Understanding Deep Learning (Still) Requires Rethinking Generalization

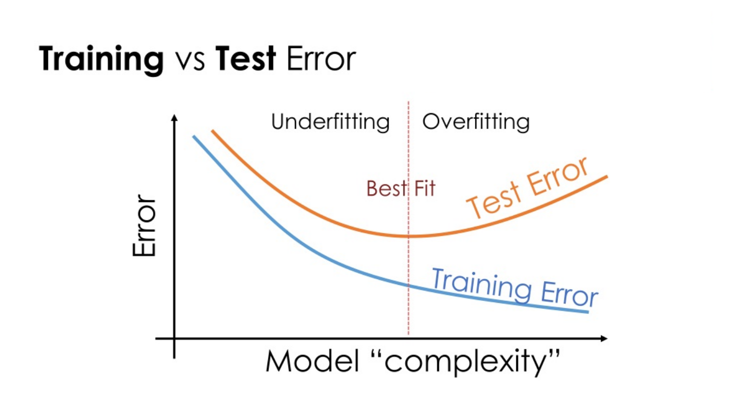
Alban MARIE [†]

March 10, 2022

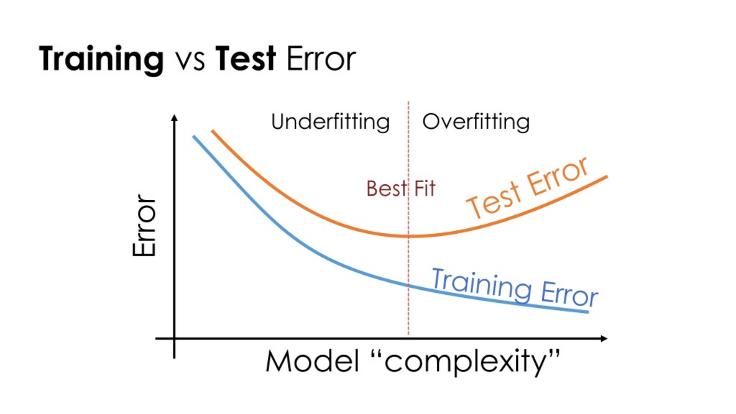
[†] Univ Rennes, INSA Rennes, CNRS, IETR - UMR6164, France

Why something is wrong with deep learning

Bias vs Variance



Bias vs Variance



➜ Where is deep learning on the x-axis?

Underfitting vs Overfitting

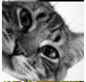
Where is deep learning on the x-axis?

Authors perform a simple experimental framework to propose an answer to this question.

Experimental framework

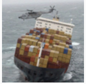
➜ True labels

Truck



Cat

Bird house



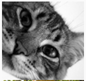
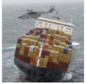
Container ship

Russian airplane probably

Dog

Experimental framework

➜ Random labels

	Truck	Russian airplane probably
	Cat	Cat
	Bird house	Container ship
	Container ship	Truck
	Russian airplane probably	Truck
	Dog	Bird house

Experimental framework

➜ Random labels

Russian airplane probably

Cat

Container ship

Truck

Truck

Bird house

Experimental framework

➜ True labels

➜ Random labels (previous slide)

Experimental framework

- ➜ True labels
- ➜ Random labels (previous slide)
- ➜ Partially corrupted labels
  - ➜ Independently for each image and with a probability p, draw a random label for this image

Experimental framework

- True labels
- Random labels (previous slide)
- Partially corrupted labels
  - Independently for each image and with a probability p, draw a random label for this image
- Shuffled pixels
  - Select one random pixel permutation. Apply this permutation to all images.

Experimental framework

- ➜ True labels
- ➜ Random labels (previous slide)
- ➜ Partially corrupted labels
  - ➜ Independently for each image and with a probability p, draw a random label for this image
- ➜ Shuffled pixels
  - ➜ Select one random pixel permutation. Apply this permutation to all images.
- ➜ Random pixels
  - ➜ Independently for each image, apply a random permutation.

