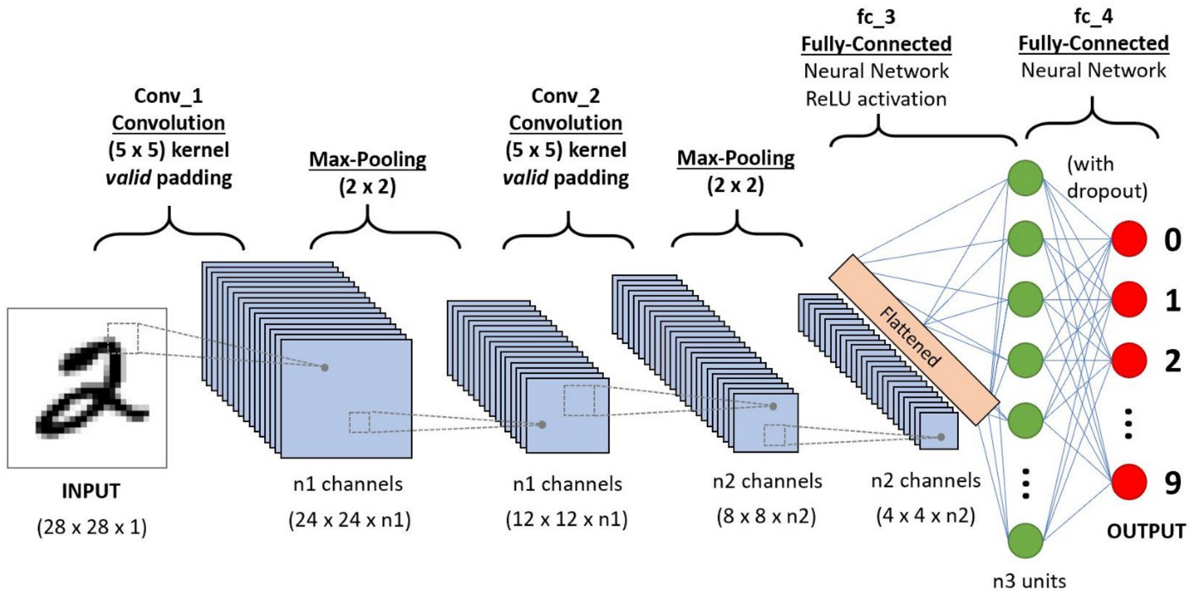
COMP 642 — Machine Learning Mar 8, 2022

Lecture 17: Convolutional Neural Networks

Lecturer: Anshumali Shrivastava
Scribe By: Fang-Yu Lin, Wei Ren Gan, Sean Lin

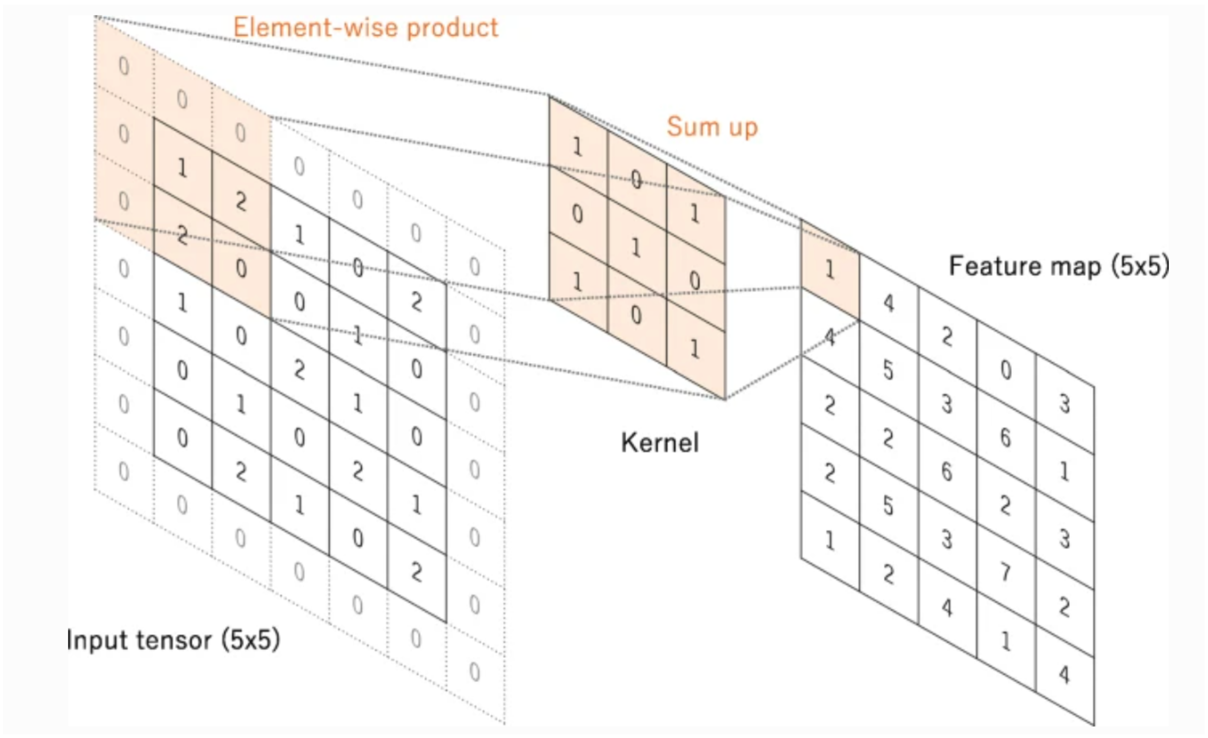
1 Introduction



A Convolutional Neural Network(CNN/ConvNet) is a class of deep learning model, which is composed of multiple building blocks(layers), typically including convolution layers, pooling layers, and fully-connected layers. CNN could take data which has a grid pattern, such as images, as inputs and automatically and adaptively learn spatial hierarchies of features through a backpropagation algorithm.

1.1 Convolution Layer

The convolution layer is the core building block of a CNN. It takes input data, filters(kernels) and produce feature maps.



A convolution operation with zero padding so as to retain in-plane dimensions.

1.1.1 Convolution

The objective of kernels(filters) is to capture features. It would move across the receptive fields of the input image, calculate the dot product between the input pixels and the filter, and feed the result into an output array, which is also called a feature map, to check whether the feature is present in part of the image. Since each filter will yield one corresponding feature map, the number of filters affect the depth of the output.

1.1.2 Stride

Stride determines the distance between two successive kernel positions on an input matrix. The common choice of stride is 1. A stride of 2 or greater is rare, sometimes it would be used to achieve downsampling of the feature maps.

1.1.3 Padding

Padding is usually used when the filters do not fit the input images.Modern CNNs usually employ zero-padding to retain in-plane dimensions. Without padding, each successive feature map would get smaller after the convolution operation.

- Valid-padding This is also know as no padding. In this case, the convolved feature will be reduced in dimensionality as compared to the input.
- Same-padding This padding ensures that the output has the same size as the input.
- Full-padding This type of padding will increase the size of the output by adding zeros the the border of the input.

1.1.4 Nonlinear Activation Function

After each convolution operation, the feature map will be passed through a non-linear activation function, which introduces nonlinearity to the model. Activation functions commonly applied to neural networks include rectified linear unit(ReLU), sigmoid, and hyperbolic tangent(tang).

$$ReLU(x) = f(x) = max(0, x) \tag{1}$$

$$sigmoid(x) = \frac{1}{1+e^{-x}} \tag{2}$$

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{3}$$

The most common nonlinear activation function used presently in CNN is Relu.

