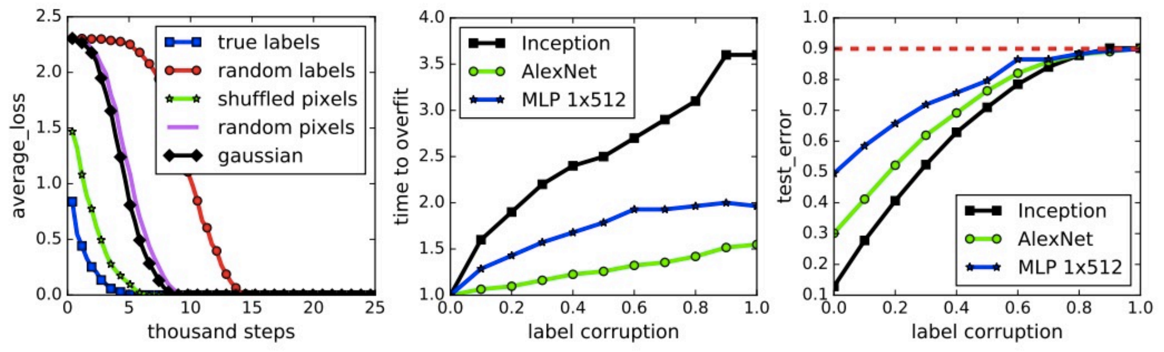
Understanding Deep Learning Requires Rethinking Generalization

Contribution

Traditional view of generalization is incapable of distinguishing between different neural networks that have radically different generalization performance.

- Randomization tests
    - **Deep neural networks easily fit random labels.**
- The role of explicit regularization
    - **Explicit regularization may improve generalization performance, but is neither necessary nor by itself sufficient for controlling generalization error.**
- Finite sample expressivity
    - **Two-layer ReLU network with p=2n+d parameters that can express any labeling of any sample of size n in d dimensions.**
- The role of implicit regularization
    - **For linear models, SGD always converges to a solution on small norm. Hence, the algorithm itself is implicitly regularizing the solution**

EFFECTIVE CAPACITY OF NEURAL NETWORKS



(a) learning curves (b) convergence slowdown (c) generalization error growth

Figure 1: Fitting random labels and random pixels on CIFAR10. (a) shows the training loss of various experiment settings decaying with the training steps. (b) shows the relative convergence time with different label corruption ratio. (c) shows the test error (also the generalization error since training error is 0) under different label corruptions.

Interesting Observations:

1. The effective capacity of neural networks is large enough for a brute-force memorization of the entire data set.
2. Even optimization on random labels remains easy. In fact, training time increases only by a small constant factor compared with training on the true labels.
3. Randomizing labels is solely a data transformation, leaving all other properties of the learning problem unchanged.
4. A steady deterioration of the generalization error as we increase the noise level. This shows that neural networks are able to capture the remaining signal in the data, while at the same time fit the noisy part using brute-force.

Implication

Preliminary

PAC learning framework

$\mathcal{X}$: input space, the set of all possible example or instances.

$\mathcal{Y}$: output space, the set of all possible labels or target values.

Concept c : $\mathcal{X} \rightarrow \mathcal{Y}$, a mapping from $\mathcal{X}$ to $\mathcal{Y}$.

Concept Class C : a set of cencept

Hypothesis set H : a set of h.

Learning process : Received a sample $S = (x_1, \ldots, x_m)$ drawn i.i.d according to D as well as the labels $(c(x_1), \ldots, c(x_m))$ which are based on specific target concept $c \in C$, the learner aims to select a $h_s \in H$ that has a small generalization error with respect to the concept c.

Generalization error : $R(h) = \underset{x \sim D}{Pr}[h(x) \neq c(x)] = \underset{x \sim D}{E}[1_{h(x) \neq c(x)}]$,

where 1_ω is the indicator function of the event ω.

Empirical error : $\hat{R}(h) = \frac{1}{m} \sum_{i=1}^{m} [1_{h(x) \neq c(x)}]$

function g : maps $(x, y) \in \mathcal{X} \times \mathcal{Y}$ to $L(h(x), y)$

G : family of loss functions associated to H

Rademacher complexity

$\hat{\mathfrak{R}}(\mathcal{H}) = \mathbb{E}_\sigma[\underset{h \in \mathcal{H}}{sup} \frac{1}{n} \sum_{i=1}^{n} \sigma_i h(x_i)]$

where $\sigma_1, \ldots \sigma_n \in \pm 1$ are i.i.d uniform random variables.

its idea : measures on average $\mathcal{H}$'s capability of fitting uniform noise to evaluate the complexity of $\mathcal{H}$.