EXPERIMENTAL FRAMEWORK

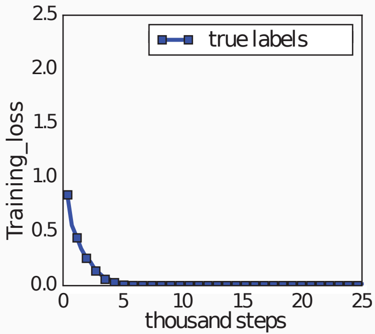
- ➜ True labels
- ➜ Random labels (previous slide)
- ➜ Partially corrupted labels
  - ➜ Independently for each image and with a probability p, draw a random label for this image
- ➜ Shuffled pixels
  - ➜ Select one random pixel permutation. Apply this permutation to all images.
- ➜ Random pixels
  - ➜ Independently for each image, apply a random permutation.
- ➜ Gaussian pixels
  - ➜ Independently for each pixel, draw a random value from gaussian distribution with mean and std from original dataset

Experimental framework

Ok cool but ...

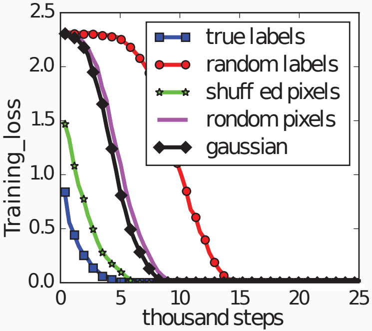
... what's the point of these experiments?

Results



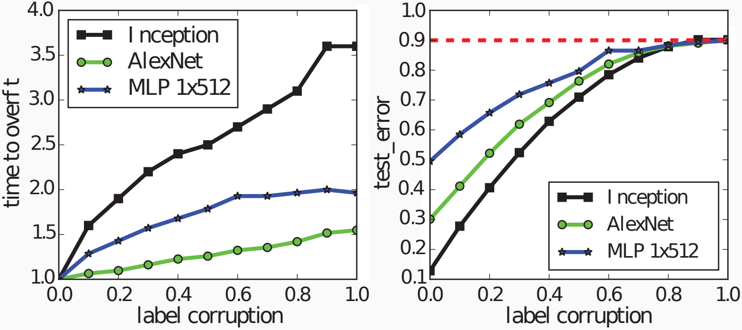
Training loss of true label experiment decaying with the training steps on CIFAR10

Results



Training loss of various experiment settings decaying with the training steps on CIFAR10

Results



(left) Relative convergence time with different label corruption ratio and **(right)** test error (also the generalization error since training error is 0) under different label corruptions.

How bad is it, doctor?

➜ Training loss always converges to 0!

How bad is it, doctor?

- ➜ Training loss always converges to 0!
  - ➜ The effective capacity of Neural Network (NN) is sufficient for memorizing the entire data set.

How bad is it, doctor?

- Training loss always converges to 0!
  - The effective capacity of Neural Network (NN) is sufficient for memorizing the entire data set.
  - NN are able to capture the remaining signal in the data if any, while at the same time fit the noisy part using brute-force.

How bad is it, doctor?

- ➜ Training loss always converges to 0!
  - ➜ The effective capacity of Neural Network (NN) is sufficient for memorizing the entire data set.
  - ➜ NN are able to capture the remaining signal in the data if any, while at the same time fit the noisy part using brute-force.
- ➜ Optimization remains easy, whatever you aim to fit.

How bad is it, doctor?

- ➜ Training loss always converges to 0!
  - ➜ The effective capacity of Neural Network (NN) is sufficient for memorizing the entire data set.
  - ➜ NN are able to capture the remaining signal in the data if any, while at the same time fit the noisy part using brute-force.
- ➜ Optimization remains easy, whatever you aim to fit.
  - ➜ Easy even with random labels (the randomization breaks any relationship between the image and the label).

Bias vs Variance

➜ The effective capacity of NN is sufficient for memorizing the entire data set.

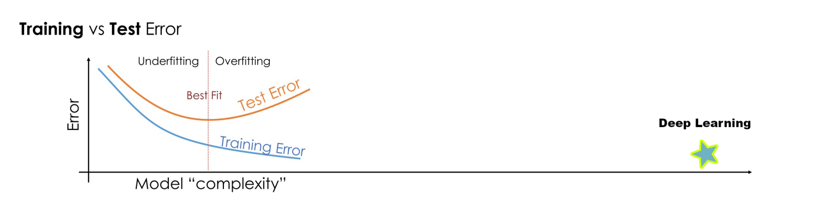
Bias vs Variance

- ➜ The effective capacity of NN is sufficient for memorizing the entire data set.
  - ➜ Thus, very high overfitting

Bias vs Variance

- ➜ The effective capacity of NN is sufficient for memorizing the entire data set.
  - ➜ Thus, very high overfitting
- ➜ At the same time, increasing model complexity allow to reduce test error (and thus generalization error since training error is 0)

Bias vs Variance



- The effective capacity of NN is sufficient for memorizing the entire data set.
  - Thus, very high overfitting
- At the same time, increasing model complexity allow to reduce test error (and thus generalization error since training error is 0)
  - Something is wrong

What about regularization?