1.2 Pooling Layer

Similar to convolution layer, pooling is also a way of downsampling. By sweeping the filter across the input, pooling layer could also help on reducing the dimension of the feature maps. This can help on speeding up the computation. However, unlike convolution layer, instead of applying weight onto the filter, pooling layer applies aggregation function such as average and maximum while generating the output. In general, the hyperparameters for pooling include filter size, stride, and average or max pooling.

- Filter Size (f) : The dimensionality of the filter.
- Stride (s) : The number of pixels shifts over the input matrix

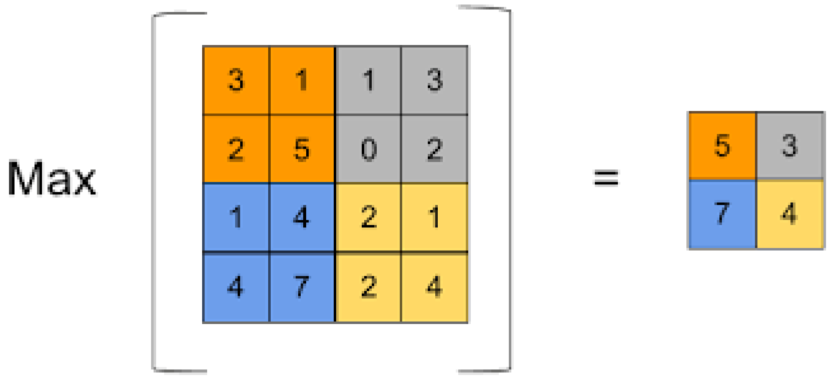
$$inputsize = n_h * n_w \tag{4}$$

$$outputsize = \frac{n_h - f}{s+1} * \frac{n_w - f}{s+1} \tag{5}$$

The followings are the two main types of pooling, using either average or maximum while pooling.

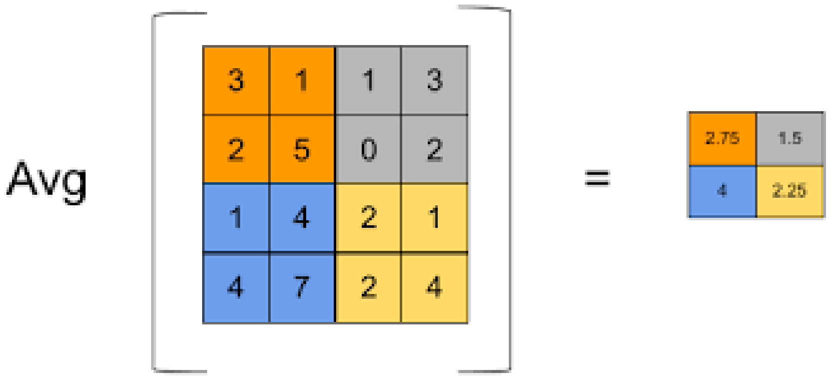
1.2.1 Max Pooling

Max pooling applies maximum function while operating. It returns the maximum value of each portion of the matrix while the filter moves over the input. Max pooling also perform as noise suppressant because it highlight the most present feature in each portion of matrix.