The experiment results above suggests the neural networks fit the training set with random labels perfectly, we get $\hat{\mathfrak{R}}(\mathcal{H}) = 1$.

Uniform stability

Uniform stability of an algorithm A measures how sensitive the algorithm is to the replacement of a single example. However, it is solely a property of the algorithm, which does not take into account specifics of the data or the distribution of the labels.

THE ROLE OF REGULARIZATION

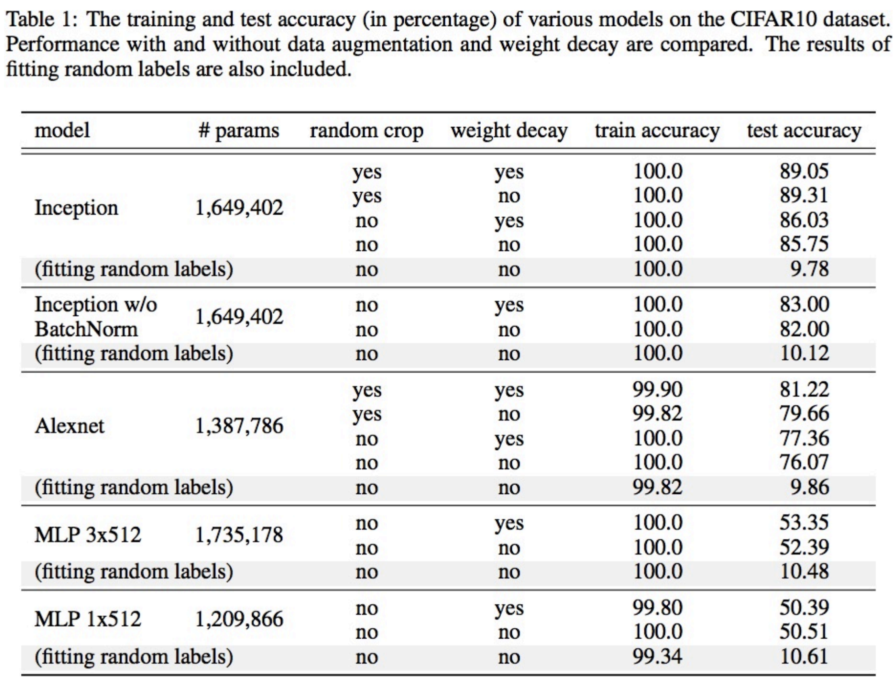


Table 1: The training and test accuracy (in percentage) of various models on the CIFAR10 dataset. Performance with and without data augmentation and weight decay are compared. The results of fitting random labels are also included.

model	# params	random crop	weight decay	train accuracy	test accuracy
Inception	1,649,402	yes	yes	100.0	89.05
		yes	no	100.0	89.31
		no	yes	100.0	86.03
		no	no	100.0	85.75
(fitting random labels)		no	no	100.0	9.78
Inception w/o BatchNorm	1,649,402	no	yes	100.0	83.00
		no	no	100.0	82.00
(fitting random labels)		no	no	100.0	10.12
Alexnet	1,387,786	yes	yes	99.90	81.22
		yes	no	99.82	79.66
		no	yes	100.0	77.36
		no	no	100.0	76.07
(fitting random labels)		no	no	99.82	9.86
MLP 3x512	1,735,178	no	yes	100.0	53.35
		no	no	100.0	52.39
(fitting random labels)		no	no	100.0	10.48
MLP 1x512	1,209,866	no	yes	99.80	50.39
		no	no	100.0	50.51
(fitting random labels)		no	no	99.34	10.61

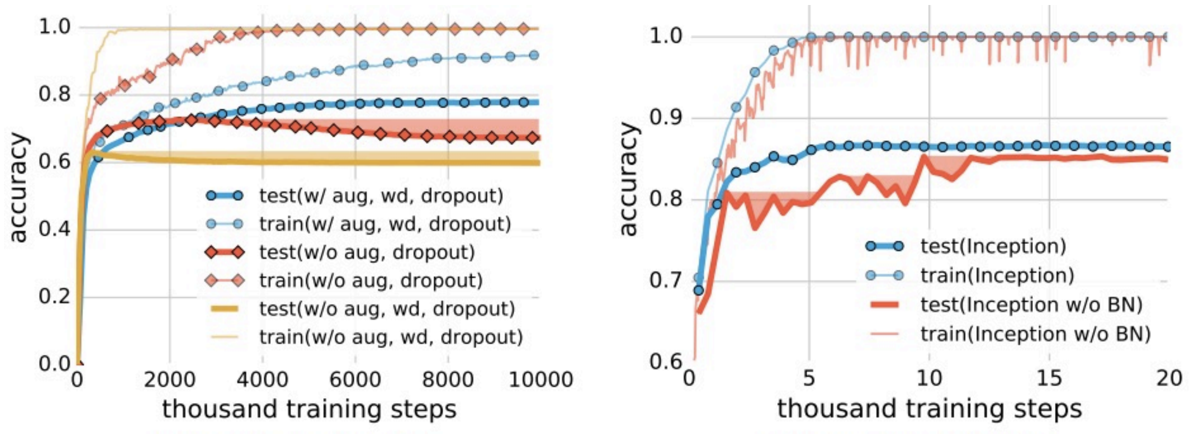
Observation : Both regularization techniques help to improve the generalization performance, but even with all of the regularizers turned off, all of the models still generalize very well.