Regularizers

Popular Belief Neural Network (NN) converges thanks to implicit and explicit regularizers

Regularizers

Popular Belief Neural Network (NN) converges thanks to implicit and explicit regularizers

Explicit regularizers are:

REGULARIZERS

Popular Belief Neural Network (NN) converges thanks to implicit and explicit regularizers

Explicit regularizers are:

➜ Data augmentation

Regularizers

Popular Belief Neural Network (NN) converges thanks to implicit and explicit regularizers

Explicit regularizers are:

- ➜ Data augmentation
- ➜ Weight decay (L_2 norm penalty on weights when too big)

REGULARIZERS

Popular Belief Neural Network (NN) converges thanks to implicit and explicit regularizers

Explicit regularizers are:

- ➜ Data augmentation
- ➜ Weight decay (L_2 norm penalty on weights when too big)
- ➜ Dropout

Regularizers

Popular Belief Neural Network (NN) converges thanks to implicit and explicit regularizers

Explicit regularizers are:

- ➜ Data augmentation
- ➜ Weight decay (L_2 norm penalty on weights when too big)
- ➜ Dropout

Implicit regularizers are:

Regularizers

Popular Belief Neural Network (NN) converges thanks to implicit and explicit regularizers

Explicit regularizers are:

- ➜ Data augmentation
- ➜ Weight decay (L_2 norm penalty on weights when too big)
- ➜ Dropout

Implicit regularizers are:

- ➜ Early stopping (stop training when generalization error is minimal)

REGULARIZERS

Popular Belief Neural Network (NN) converges thanks to implicit and explicit regularizers

Explicit regularizers are:

- ➜ Data augmentation
- ➜ Weight decay (L_2 norm penalty on weights when too big)
- ➜ Dropout

Implicit regularizers are:

- ➜ Early stopping (stop training when generalization error is minimal)
- ➜ SGD

Regularizers

Popular Belief Neural Network (NN) converges thanks to implicit and explicit regularizers

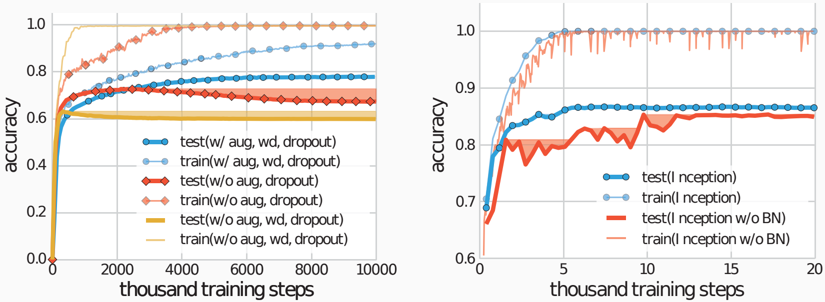
Explicit regularizers are:

- ➜ Data augmentation
- ➜ Weight decay (L_2 norm penalty on weights when too big)
- ➜ Dropout

Implicit regularizers are:

- ➜ Early stopping (stop training when generalization error is minimal)
- ➜ SGD
- ➜ NN architecture

Regularizers experiments



Regularizers impact on generalization for **(left)** Imagenet and **(right)** CIFAR10.
Data augmentation, **weight decay** and **batch normalization**
are referred as **aug**, **wd** and **BN**, respectively.

Regularizers conclusions

➜ NN training loss converge to 0, with or without regularizers

Regularizers conclusions

- ➜ NN training loss converge to 0, with or without regularizers
  - ➜ Training space with regularizers is still huge

Regularizers conclusions

- ➜ NN training loss converge to 0, with or without regularizers
  - ➜ Training space with regularizers is still huge
- ➜ Regularizers improve generalization but...

