ASSIGNMENT-LAB 07

GEETHU SUNNY

21BDA03

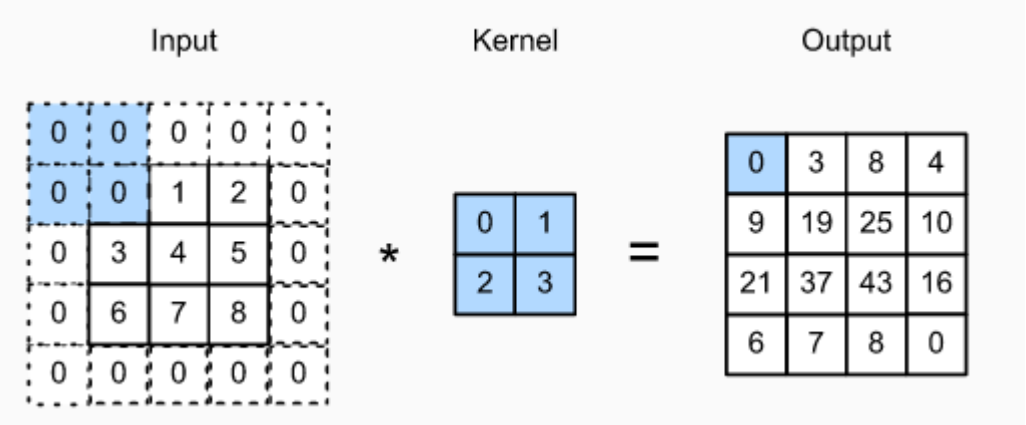
2.What is Stride, Padding & Pooling? Explain with an example.

Padding

There are two problems arises with convolution:

- Every time after convolution operation, original image size getting shrinks, as we have seen in above example six by six down to four by four and in image classification task there are multiple convolution layers so after multiple convolution operation, our original image will really get small but we don't want the image to shrink every time.
- The second issue is that, when kernel moves over original images, it touches the edge of the image less number of times and touches the middle of the image more number of times and it overlaps also in the middle. So, the corner features of any image or on the edges aren't used much in the output.

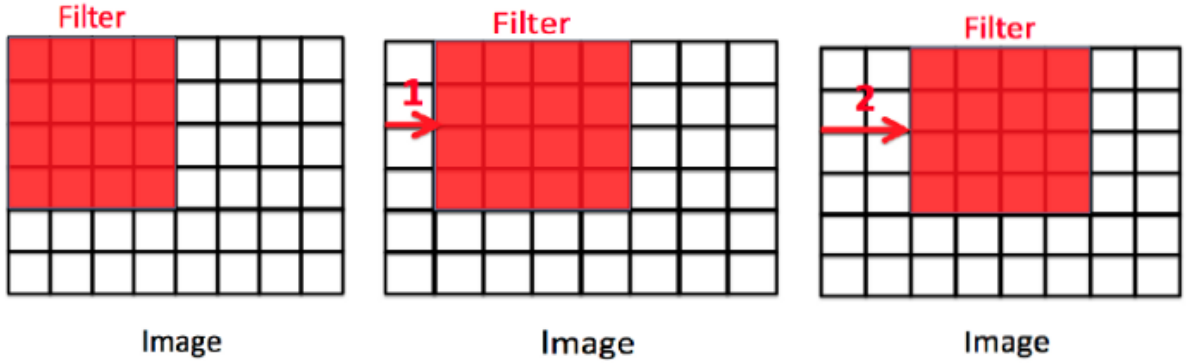
So, in order to solve these two issues, a new concept is introduced called **padding.** Padding preserves the size of the original image.



Padded image convolved with 2*2 kernel

So if a $n*n$ matrix convolved with an f*f matrix the with padding p then the size of the output image will be (n + 2p — f + 1) * (n + 2p — f + 1) where p =1 in this case.

Stride



left image: stride =0, middle image: stride = 1, right image: stride =2

Stride is the number of pixels shifts over the input matrix. For padding p, filter size $f*f$ and input image size $n * n$ and stride 's' our output image dimension will be $[\{(n + 2p - f + 1) / s\} + 1] * [\{(n + 2p - f + 1) / s\} + 1]$.