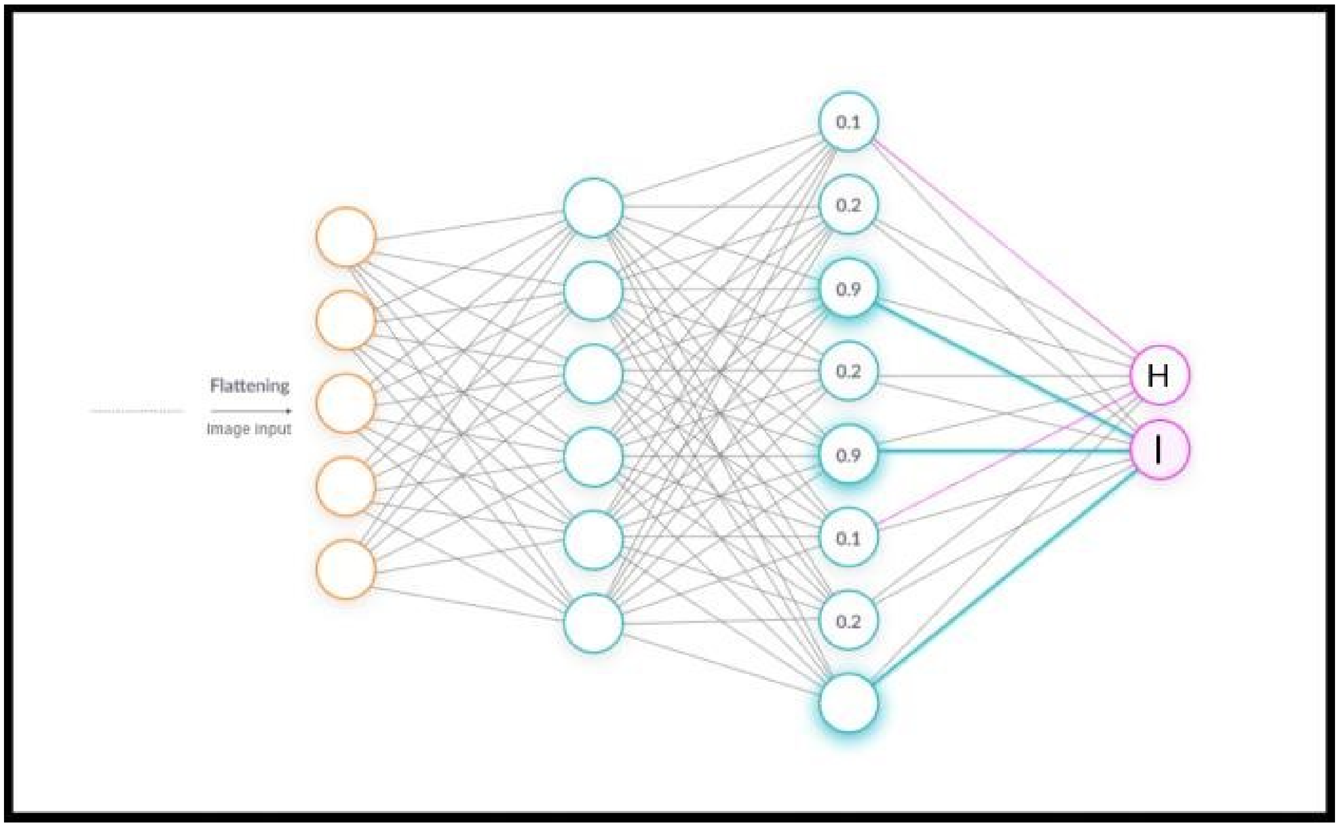
1.2.2 Average Pooling

Average Pooling pooling applies average function while operating. It returns the average value of each value within the portion of the matrix while the filter moves over the input.



1.3 Fully-Connected Layer

Fully connected layers are usually the last few layers in the neural network. It takes the output of the last pooling layer or last convolution layer as input, and flatten it into an one dimension array. After flattening the array, it is then fed into one or multiple fully connected layers, which is also called dense layer. In the fully-connected layers, each node is connected to the previous node with a learnable weight.



