Program syllabus for : Classroom

Varun Kapoor

Kapoorlabs, Paris, France

February 2022

1 Introduction to AI based image analysis

1.1 Package installation

1.2 Creating training data for instance segmentation

1.3 Creating training data for image classification

1.4 Bias Variance tradeoff

1.5 Keras features for model training

1.6 Using keras to create encoder-decoder networks

1.7 Model hyperparameters

1.8 Activation functions

Activation functions of the last layer are a crucial choice and depends on the task, if it is a segmentation task the final activation is linear as it estimates the mean value of the Gaussian distribution (assumption being that the data generating distribution is Gaussian). If the task is image classification the last activation function is either softmax or sigmoid. The activation functions are strongly coupled to the choice of the training loss. For segmentation task the training loss is mean absolute or mean squared error while for classification tasks is categorical or binary cross entropy.

1.9 Before Training: Over/Underfitting

Before starting the model training, it is essential to split the data into training and validation datasets. This is to perform cross validation to judge if the model is over/under fitting on the training dataset. Such information can be seen in the training and validation loss curves.
Your model is overfitting on the data if you see the validation loss curve rising instead of following the training loss curve. This gap between the two is called the generalization gap and is a result of high variance of the estimator 1.4. High variance comes due to large coefficients in the weight vector. Regularization is a technique used to penalize such large coefficients in the weight vector.

Increasing the regularization coefficient adds to the penalty being higher, **increasing regularization coefficient helps avoid over fitting**. Your model can be overfitting on the data if the **training data is not enough**, **either increase the training data or use data augmentation** to avoid overfitting. It is also a good idea to check if your **model is too complex**, reducing the number of hidden units (reducing depth, number of convolutional filters) also helps in avoiding overfitting. You can also add a **keras callback for early stopping to avoid overfitting**. The opposite sitution to this is underfitting. **Your model is underfitting on the data** if you see high training and validation loss that does not improve. This situation arises due to high bias of the estimator 1.4. In this case **increasing the model complexity** by adding more hidden neurons, increasing the network depth and or the number of convolutional filters would help. If you are using regularization then **decreasing the regularization coefficient** and increasing the training time simultaneously would avoid this problem.