DEEP DOUBLE DESCENT: WHERE BIGGER MODELS AND MORE DATA HURT

Nakkiran et al. (2019)

Group 7

Double Descent: An Overview

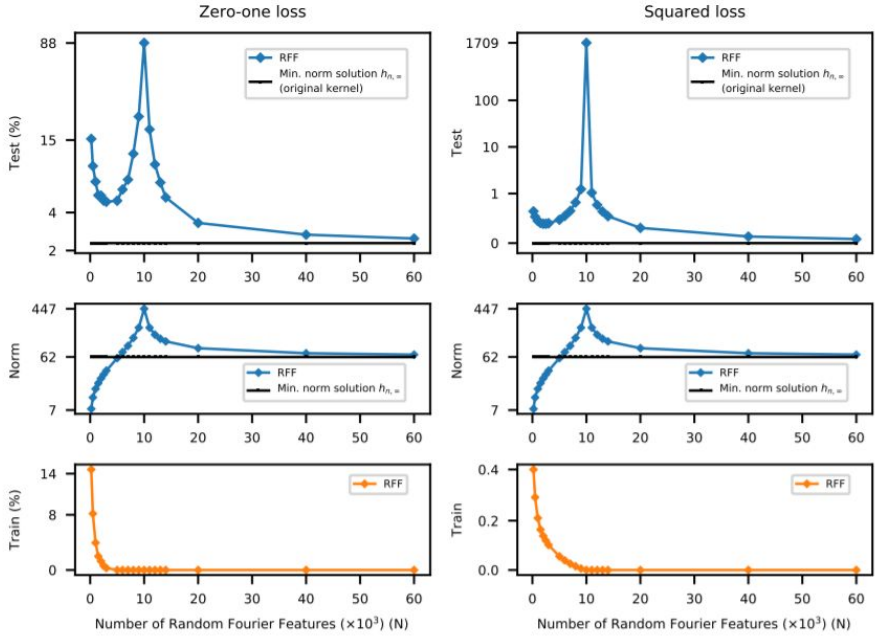
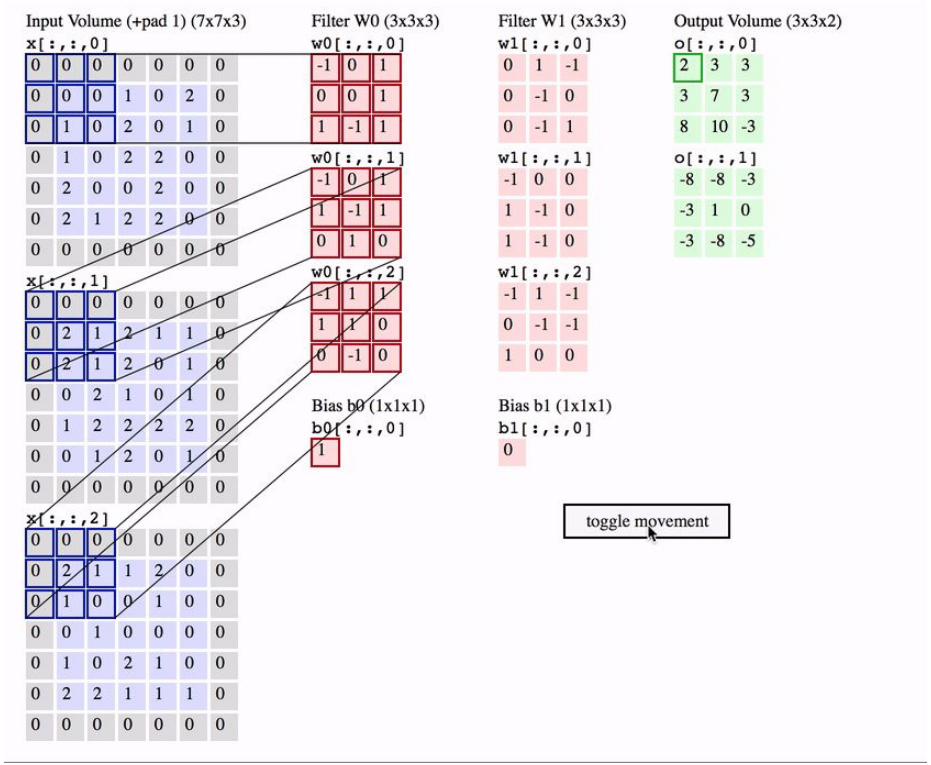


Figure 2: **Double descent risk curve for RFF model on MNIST.** Test risks (log scale), coefficient ℓ_2 norms (log scale), and training risks of the RFF model predictors $h_{n,N}$ learned on a subset of MNIST ($n = 10^4$, 10 classes). The interpolation threshold is achieved at $N = 10^4$.

