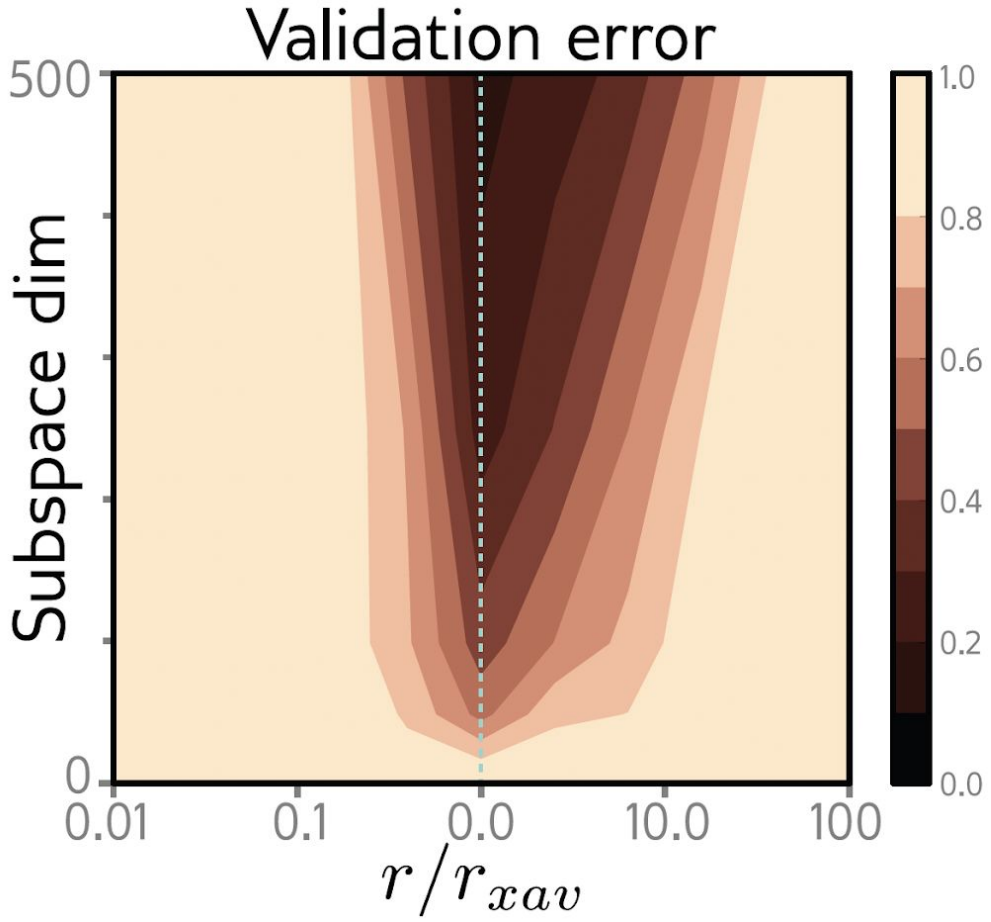


Figure 20.12 Generalization on hyperspheres. A fully connected network with two hidden layers, each with 200 units (198,450 parameters) was trained on the MNIST database. The parameters are initialized to a given ℓ_2 norm and then constrained to maintain this norm and to lie in a subspace (vertical direction). The network generalizes well in a small range around the radius r defined by Xavier initialization (cyan dotted line). Adapted from Fort & Scherlis (2019).

