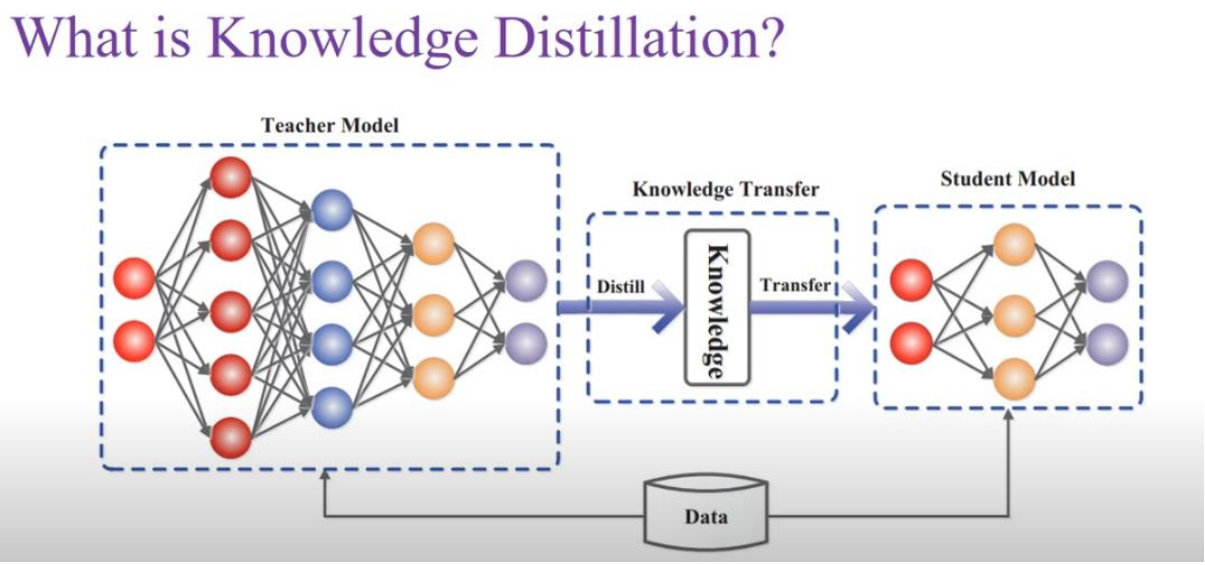
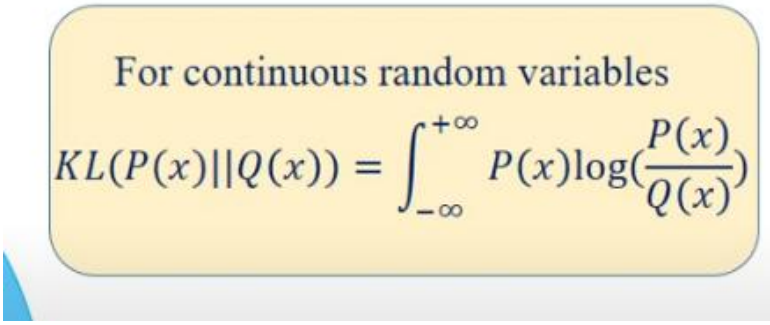
Knowledge Distillation



We have a complex model which is large and has many parameters, we call this the teacher model. Our goal is to replicate the performance of the teacher model to the student model which is less complex and has less parameters as compared to the teacher model.

This process is achieved via knowledge distillation where the teacher model teaches the student model to produce better performance on the same dataset.

Since there are 2 models, we use the concept of KL divergence, since it tells us how the probability distribution of 2 models are similar to each other



Student is forced to mimic the behaviour of teacher by minimizing the cross entropy loss function and by KL divergence.

Nueral networks produce class probabilities using “softmax” output layer but while doing so, it makes the higher probability class larger and pushes the lower probability class values relatively small.

Softmax function tends to hide the relative similarity between other classes. Hence we introduce the concept of temperature, where we make the values of the logits lower before passing to the function.

$$q_i = \frac{exp(z_i/T)}{\sum_j exp(z_j/T)}$$

Applying on MNIST dataset, we get the following performance

```
Teacher Model Parameters: 2395210
Student Model Parameters: 636010
Teacher Accuracy: 97.25%, Parameters: 2395210
Student Accuracy: 97.09%, Parameters: 636010
```

Extra Points

model.eval()

Dropout wouldn’t be used.

Batch Normalization would be switched off and the model would only apply mean and std learnt during training.But it has nothing to do with gradients.