Machine Learning

Gcinizwe Dlamini

INNOPOLIS UNIVERSITY

Today’s Objectives

- A quick recap of the last lecture
- Techniques to improve deep learning Performance
- Data Augmentation, Rescaling and Transformation
- Transfer Learning
- Dropout , Batch normalization, gradient clipping, optimizers and early stopping

Recap

Applications of Convolutions in Images

- Edge Detection
- Image smoothing
- Image sharpening
- Image enhancement

Pooling Layer: Max Pooling

1	3	2	1
2	9	1	1
1	3	2	3
5	6	1	2

→

9	2
6	3

- Filter size and stride length are the hyerparameters of the pooling layer
- There are no parameters to learn
- You do pooling on each channel in the input separately

What is a convolution?

"***In terms of deep learning, an (image) convolution is an element-wise multiplication of two matrices followed by a sum***"

1. Take two matrices (both have the same dimensions).
2. Multiply them, element-by-element (i.e., not the dot product, simple element-to-element multiplication).
3. Sum the elements of the resulting Matrix.

What is a convolution?

$$\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \xrightarrow{\text{element-to-element multiplication}} \begin{bmatrix} 3 & 0 & 1 \\ 1 & 5 & 8 \\ 2 & 7 & 2 \end{bmatrix} = \begin{bmatrix} 3 & 0 & -1 \\ 1 & 0 & -8 \\ 2 & 0 & -2 \end{bmatrix}$$

-5 = Sum all Elements

Recap (2)

Dimensions with Stride

- Input Size: n x n
- Filter Size: f x f
- Padding: p
- Stride: S
- Output Size: $\left\lfloor \frac{n+2p-f}{S} + 1 \right\rfloor$ x $\left\lfloor \frac{n+2p-f}{S} + 1 \right\rfloor$

Padding (Dimensions)

- Input Size: n x n
- Filter Size: f x f
- Padding: p
- Output Size: (n + 2p – f + 1) x (n + 2p – f + 1)

Types of Layers in a Convolutional Net:

- Convolutional Layers (CONV)
- Pooling Layers (POOL)
- Fully connected (FC)

Techniques to improve deep learning Performance

Diagnostics

- Before you start solving a problem, measure level of criticality
- Measure the performance first. Why?
- Select the appropriate evaluation metric : Accuracy, MSE, Precision
- Most deep learning frameworks already have tools for monitoring model performance : TensorBoard, Neptune, Weights & Biases