From Belkin et al. (2018) Reconciling modern machine learning practice and the bias-variance trade-off *PNAS* https://doi.org/10.1073/pnas.1903070116

- Breaks with classical theory: *bias-variance tradeoff*
- Belkin et al. (2018) demonstrate *double descent* of test loss beyond *interpolation threshold: n = p*
- Model underspecified; zero train error
  - minimise Euclidean norm
- Allows reduction of *inductive bias*
- Caveats: SGD sensitive to initialisation in underparametrised setting; computational constraints (e.g. ImageNet)
- Demonstrated in tools often deployed with high *p* e.g. 2-layer neural networks, but also others e.g. decision trees using MNIST and CIFAR-10

Convolutional Neural Networks (CNN)



- Features depend on small neighborhoods of pixels -> Retain locality
- Same features can been located everywhere on the image - Shift / Translation-Invariance

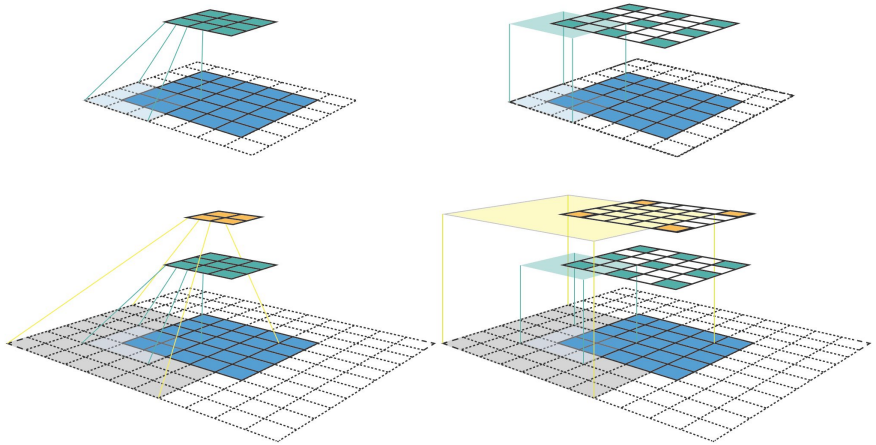
Hyper-parameters

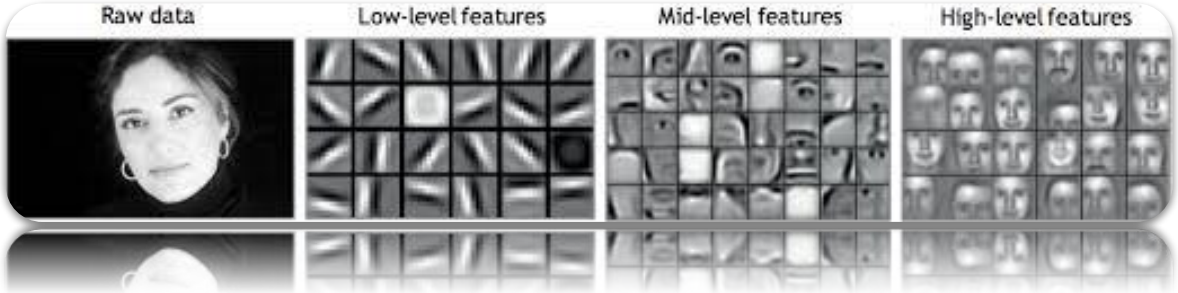
- Width and Height
- Number of input and output channels