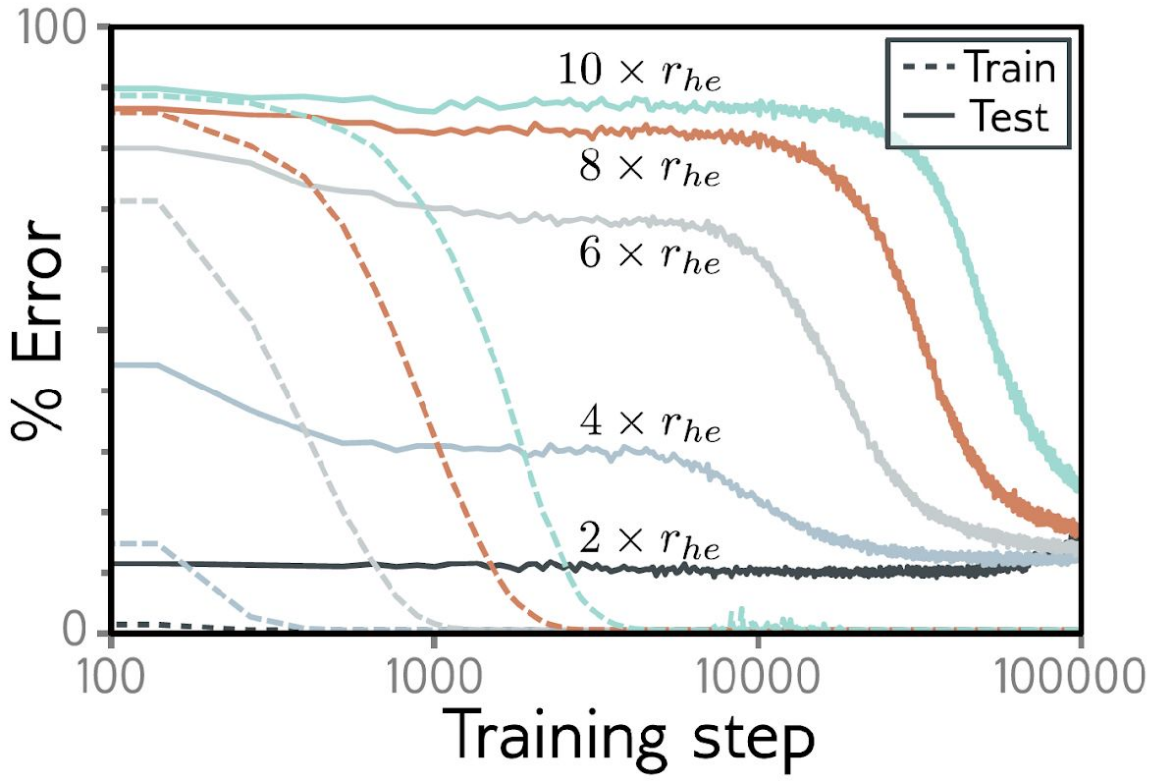
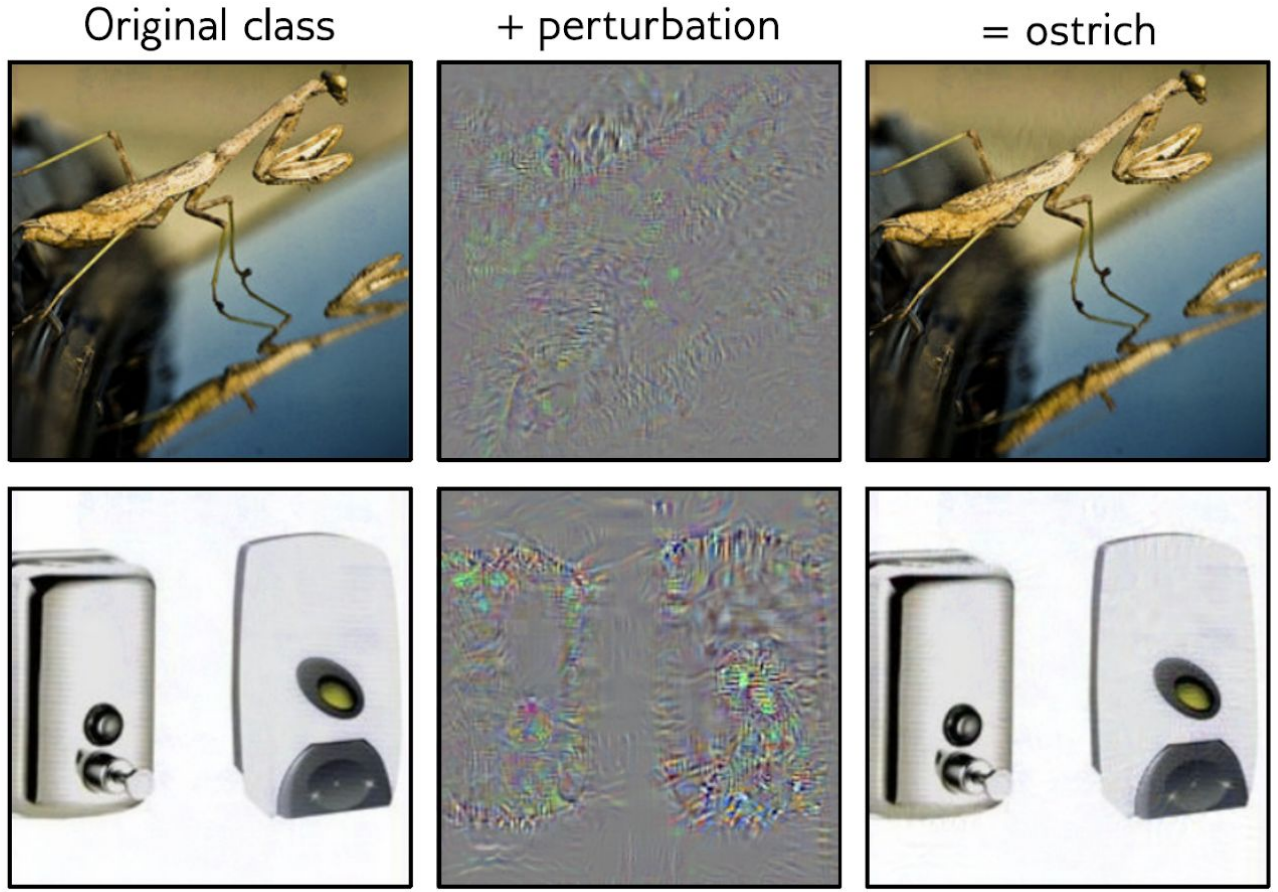