Levels of improving performance

DATA

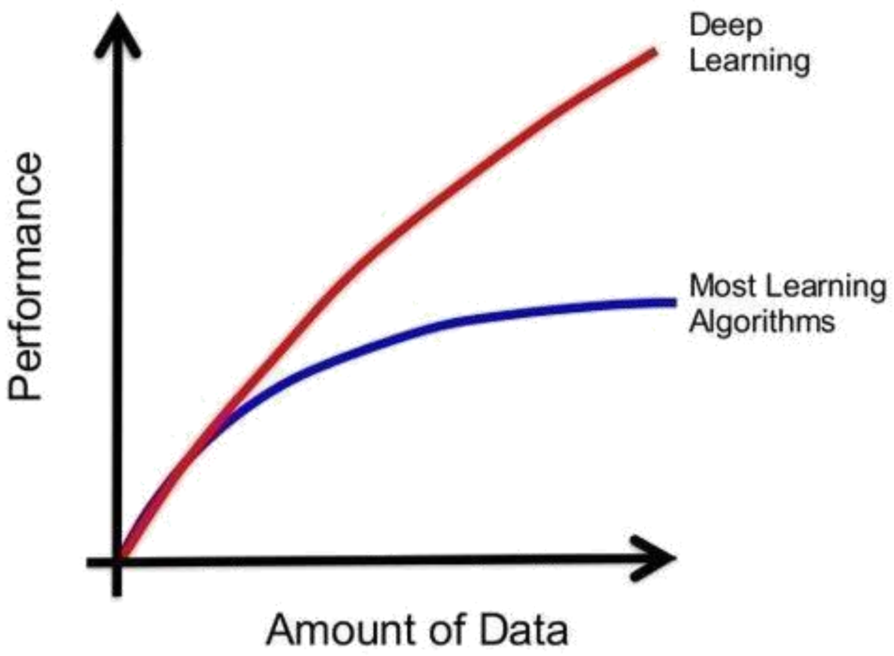
MODEL ARCHITECTURE

MODEL TRAINING

OTHER

DATA LEVEL

Data Augmentation



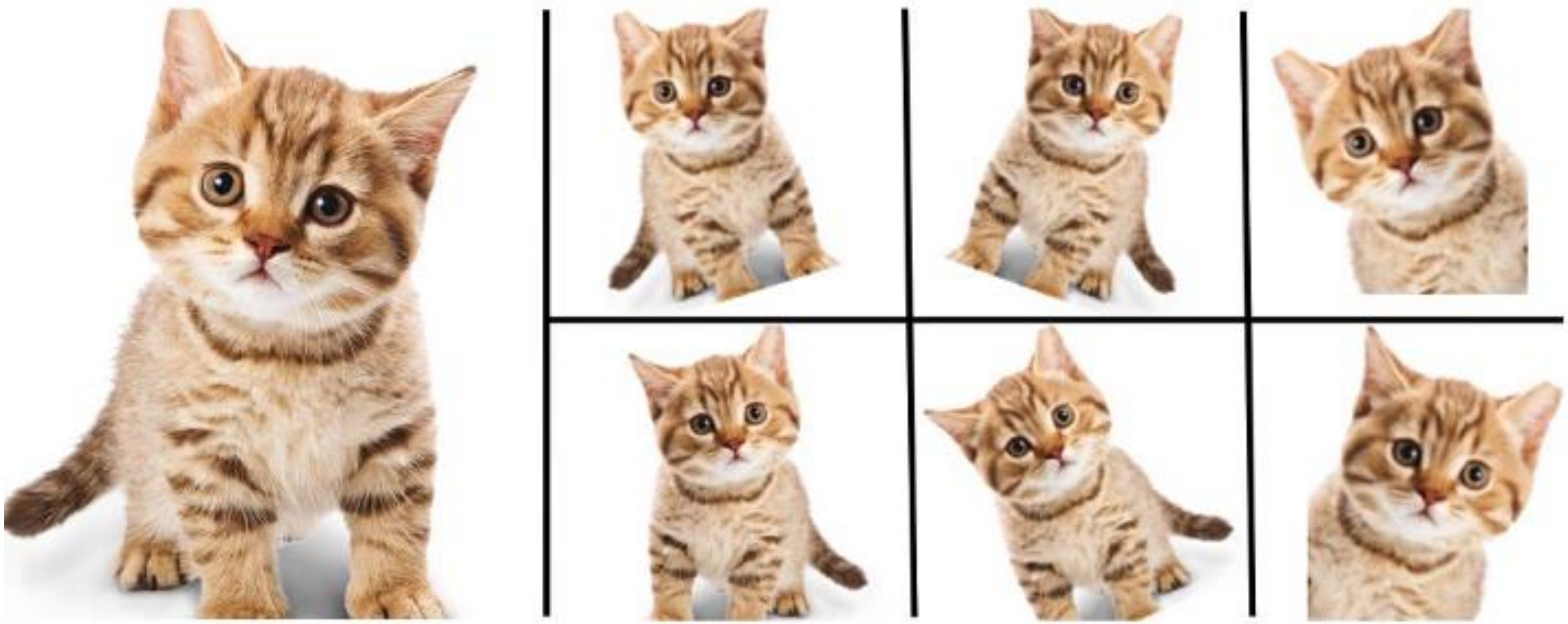
- How machine learning techniques scale with amount of data?
- So let's try to increase our data for our DNN
- **NB**: Creating duplicates will lead to overfitting
- Data augmentation : a smart and simple way that minimizes chances of overfitting

Why Deep Learning? Slide by Andrew Ng

Data Augmentation

- Rotation
- Random cropping
- Whitening
- Center the input mean to zero
- Random Affine transformation
- Random Brightness shifting
- Random width/height shifting
- Custom and more..