Example Applications

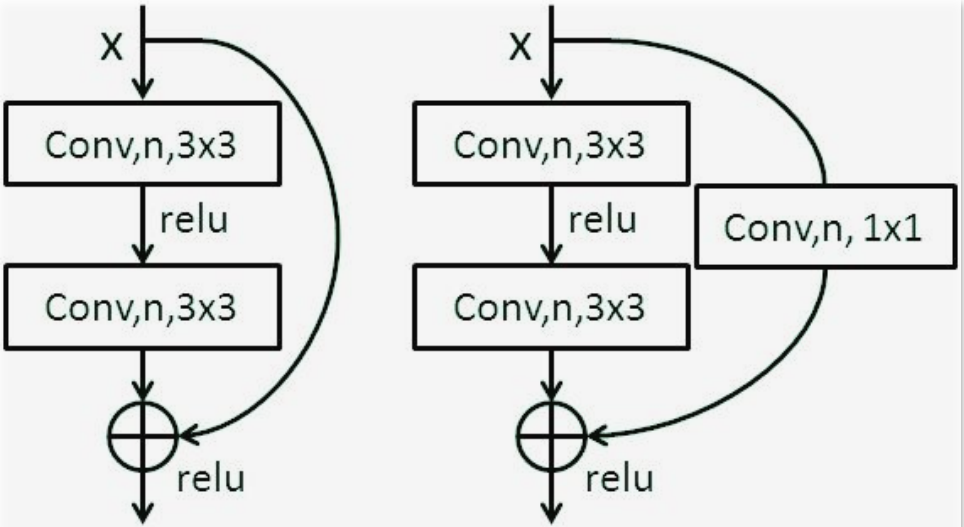
- Image and video classification recognition
- recommender systems
- natural language processing

Features Map and Width Parameter k





The **feature maps** (also *activation maps*) capture the result of applying specific filters that scanning the expanse of the input image to a layer, the feature map is the output of that layer. The reason for visualising a feature map for a specific input image is to try to gain some understanding of what features our CNN detects and discern specific characteristics or attributes.



Residual-Connections:

$$F(x) + x$$

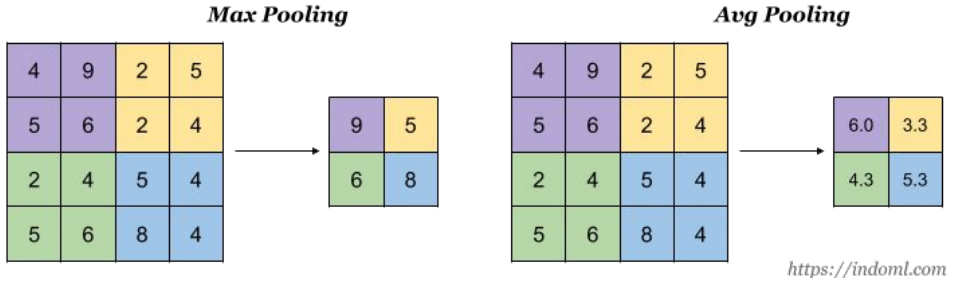
- *Skip connections*, or *shortcuts* to jump over some layers
- Deals with the Vanishing Gradient Problem