REGULARIZERS CONCLUSIONS

- NN training loss converge to 0, with or without regularizers
    - Training space with regularizers is still huge
- Regularizers improve generalization but...
    - it is not necessary for NN to converge

Regularizers conclusions

- ➜ NN training loss converge to 0, with or without regularizers
  - ➜ Training space with regularizers is still huge
- ➜ Regularizers improve generalization but...
  - ➜ it is not necessary for NN to converge
  - ➜ it is unlikely that regularizers are the fundamental reason for generalization

Regularizers conclusions

- ➜ NN training loss converge to 0, with or without regularizers
  - ➜ Training space with regularizers is still huge
- ➜ Regularizers improve generalization but...
  - ➜ it is not necessary for NN to converge
  - ➜ it is unlikely that regularizers are the fundamental reason for generalization
- ➜ Implicit Regularization with NN architecture is more powerful to reduce generalization error

Bridge the gap with theory

VC Dimension - Statistical learning theory

Let f be a classification model with weights θ that aims to predicts labels $y_{i,\ i \in \{1,\ldots N\}}$ based on input features $x_{i,\ i \in \{1,\ldots\ N\}}$.

VC Dimension - Statistical learning theory

Let f be a classification model with weights θ that aims to predicts labels $y_{i,\ i \in \{1,\dots N\}}$ based on input features $x_{i,\ i \in \{1,\dots\ N\}}$.

It it said that f **shatters** a dataset with $N \in \mathbb{N}$ elements if there exists a configuration for θ such that model f makes no errors while predicting y_i based on x_i for each $i \in \{1, \dots\ N\}$.

VC Dimension - Statistical learning theory

Let f be a classification model with weights θ that aims to predicts labels $y_{i,\ i \in \{1, \ldots N\}}$ based on input features $x_{i,\ i \in \{1, \ldots\ N\}}$.

It it said that f **shatters** a dataset with $N \in \mathbb{N}$ elements if there exists a configuration for θ such that model f makes no errors while predicting y_i based on x_i for each $i \in \{1, \ldots\ N\}$.

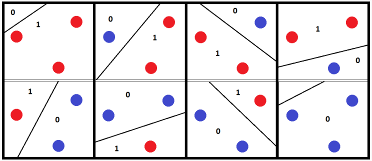
The **VC dimension** of a model f is greater than or equal to N if there exists at least one set of N points where f can shatter all arrangements.

VC Dimension - Statistical learning theory

Let f be a classification model with weights θ that aims to predicts labels $y_{i,\ i\in\{1,\ldots N\}}$ based on input features $x_{i,\ i\in\{1,\ldots\ N\}}$.

It it said that f **shatters** a dataset with $N \in \mathbb{N}$ elements if there exists a configuration for θ such that model f makes no errors while predicting y_i based on x_i for each $i \in \{1, \ldots\ N\}$.

The **VC dimension** of a model f is greater than or equal to N if there exists at least one set of N points where f can shatter all arrangements.



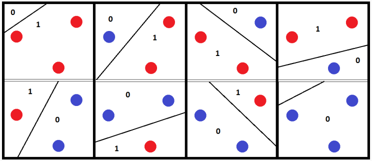
VC dim ≥ 3

VC Dimension - Statistical learning theory

Let f be a classification model with weights θ that aims to predicts labels $y_{i,\ i \in \{1,\ldots N\}}$ based on input features $x_{i,\ i \in \{1,\ldots\ N\}}$.

It it said that f **shatters** a dataset with $N \in \mathbb{N}$ elements if there exists a configuration for θ such that model f makes no errors while predicting y_i based on x_i for each $i \in \{1,\ldots\ N\}$.

The **VC dimension** of a model f is greater than or equal to N if there exists at least one set of N points where f can shatter all arrangements.



There is no set of 4 points that can be shattered by a line.

VC dim ≥ 3

VC dim < 4

VC Dimension - Statistical learning theory

Let N be the size of the dataset and D_{VC} the VC dimension of model f. With probability $1-\delta$:

$$err_{test} \leq err_{train} + \sqrt{\frac{1}{N}[D_{VC}(\log(\frac{2N}{D_{VC}})+1)-\log(\frac{\delta}{4})]}$$

VC Dimension - Statistical learning theory

Let N be the size of the dataset and D_{VC} the VC dimension of model f. With probability $1 - \delta$:

$$err_{test} \leq err_{train} + \sqrt{\frac{1}{N}[D_{VC}(\log(\frac{2N}{D_{VC}}) + 1) - \log(\frac{\delta}{4})]}$$

✘ For deep learning, $D_{VC} \gg N$ most of the time (complex solutions)

VC Dimension - Statistical learning theory

Let N be the size of the dataset and D_{VC} the VC dimension of model f. With probability $1-\delta$:

$$err_{test} \leq err_{train} + \sqrt{\frac{1}{N}[D_{VC}(\log(\frac{2N}{D_{VC}})+1) - \log(\frac{\delta}{4})]}$$

- ✘ For deep learning, $D_{VC} \gg N$ most of the time (complex solutions)
- ➜ The effective capacity of NN is sufficient for memorizing the entire data set.