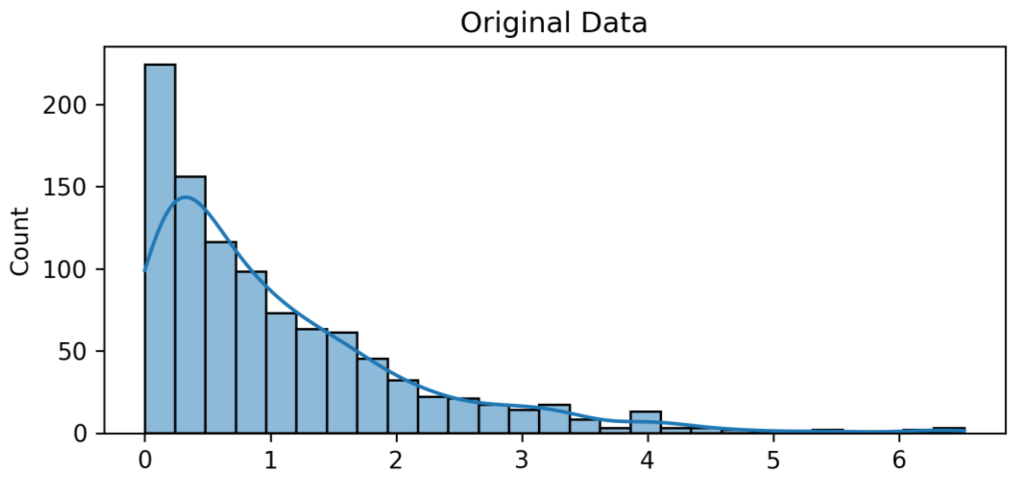
Data Augmentation example

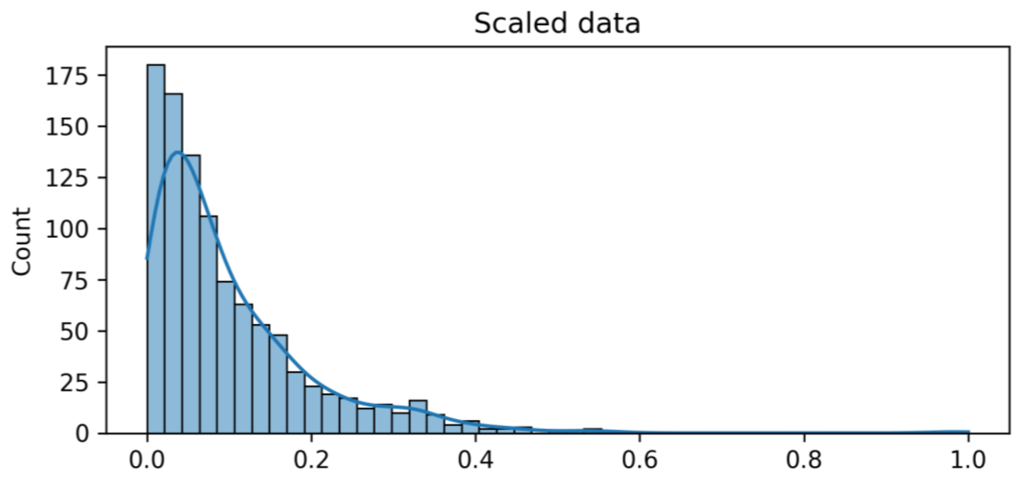


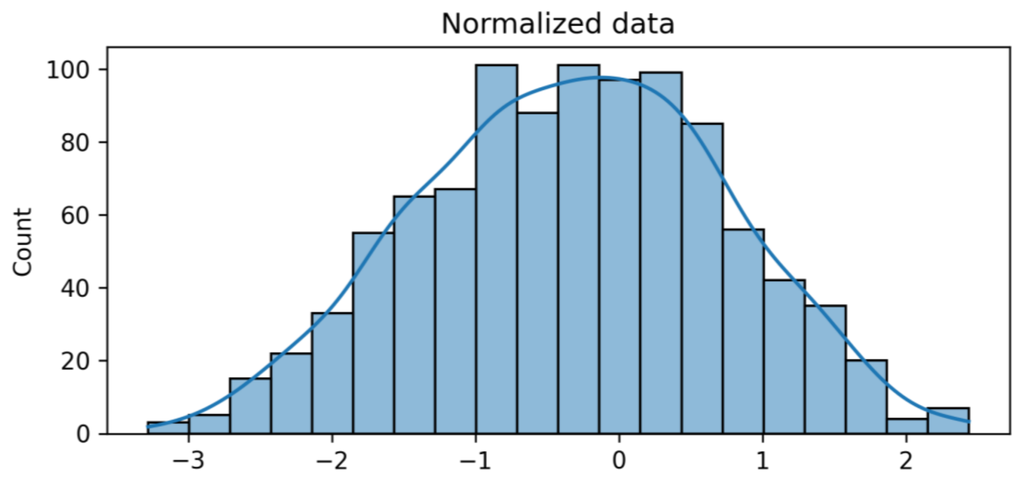
Data Rescaling and Transformation

- Get to know your data before you ask for machine learning model's help
- Normalize to mean 0 and variance 1
- Rescale : -1 to 1
- Evaluate the performance of your model on each technique