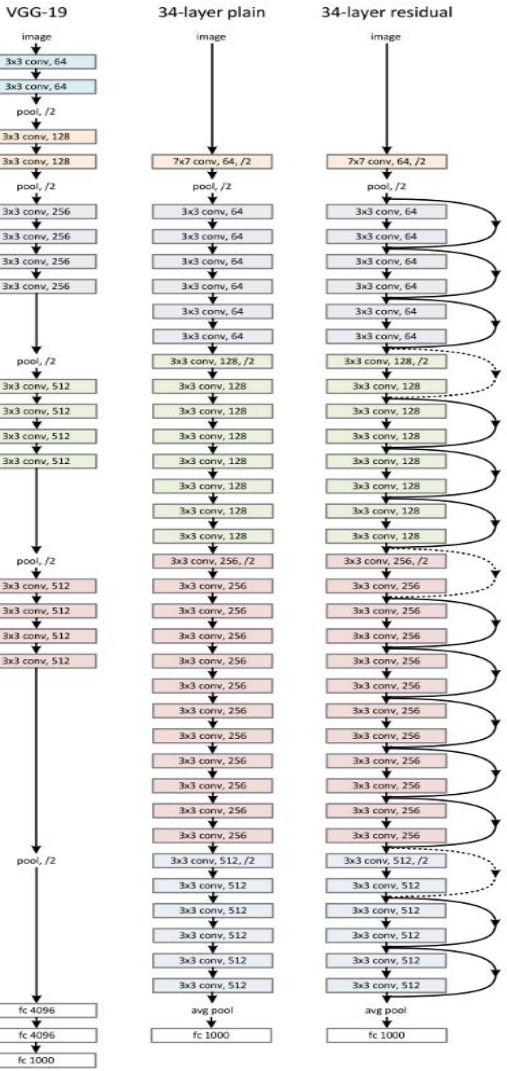
Batch Normalization

- Normalize the input by re-scaling and re-centering
- Fix means and variances at each layer's inputs
- Improving speed, performance and stability of ANN

Pooling Layers

- Reduces the dimensionality of the data
  - *Max* or *Average* Pooling
- Improving speed, performance and stability of ANN

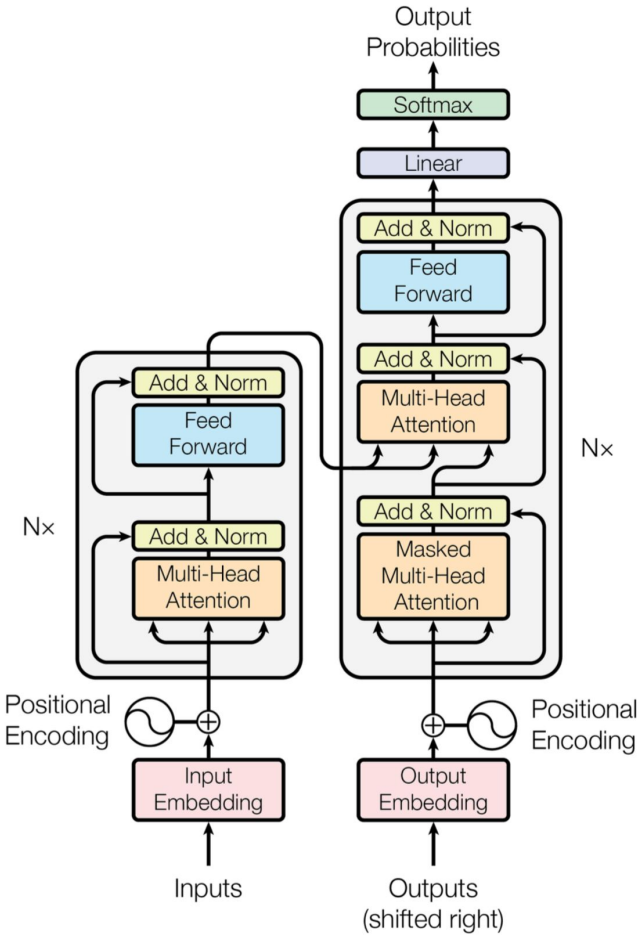




Various Architectures:
AlexNet/ResNet/VGG/GoogleNet

Serial or Parallel Deployment
Of **BLOCKS** (Residual or not) composed of:

- Convolutive Filters
- Max/Average Pooling Layers
- Batch Normalization Layers
- Dropout Layers (Used to avoid model overfitting)
- Activation Functions:
  (*Linear*,*Sigmoid*,*Tanh*,*RELU*,*Softmax*,*ELU*,*Leaky RELU*,...)
- Flattening and feed to common *Fully Connected* neuron layers
- etc...
  (can be arranged in multiple combinations / depends the task)



Transformers

- Attention Based Model Architectures
- Positional Encoding of the embeddings
- Maintain Time-Dependency (like RNN)
- Able to parallelize (per layer)

Currently state of the art for various NLP applications and Neural Machines Translators.

Nakkiran et al. (2019) Deep Double Descent: Key Contributions

- Generalise double descent phenomenon via *effective model complexity*: maximum number of samples on which procedure can reach approx. 0 training error
    - Depends on *training time*, as well as data distribution and classifier architecture
- Demonstrate *model-wise* double descent for DL
- *Epoch-wise* double descent; except for *critically parametrised models*
- Non-monotonicity of loss by sample size: *More data is worse!*
- Investigate effects of label noise (proxy for model misspecification)

...conventional wisdom in classical statistics is that, once we pass a certain threshold, "larger models are worse." However modern neural networks exhibit no such phenomenon. Such networks have millions of parameters...and yet they perform much better on many tasks than smaller models.