Pooling

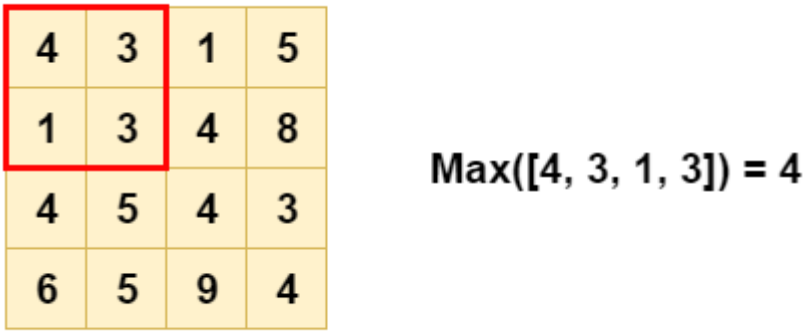
A pooling layer is another building block of a CNN**.** Pooling Its function is to progressively reduce the spatial size of the representation to reduce the network complexity and computational cost.

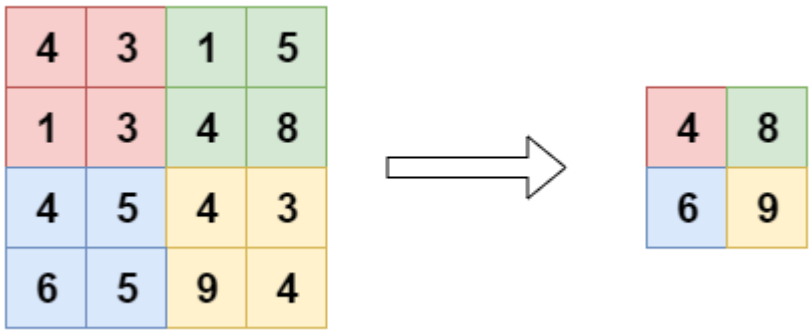
There are two types of widely used pooling in CNN layer:

1. Max Pooling
2. Average Pooling

Max Pooling

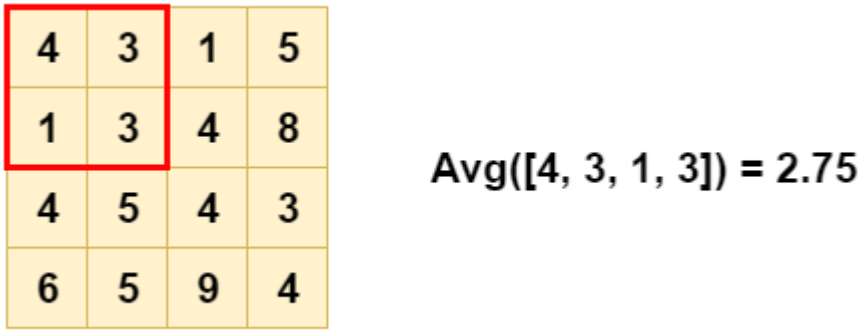
Max pooling is simply a rule to take the maximum of a region and it helps to proceed with the most important features from the image. Max pooling selects the brighter pixels from the image. It is useful when the background of the image is dark and we are interested in only the lighter pixels of the image.

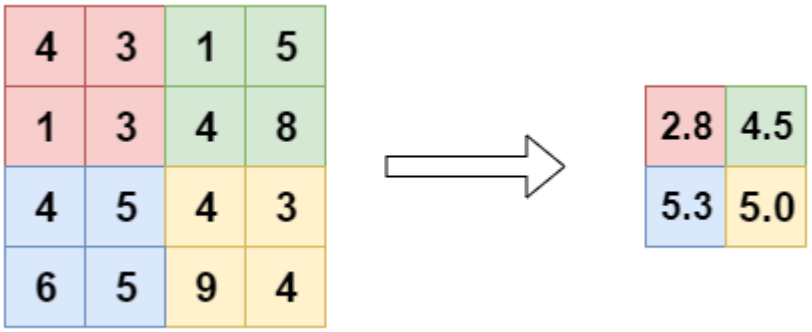




Average Pooling

Average Pooling is different from Max Pooling in the sense that it retains much information about the “less important” elements of a block, or pool. Whereas Max Pooling simply throws them away by picking the maximum value, Average Pooling blends them in. This can be useful in a variety of situations, where such information is useful.





4.What is overfitting? How to overcome overfitting in an ML model?

A statistical model is said to be overfitted when we feed it a lot more data than necessary. When a model fits more data than it actually needs, it starts catching the noisy data and inaccurate values in the data. As a result, the efficiency and accuracy of the model decrease.