- Normalization ensures that there are both positive and negative values used as inputs for the next layer which makes learning more flexible (Need to verify by comparing sigmoid and tanh)

Data Scaling vs Normalization







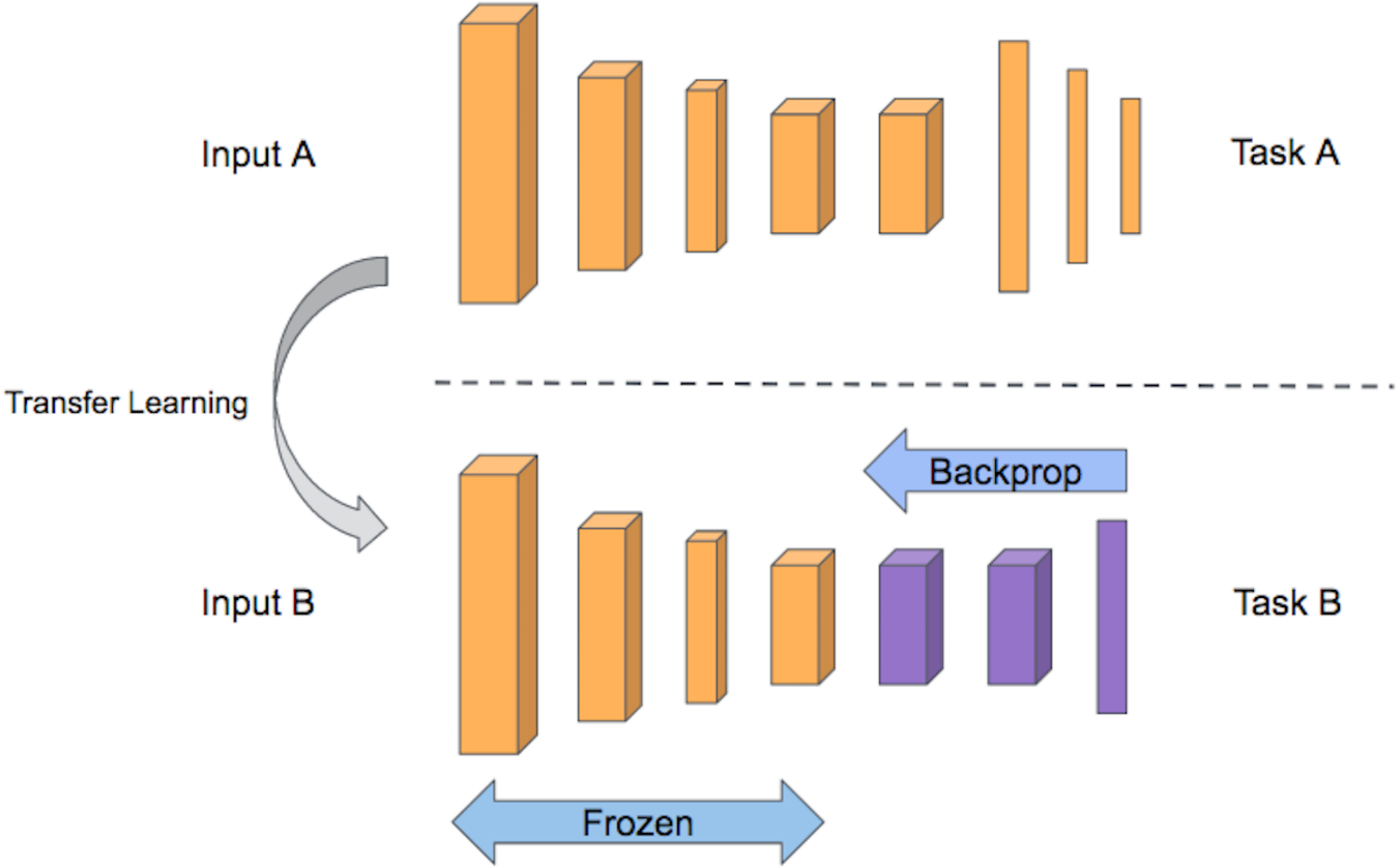
- Scaling changes the range of your data
- Normalization changes the shape of the distribution of your data