1.3.1 Last Layer Activation Function

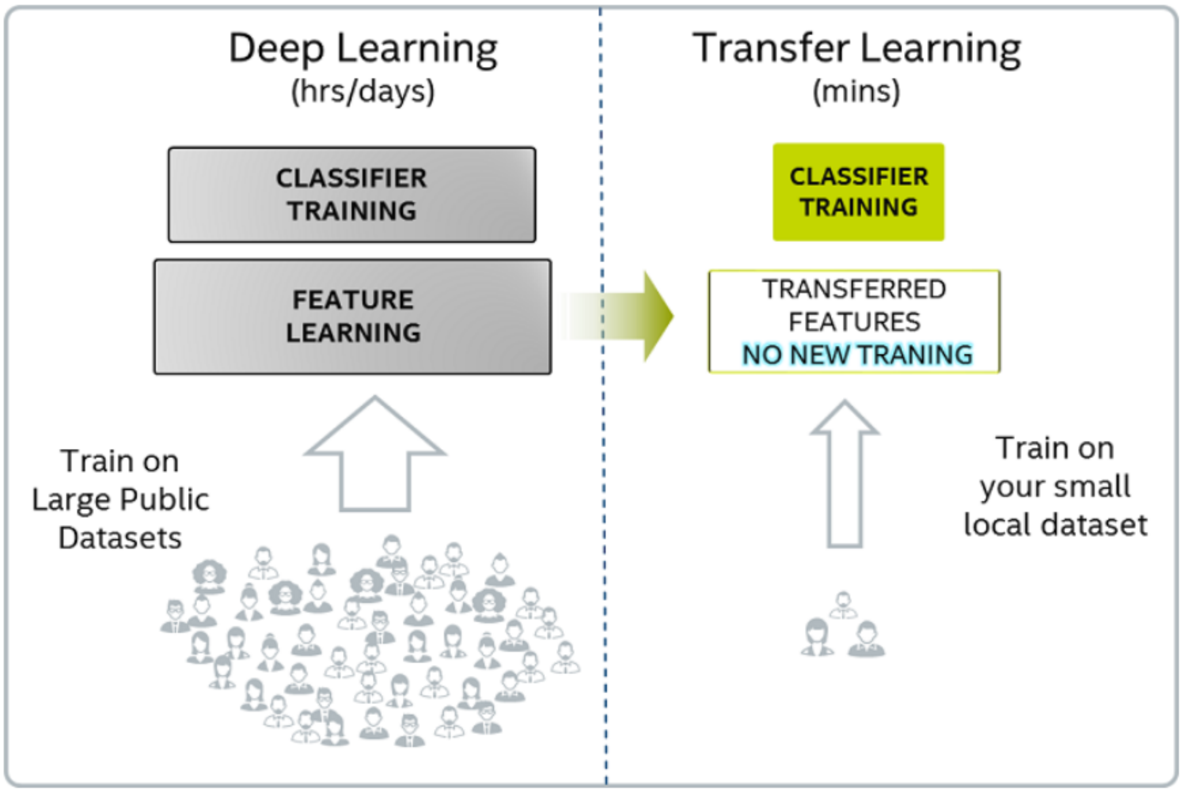
While both convolution layer and pooling layer usually use ReLu as the nonlinear activation function, the last layer activation function tend to use different activation function from them. One of the most common activation function is Softmax activation function, that produce a probability from 0 to 1. The following table is some commonly applied activation function that is used in the last layer.

Task	Last layer activation function
Binary classification	Sigmoid
Multiclass single-class classification	Softmax
Multiclass multiclass classification	Sigmoid
Regression to continuous values	Identity

2 Rules of Thumb in Machine Learning Exploration

2.0.1 Transfer Learning

When starting with a new problem, one should use existing data, data pipeline and solutions. The idea behind transfer learning is that we use the knowledge a model has gathered training on a specific task to solve a different but related task. Using a pre-trained model, the model can profit from the things it has learned from the previous task to learn the new one faster. If the two models are developed to perform similar tasks, then generalized knowledge can be shared between them. The left portion of the image below indicates the pre-trained model, using large volume of samples in the dataset. Whereas in the right portion and depending on the transfer learning scenarios, one or more layers might be added or replaced in the architecture. The usual case is that we only replace the last fully-connected layer.