Table 2: The top-1 and top-5 accuracy (in percentage) of the Inception v3 model on the ImageNet dataset. We compare the training and test accuracy with various regularization turned on and off, for both true labels and random labels. The original reported top-5 accuracy of the Alexnet on ILSVRC 2012 is also listed for reference. The numbers in parentheses are the best test accuracy during training, as a reference for potential performance gain of early stopping.

data aug	dropout	weight decay	top-1 train	top-5 train	top-1 test	top-5 test
ImageNet 1000 classes with the original labels						
yes	yes	yes	92.18	99.21	77.84	93.92
yes	no	no	92.33	99.17	72.95	90.43
no	no	yes	90.60	100.0	67.18 (72.57)	86.44 (91.31)
no	no	no	99.53	100.0	59.80 (63.16)	80.38 (84.49)
Alexnet (Krizhevsky et al., 2012)			-	-	-	83.6
ImageNet 1000 classes with random labels						
no	yes	yes	91.18	97.95	0.09	0.49
no	no	yes	87.81	96.15	0.12	0.50
no	no	no	95.20	99.14	0.11	0.56

Table 2 shows the performance on Imagenet with true labels and random labels, respectively.

Observation : So while regularization is important, bigger gains can be achieved by simply changing the model architecture. It is difficult to say that the regularizers count as a fundamental phase change in the generalization capability of deep nets.



(a) Inception on ImageNet (b) Inception on CIFAR10

Figure 2: Effects of implicit regularizers on generalization performance. aug is data augmentation, wd is weight decay, BN is batch normalization. The shaded areas are the cumulative best test accuracy, as an indicator of potential performance gain of early stopping. (a) early stopping could potentially improve generalization when other regularizers are absent. (b) early stopping is not necessarily helpful on CIFAR10, but batch normalization stablize the training process and improves generalization.

Observation : The shaded area indicate the accumulative best test accuracy, as a reference of potential performance gain for early stopping. However, on the CIFAR10 dataset, we do not observe any potential benefit of early stopping.

Summary : Our observations on both explicit and implicit regularizers are consistently suggesting that regularizers, when properly tuned, could help to improve the generalization performance. However, it is unlikely that the regularizers are the fundamental reason for generalization, as the networks continue to perform well after all the regularizers removed.

FINITE-SAMPLE EXPRESSIVITY

Theorem 1. There exists a two-layer neural network with ReLU activations and 2n+d weights that can represent any function on a sample of size n in d dimensions.

IMPLICIT REGULARIZATION: AN APPEAL TO LINEAR MODELS

n distinct data points $\{(x_i, y_i)\}$ where x_i are d-dimensional feature vectors and y_i are labels. Letting loss denote a nonnegative loss function with loss(y, y) = 0, consider the empirical risk minimization (ERM) problem

$min_{\omega \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} loss(\omega^T x_i, y_i)$

if $d \geq n$, Xw = y has an infinite number of solutions. One popular way to understand quality of minima is the curvature of the loss function at the solution.

For linear model, the Hessian is degenerate at all global optimal solutions.

A promising direct is to consider the workhorse algorithm SGD.

$w_{t+1} = w_t - \eta_t e_t x_{i_t}$

η_t : step size

e_t : prediction error loss

$w = \sum_{i=1}^{n} \alpha_i x_i$

$w = X^T \alpha$

$XX^T \alpha = y$

kernel solution is equivalent to the **minimum l2-norm** solution of Xw = y

problems : this notion of minimum norm is not predictive of generalization performance. Preprocessing can obtain better performance but may get larger norm.

CONCLUSION

The classical view of machine learning rests on the idea of parsimony. In fact, sheer memorization is possible to be effective for natural tasks.

Understanding neural networks requires rethinking generalization.