MODEL ARCHITECTURE LEVEL

Model architecture

- It depends on many factors : Computational power, problem nature...
- CNN or Simple fully connected
- Long short-term memory (LSTM) or Recurrent neural network (RNN) or Gated recurrent units (GRU)
- Use of pretrained models : Transfer learning
- Use of architecture from literature already tested
- Start with very simple architecture

Transfer Learning



MODEL TRAINING LEVEL

Model training components

- Weight Initialization
- Gradient Clipping
- Optimizers, Loss and Learning Rate
- Activation Functions
- Network Topology
- Batch normalization
- Regularization : L1, L2 & Dropout
- Early Stopping

Dropout

- Type of regularization of the network to reduce the overfitting
- Technique applied only in model training stage
- Prerequisite : Bagging (to be covered in detail in lecture 10 and 11)
- **Bagging** is an ensamble method that average the predictions of many high-variance learners so the collective prediction has lower variance.

Dropout (2)

- Let the predictions of each learner be a random variable and all of them are $i.i.d.$ with variance σ^2 so the variance of their mean:

$$var\left(\frac{1}{n}\sum x_i\right) = \frac{1}{n^2} var\left(\sum x_i\right) = \frac{1}{n^2}\sum var(x_i)$$

$$\frac{1}{n^2}\sum \sigma^2 = \frac{1}{n^2} * n * \sigma^2 = \frac{\sigma^2}{n}$$

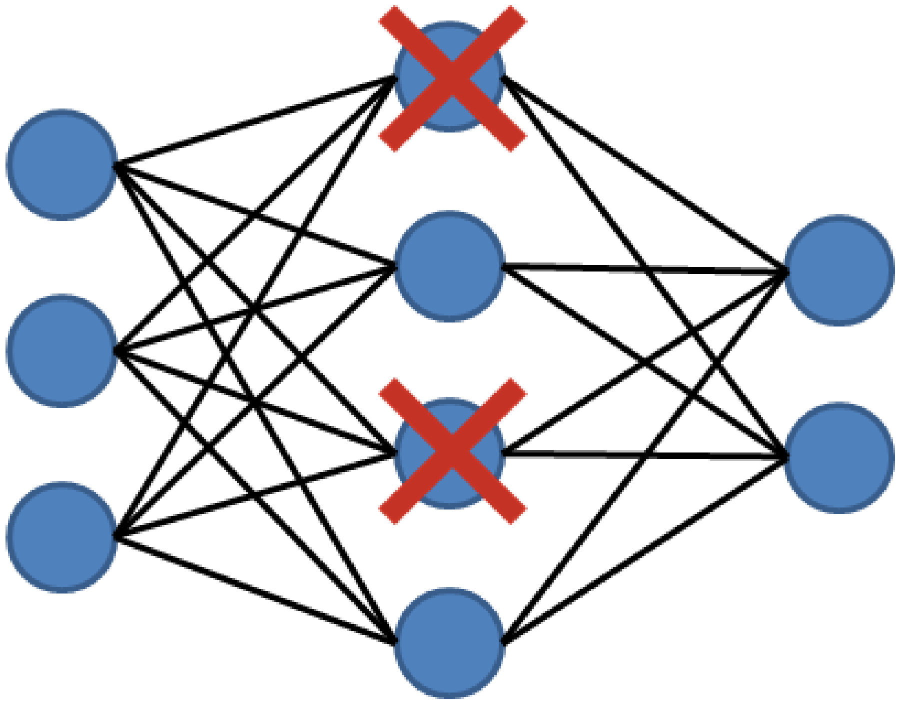
Variance of the mean gets to be less, which means less overfitting

Dropout (3)

- Approximation of the process of bagging by randomly removing nodes from the neural network, so all possible produced networks are theoretically enough to make prediction (creating subgraphs)

- Turning off/zeroing/removing random nodes (with probability p) and stop their forward propagation to the next layers, which drives the rest nodes in the network to form sufficient model on its own to give predictions.