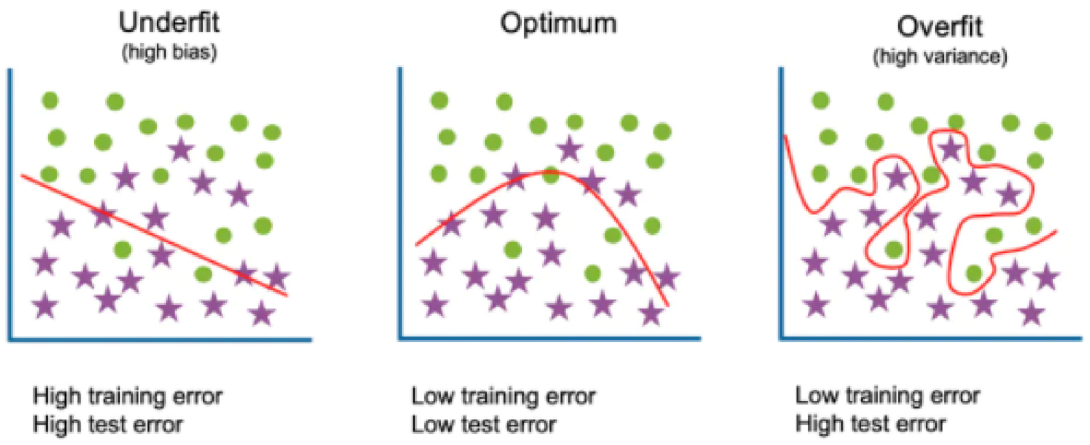
2.0.2 Overfitting

Overfitting is a concept which occurs when a statistical model fits exactly against its training data. When this happens, the algorithm cannot perform accurately against unseen data, defeating its purpose. Generalization of a model to new data is ultimately what allows us to use machine learning algorithms every day to make predictions and classify data.

Reasons for overfitting and ways to handle it:

- Model capacity high: The model is too complex. Reducing model capacity can help in the case of overfitting.
- Mismatch pipeline data: If model performs poorly on the validation set, the problem could come from data mismatch. Additional factors include useless, un-cleaned and noisy data during training or model learned with that data is not implemented with the correct pipeline.
- Data augmentation: Data augmentation is a technique used to increase the amount of data by adding slightly modified copies of already existing data or newly created synthetic data from existing data. It acts as a regularizer and helps reduce overfitting when training a model.
- The size of the training dataset used is not enough. Train with sufficient data.
- The model has high variance (see far right on image below).
- Use K-fold cross validation

- Using Regularization techniques such as Lasso and Ridge, although it may not be the most helpful
- Adopting ensembling techniques