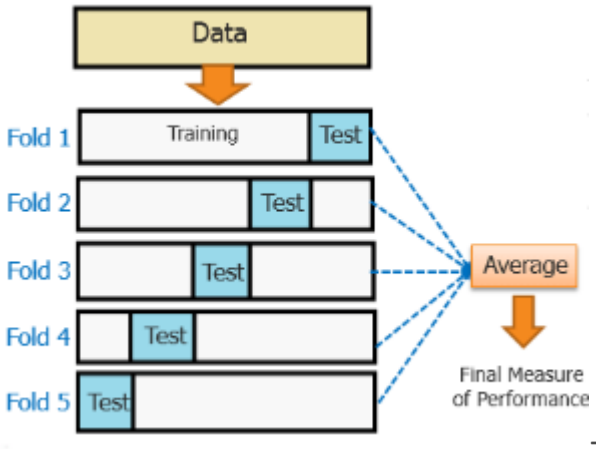
There are several techniques to avoid overfitting in Machine Learning altogether listed below.

1. Cross-Validation
2. Training With More Data
3. Removing Features
4. Early Stopping
5. Regularization
6. Ensembling

1. Cross-Validation

One of the most powerful features to avoid/prevent overfitting is cross-validation. The idea behind this is to use the initial training data to generate mini train-test-splits, and then use these splits to tune your model.

In a standard k-fold validation, the data is partitioned into k-subsets also known as folds. After this, the algorithm is trained iteratively on k-1 folds while using the remaining folds as the test set, also known as holdout fold.



The cross-validation helps us to tune the hyperparameters with only the original training set. It basically keeps the test set separately as a true unseen data set for selecting the final model. Hence, avoiding overfitting altogether.

2. Training with More Data

This technique might not work every time, as we have also discussed in the example above, where training with a significant amount of population helps the model. It basically helps the model in identifying the signal better.

But in some cases, the increased data can also mean feeding more noise to the model. When we are training the model with more data, we have to make sure the data is clean and free from randomness and inconsistencies.

3. Removing Features

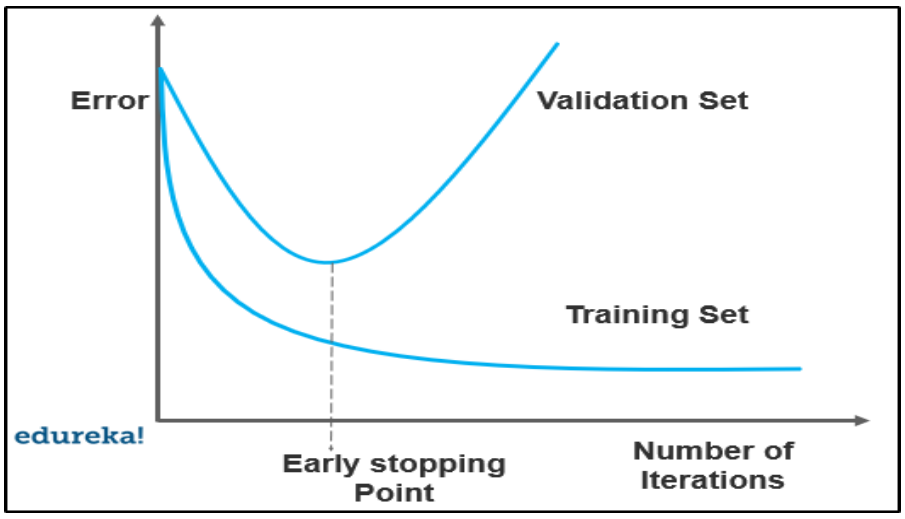
Although some algorithms have an automatic selection of features. For a significant number of those who do not have a built-in feature selection, we can manually remove a few irrelevant features from the input features to improve the generalization.

One way to do it is by deriving a conclusion as to how a feature fits into the model. It is quite similar to debugging the code line-by-line.

In case if a feature is unable to explain the relevancy in the model, we can simply identify those features. We can even use a few feature selection heuristics for a good starting point.

4. Early Stopping

When the model is training, you can actually measure how well the model performs based on each iteration. We can do this until a point when the iterations improve the model's performance. After this, the model overfits the training data as the generalization weakens after each iteration.



So basically, early stopping means stopping the training process before the model passes the point where the model begins to overfit the training data. This technique is mostly used in deep learning.

5. Regularization

It basically means, artificially forcing your model to be simpler by using a broader range of techniques. It totally depends on the type of learner that we are using. For example, we can prune a decision tree, use a dropout on a neural network or add a penalty parameter to the cost function in regression.

Quite often, regularization is a hyperparameter as well. It means it can also be tuned through cross-validation.

6. Ensembling

This technique basically combines predictions from different Machine Learning models. Two of the most common methods for ensembling are listed below:

- Bagging attempts to reduce the chance overfitting the models
- Boosting attempts to improve the predictive flexibility of simpler models

Even though they are both ensemble methods, the approach totally starts from opposite directions. Bagging uses complex base models and tries to smooth out their predictions while boosting uses simple base models and tries to boost its aggregate complexity.