Dropout (4)



- Usually the dropping probability p is 0.5 or lower
- Researchers suggest values between 0.2 and 0.5
- Dropout is commonly used after dense (fully-connected) layers and rarely used after convolutional layers

No dropout on output layer

Batch normalization

- **Problem :** the model is updated layer-by-layer backward from the output to the input using an estimate of error that assumes the weights in the layers prior to the current layer are fixed
- **Covariate shift** : the distribution of input data shifts between the training environment and deploy environment (i.e. speech and facial recognition, translation software, etc.)
- Accelerates training, in some cases by halving the epochs or better, and provides some regularization, reducing generalization error
- It has a great effect in stabilizes the training process and reduce the number of epochs needed to converge

Reference and More details

Batch normalization (2)

- Standardizes the input mini-batch before being fed to the next layer
- For inferencing Exponential Moving Average of the mean and variance are calculated
- γ and β are learned parameters

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1\ldots m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Algorithm 1: Batch Normalizing Transform, applied to activation x over a mini-batch.

Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift

Gradient clipping

- This is a way of preventing the gradient from exploding and going out of control in deep architecture specially in recurrent neural nets
- **Clip by norm**
  - if the l2-norm of the gradient tensor is more than a specific value we normalize the gradient tensor with this equation:

$$gradient = gradient \ * clipnorm \ \div \ \| \, gradient \, \|_2$$

- **Clip by value**
  - if the value at any index in the gradient tensor is greater than a specific value or less than the negative of the same value normalize the gradient tensor with this equation:

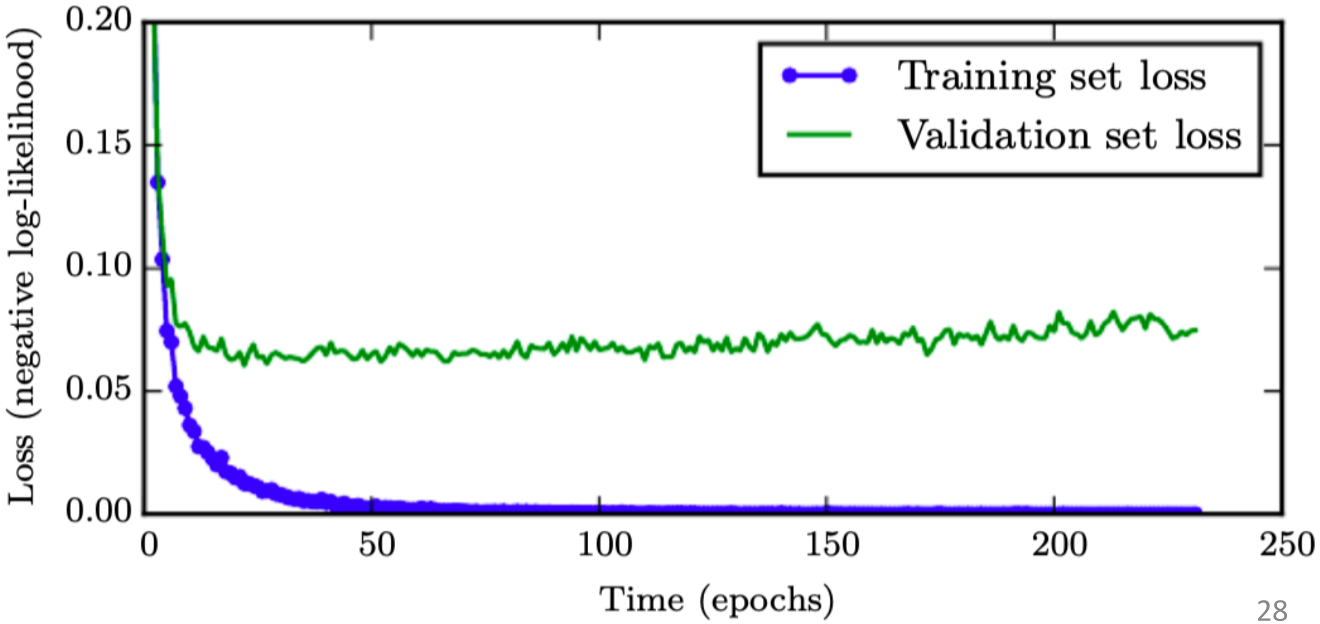
$$gradient = clip(gradient, -clipvalue, clipvalue)$$