Effective Model Complexity

The EMC of a training procedure is defined to be the maximum number of samples on which the training procedures achieve a training error equivalent to 0.

$$\mathrm{EMC}_{\mathcal{D},\epsilon}(\mathcal{T}) := \max\{n \mid \mathbb{E}_{S\sim\mathcal{D}^n}[\mathrm{Error}_S(\mathcal{T}(S))] \leq \epsilon\}$$

- Under-Parameterized
- Over-parameterized
- Critically-parameterized

The label noise

LB comes from several *sources* :

- Encoding problems
- Insufficient information

LB can have several *effects* :

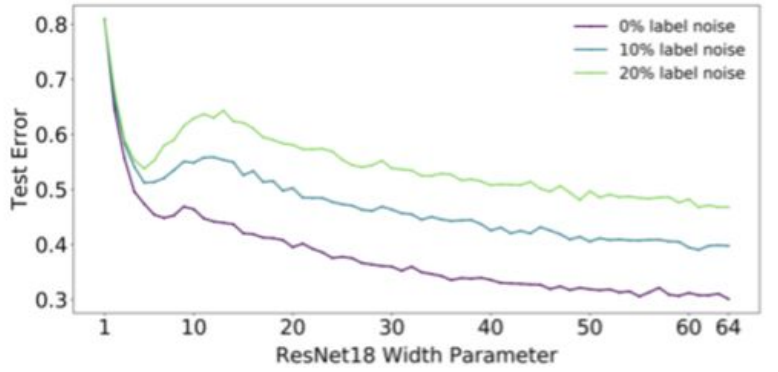
- Decreasing or increasing the complexity of learned models
- Accentuate the Double Descent phenomenon

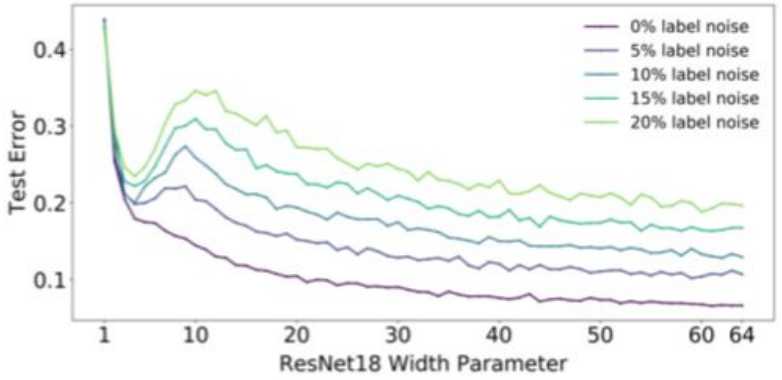
Model-wise : The test error of models of increasing size

Aim : Varying the number of parameters to increase the model size of the model capacity

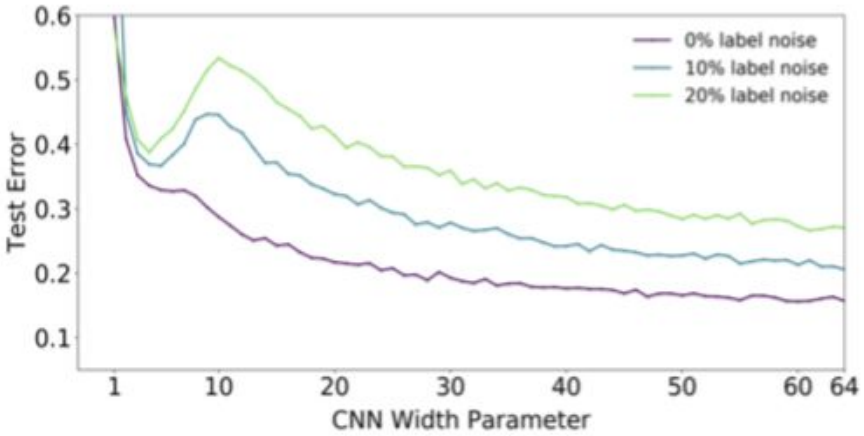
How? By varying the width of the network