Figure 20.13 Grokking. When the parameters are initialized so that their ℓ_2 norm (radius) of the parameters is considerably larger than is specified by He initialization, training takes longer (dashed lines), and generalization takes *much* longer (solid lines). The lag in generalization is attributed to the time taken for the norm of the weights to decrease back to the Goldilocks zone. Adapted from Liu et al. (2023c).

Figure 20.14 Adversarial examples. In each case, the left image is correctly classified by AlexNet. By considering the gradients of the network output with respect to the input, it's possible to find a small perturbation (center, magnified by 10 for visibility) that, when added to the original image (right), causes the network to misclassify it as an ostrich. This is despite the fact that the original and perturbed images are almost indistinguishable to humans. Adapted from Szegedy et al. (2014).



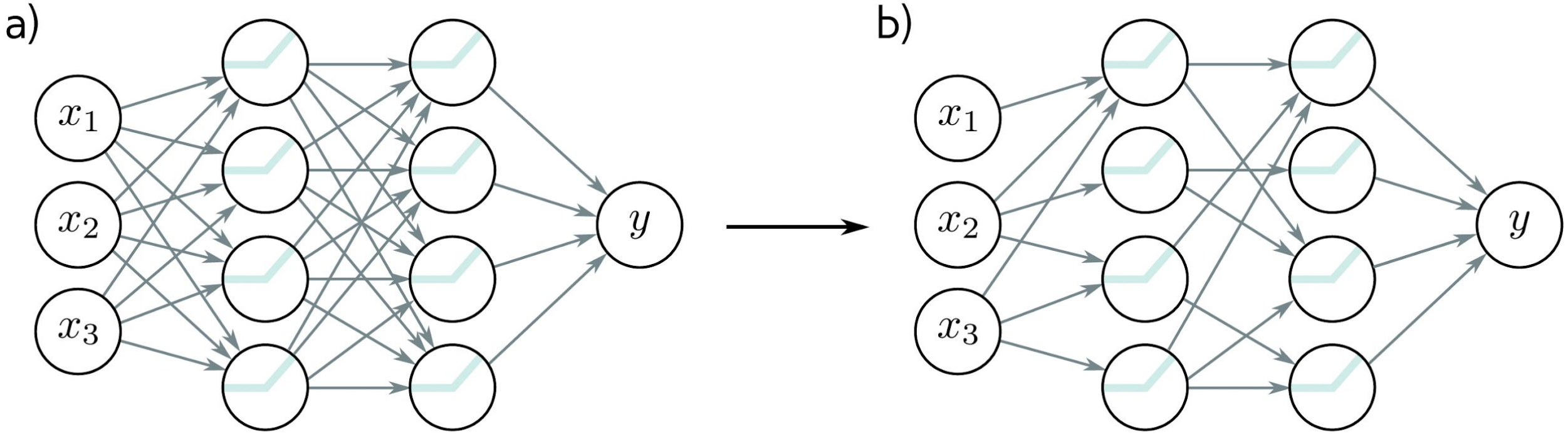


Figure 20.15 Pruning neural networks. The goal is to remove as many weights as possible without decreasing performance. This is often done just based on the magnitude of the weights. Typically, the network is fine-tuned after pruning. a) Example fully connected network. b) After pruning.