Early Stopping

- Stop learning once performance starts to degrade
- It’s a type of regularization
- Requires monitoring the model performance & validation data
- Requires **checkpointing**
- Checkpointing allows the saving of model multiple times if a specific condition is met. Multiple models a produced in one training session.

Early Stopping

- While training large capacity neural network usually overall training loss will be steadily decreasing, however, the validation loss will take a kind of U-shape curve decrease then increase.
- Stop the training at the lowest validation loss point



[Must read Resource]

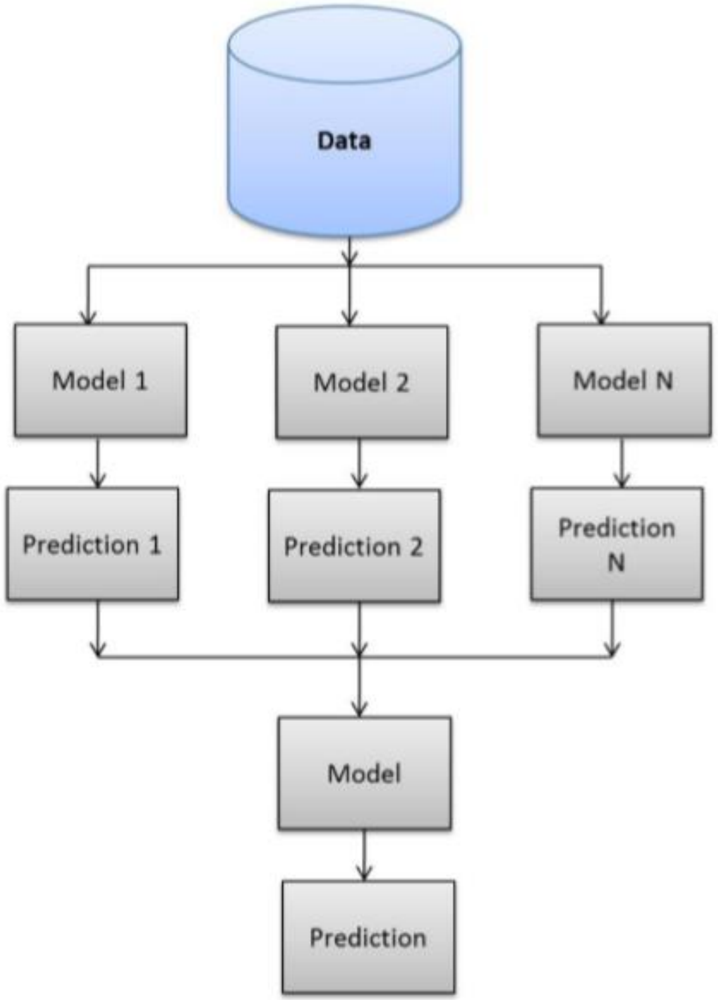
Optimizers

- Stochastic Gradient Descent is the default.
- Use different learning rates together with learning rate schedules.
- SGD has constant learning rate : **Disadvantage** .. Why?
- Try adaptive optimizers (i.e Adam, Nadam, RMSProp)
- **Adaptive optimizers** compute the step size using moving average which indicates show how much we need to reach the minimal point

OTHER

Other techniques

- Create an Ensemble model from multiple DNN’s. More on ensemble in coming lectures
- Reframe the problem
- Try Staking : **Stacking** is an ensemble learning method that combines multiple machine learning algorithms via meta-learning



Summary

- Techniques for improving DNN's performance
- Data Level : Data Augmentation, Data Rescaling and Transformation
- Architecture Level : Transfer Learning
- Model Training Level : Dropout, Batch Normalization, Gradient clipping, Optimizers, Early stopping
- Other techniques : Staking, ensemble and problem reformulation

Thank You