It means? Increasing the number of filters of the convolutional layers

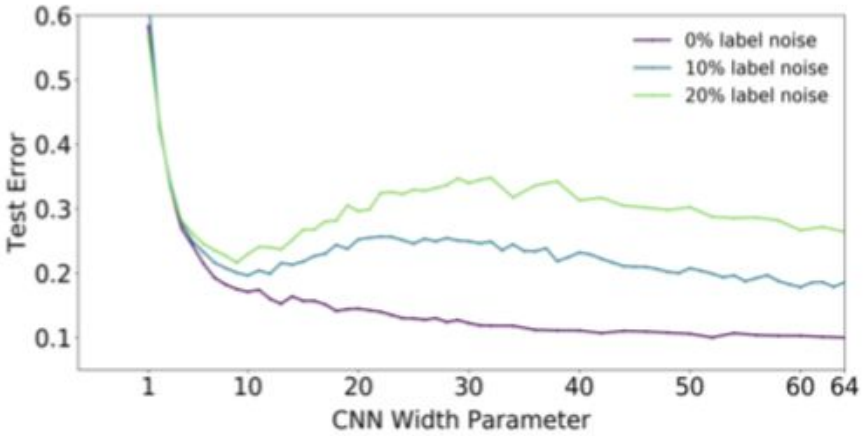




Model-wise : CNN



Without data augmentation



With data augmentation

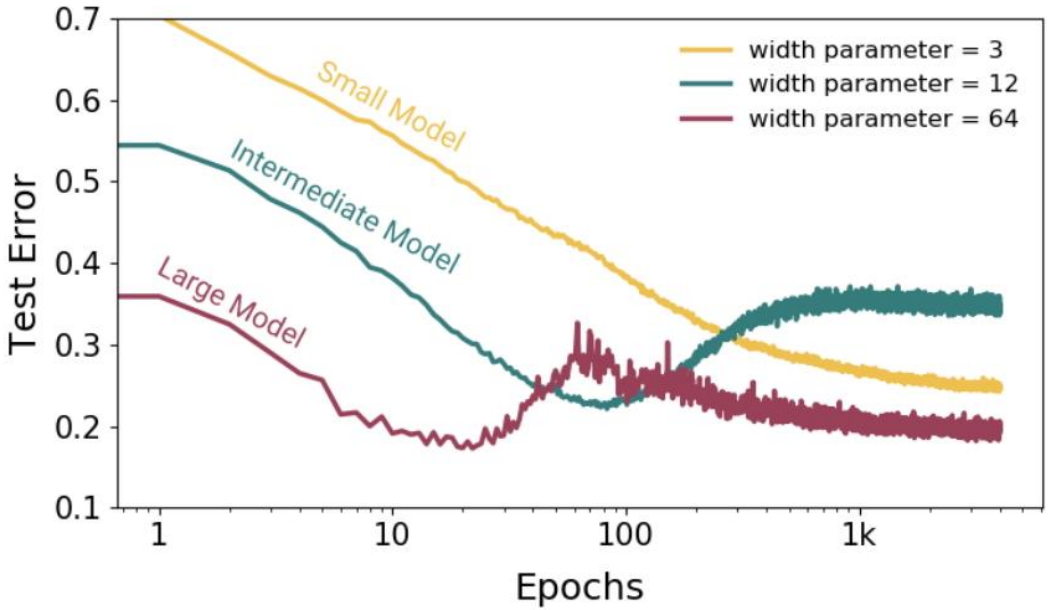
Data augmentation : Process of increasing the amount of training data by creating new data from existing ones

Epoch-wise

- One Epoch is when an entire dataset is passed forward and backward through the neural network only once.

Epoch-wise

- Sufficiently large model can undergo a “double descent” behaviour
- Intermediate Models follow the U curve