VC Dimension - Statistical learning theory

Let N be the size of the dataset and D_{VC} the VC dimension of model f. With probability $1 - \delta$:

$$err_{test} \leq err_{train} + \sqrt{\frac{1}{N}[D_{VC}(\log(\frac{2N}{D_{VC}}) + 1) - \log(\frac{\delta}{4})]}$$

- ✘ For deep learning, $D_{VC} \gg N$ most of the time (complex solutions)
- ➜ The effective capacity of NN is sufficient for memorizing the entire data set.
  - ➜ Thus, $D_{VC} \geq N$

NN Finite Sample Expressivity

➜ Most work in the litterature try to characterize NN expressivity at the **population level**

NN Finite Sample Expressivity

- ➜ Most work in the litterature try to characterize NN expressivity at the **population level**
- ➜ **population level**: infinite sample size (dataset with infinite number of elements)

NN Finite Sample Expressivity

- ➜ Most work in the litterature try to characterize NN expressivity at the **population level**
- ➜ **population level**: infinite sample size (dataset with infinite number of elements)
- ➜ Instead, authors propose to express NN expressivity on a finite sample size N

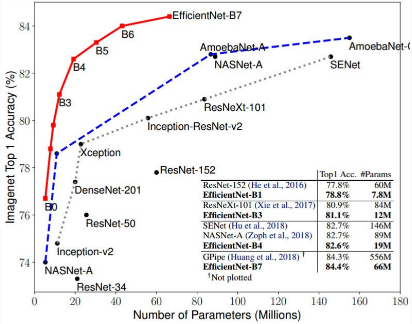
NN Finite Sample Expressivity

- ➜ Most work in the litterature try to characterize NN expressivity at the **population level**
- ➜ **population level**: infinite sample size (dataset with infinite number of elements)
- ➜ Instead, authors propose to express NN expressivity on a finite sample size N

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.*

NN Finite Sample Expressivity

Dataset	N	d	$2N + d$
MNIST	70.000	$28^2 = 784$	140.784
CIFAR10	50.000	$3 \times 32^2 = 3.072$	103.072
ImageNet	$1.281.165$	$3 \times 224^2 = 150.528$	$1.431.693$



Number of parameters of ImageNet state-of-the-art models

NN Finite Sample Expressivity

This explains why NN manage to have 0 training error on random labels

They just have way too many parameters!

Conclusion

CONCLUSION

➜ Authors propose a simple framework to evaluate NN expressivity

Conclusion

- ➜ Authors propose a simple framework to evaluate NN expressivity
- ➜ Deep learning algorithms are large enough to shatter existing datasets, but still generalize to unseen examples

Conclusion

- ➜ Authors propose a simple framework to evaluate NN expressivity
- ➜ Deep learning algorithms are large enough to shatter existing datasets, but still generalize to unseen examples
- ➜ Regularizers is not the reason why

Conclusion

- ➜ Authors propose a simple framework to evaluate NN expressivity
- ➜ Deep learning algorithms are large enough to shatter existing datasets, but still generalize to unseen examples
- ➜ Regularizers is not the reason why
- ➜ It remains easy to converge on data where generalization is impossible

Conclusion

- ➜ Authors propose a simple framework to evaluate NN expressivity
- ➜ Deep learning algorithms are large enough to shatter existing datasets, but still generalize to unseen examples
- ➜ Regularizers is not the reason why
- ➜ It remains easy to converge on data where generalization is impossible
- ➜ Authors show an upper bound on the number of parameter for a NN with ReLU activations to represent any function

Conclusion

- ➜ Authors propose a simple framework to evaluate NN expressivity
- ➜ Deep learning algorithms are large enough to shatter existing datasets, but still generalize to unseen examples
- ➜ Regularizers is not the reason why
- ➜ It remains easy to converge on data where generalization is impossible
- ➜ Authors show an upper bound on the number of parameter for a NN with ReLU activations to represent any function
- ➜ There is more in the paper, but I did not fully understand to present it

Thank you for listening!

Any questions?