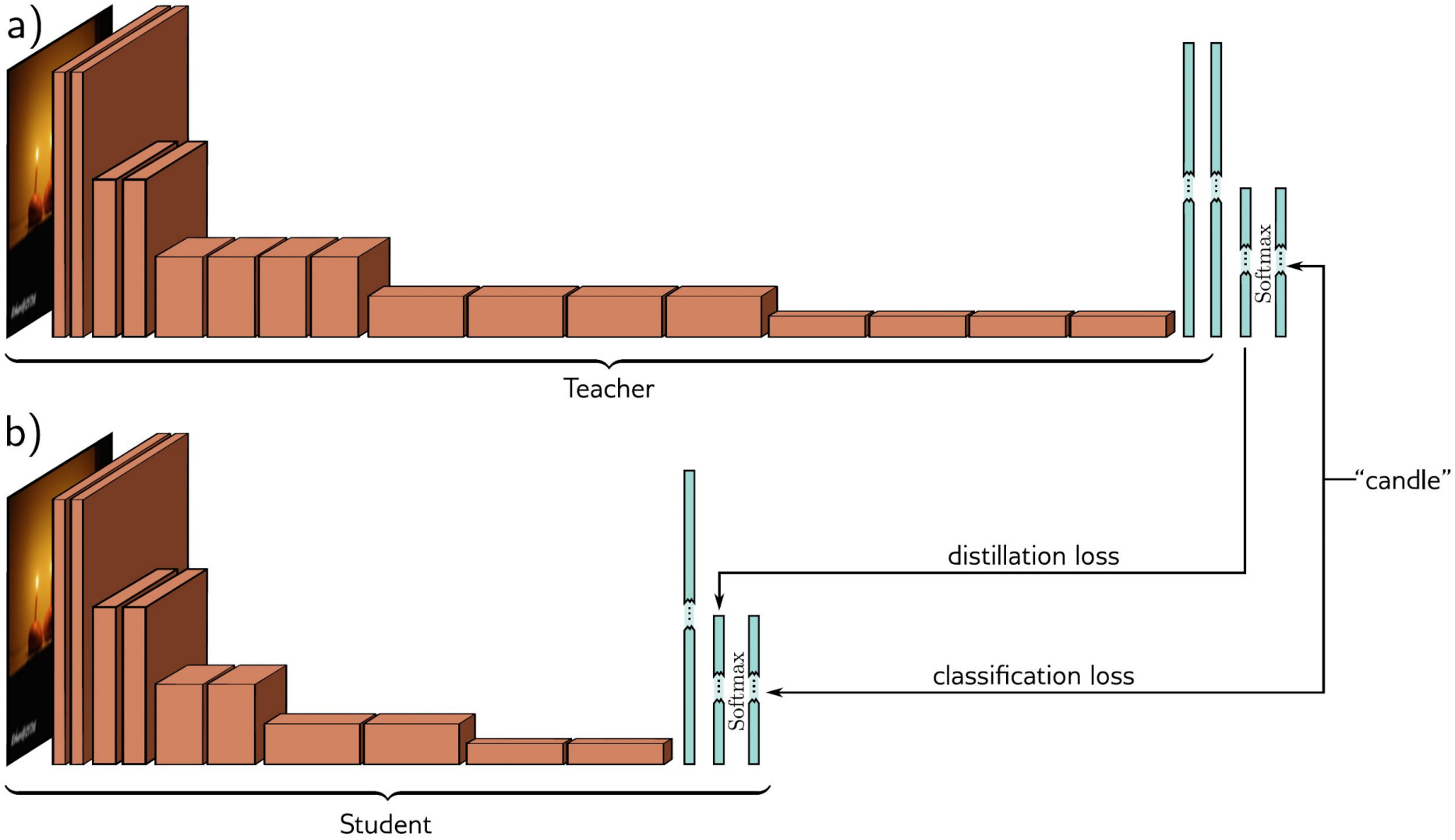


Figure 20.16 Knowledge distillation. a) A teacher network for image classification is trained as usual, using a multiclass cross-entropy classification loss. b) A smaller student network is trained with the same loss, plus also a distillation loss that encourages the pre-softmax activations to be the same as for the teacher.