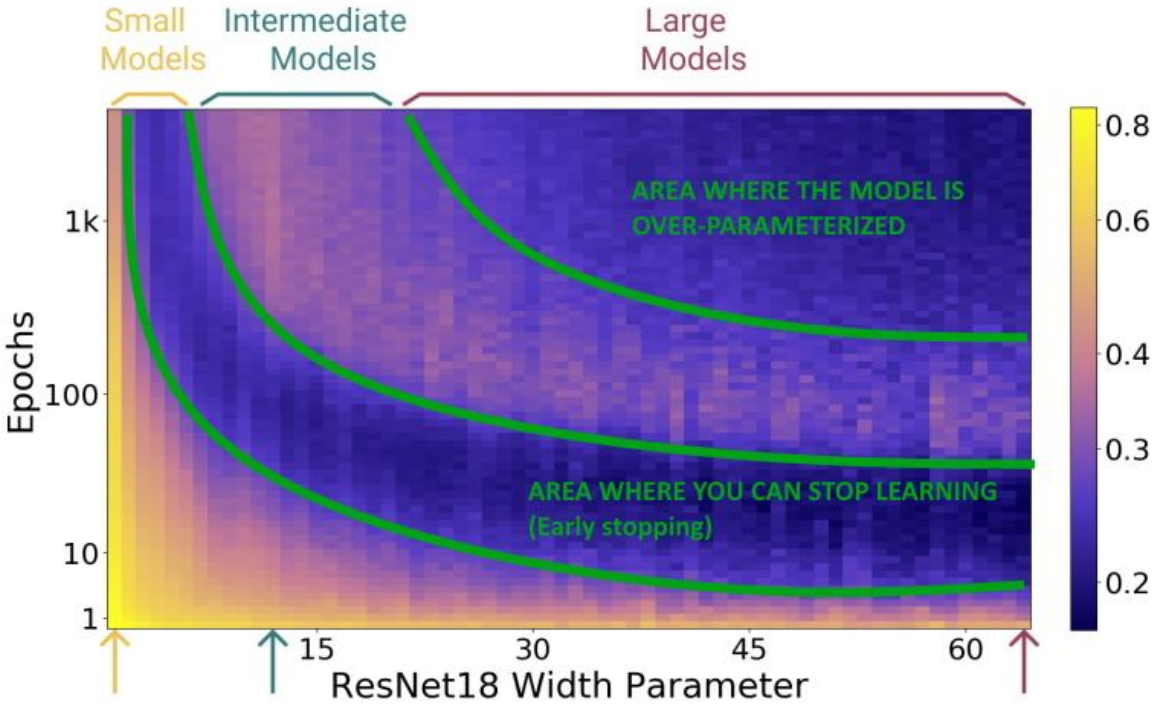


Training dynamics for models in three regimes.

Models are ResNet18s on CIFAR10 with 20% label noise,
trained using Adam with learning rate 0.0001 and data augmentaion.

Epoch-wise

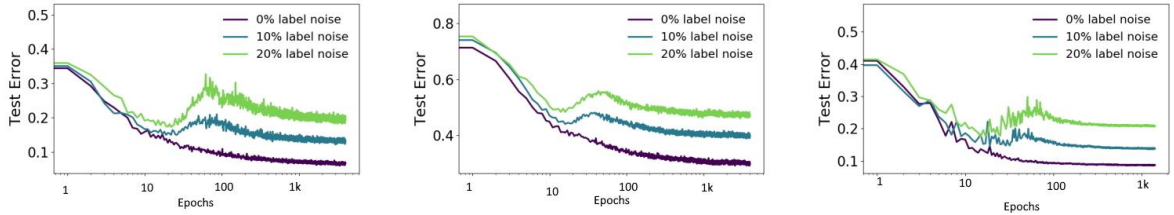
- Early stopping is a form of regularization used to avoid overfitting



Test error over (Model × Epochs)

Epoch-wise

- Convention wisdom will say it is the bias/variance tradeoff.
- First the neural network generalizes, then it over-fit the training data.
- The paper suggests that this knowledge is not always true.



(a) ResNet18 on CIFAR10. (b) ResNet18 on CIFAR100. (c) 5-layer CNN on CIFAR 10.