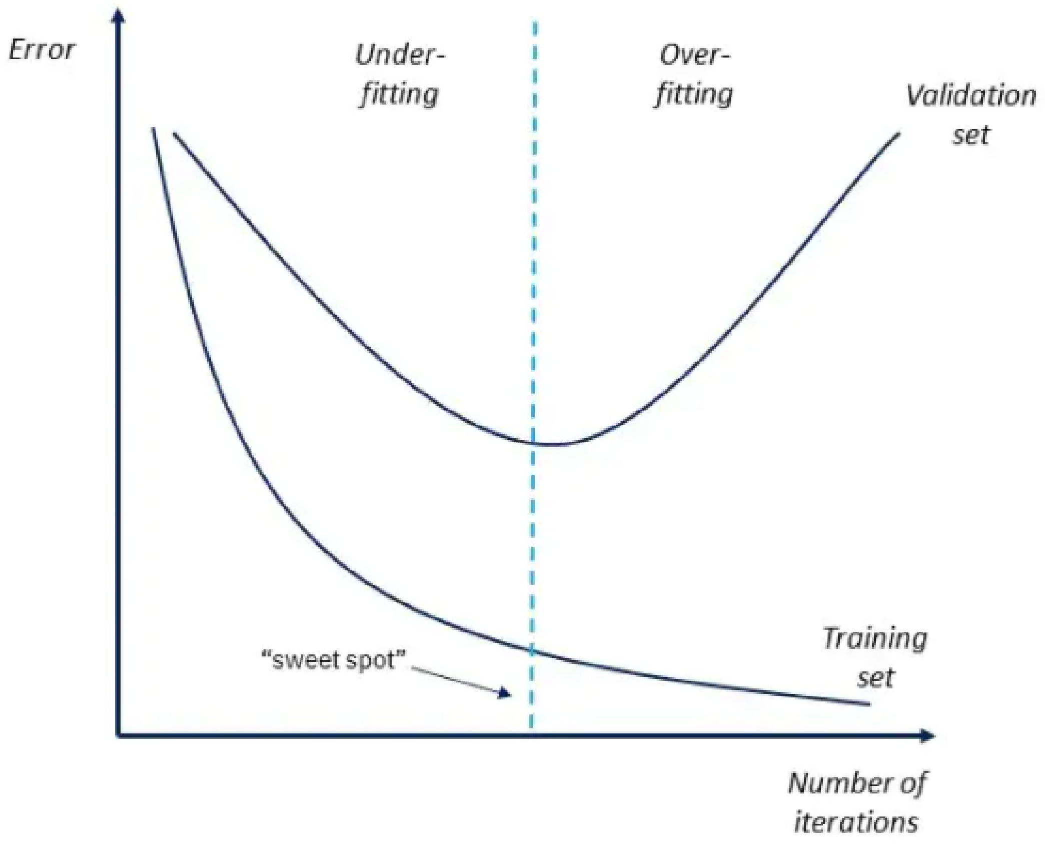


2.0.3 Underfitting

Underfitting occurs when the data model is unable to capture the relationship between the input and output variables accurately, generating a high error rate on both the training set and unseen data. Like overfitting, when a model is underfitted, it cannot establish the dominant trend within the data, resulting in training errors and poor performance of the model. If a model cannot generalize well to new data, then it cannot be leveraged for classification or prediction tasks.

Reasons for underfitting and ways to handle it:

- Model capacity low: The model is too simple. Increasing model capacity (input features) can help in the case of underfitting.
- Mismatch pipeline data: If model performs poorly on the validation set, the problem could come from data mismatch. Additional factors include useless, un-cleaned and noisy data during training or model learned with that data is not implemented with the correct pipeline.
- Optimization issues: Fine-tune hyperparameters such as learning rate, momentum, mini-batch size, number of layers, number of hidden units, learning rate decay and more.
- The model has high bias (see far left on image above).
- Increase the duration of training the data.
- Reduce noise in data.



Notice that the underfitting has high bias in both the training and validation sets. Whereas in overfitting, there is low bias in the training set and high bias in the validation set.

2.0.4 Handling data mismatch

- Check if the data is not random (sanity check).
- Check the representation. For example: if the desired similarity (σ_i, σ_j) is high, then the observed similarity in representation should be higher than random.
- Carry out manual error analysis to try to understand difference between training and dev/test sets.
- Make training data more similar, or collect more data similar to dev/test sets

3 References

- https://medium.com/nerd-for-tech/transfer-learning-7914c6ab2b56
- https://www.ibm.com/cloud/learn/overfitting
- https://www.simplilearn.com/tutorials/machine-learning-tutorial/overfitting-and-underfitting
- https://www.coursera.org/lecture/machine-learning-projects/addressing-data-mismatch-biLiy
- https://androidkt.com/explain-pooling-layers-max-pooling-average-pooling-global-average-pooling-and-global-max-pooling/
- https://www.researchgate.net/figure/Fully-connected-layer_fig3_343263135
- https://insightsimaging.springeropen.com/articles/10.1007/s13244-018-0639-9
- https://www.ibm.com/cloud/learn/convolutional-neural-networks
- https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53
- https://www.analyticsvidhya.com/blog/2018/12/guide-convolutional-neural-network-cnn/