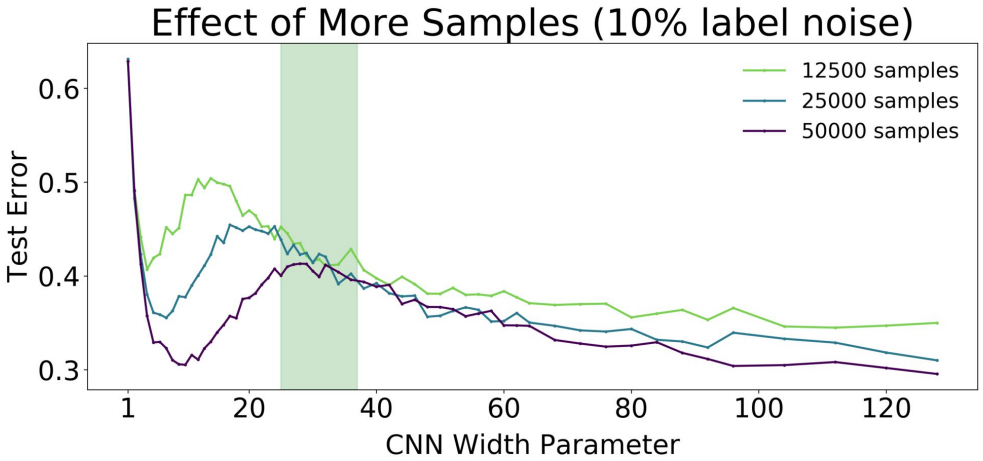
Figure 10: **Epoch-wise double descent** for ResNet18 and CNN (width=128). ResNets trained using Adam with learning rate 0.0001, and CNNs trained with SGD with inverse-squareroot learning rate.

Sample-wise Non-monotonicity

Test Error VS Training Sample Size

Sample-wise Non-monotonicity

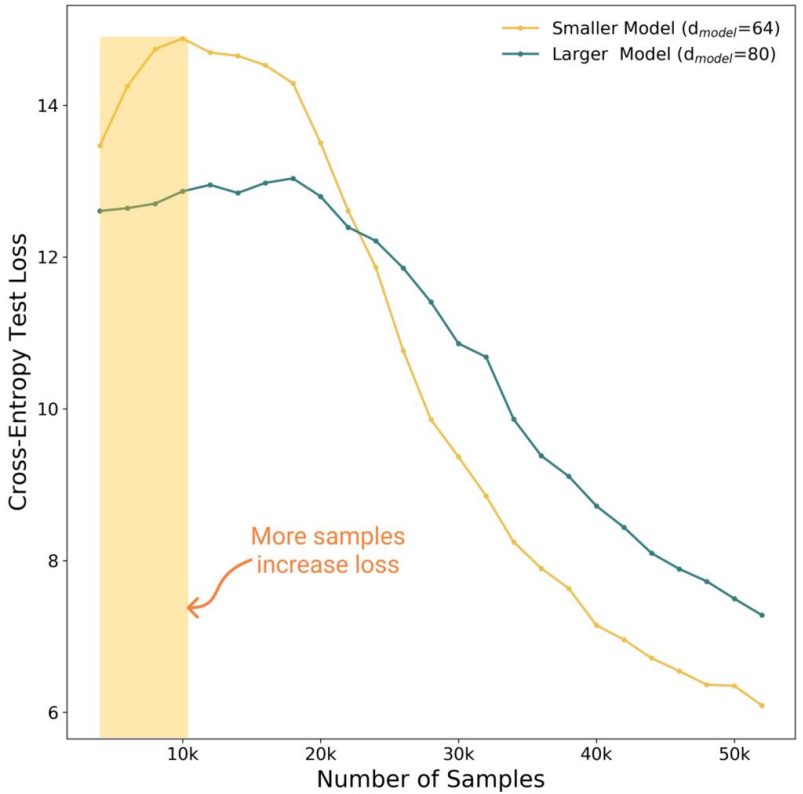
- Adding data points doesn't continually decrease our test error.
- Around the *critical interval* where EMC ≃ n, an increase in the sample size may result to an **increase** of the test error
- An increase of data points causes some kind of "shift" of the curve to the right



Sample-wise Non-monotonicity

Here, when we combine the effects that model size and sample size have on the model, we can see that :

MORE DATA HURTS TEST PERFORMANCE



Criticisms and Perspective

- Computationally infeasible for many datasets or model types
- Sensitivity of some optimisation techniques
- Lack of specification of ε in the effective model complexity metric in a principled way
- Authors do not specify what constitutes a sufficiently distant *effective model complexity* from the sample size, *n*

$$\mathrm{EMC}_{\mathcal{D},\epsilon}(\mathcal{T}) := \max\{n \mid \mathbb{E}_{S\sim\mathcal{D}^n}[\mathrm{Error}_S(\mathcal{T}(S))] \leq \epsilon\}$$