Deep Learning

Prof. Prabir Kumar Biswas
Department of Electronics and Electrical Communication Engineering
Indian Institute of Technology, Kharagpur

Lecture - 16
Optimization

(Refer Slide Time: 00:24)

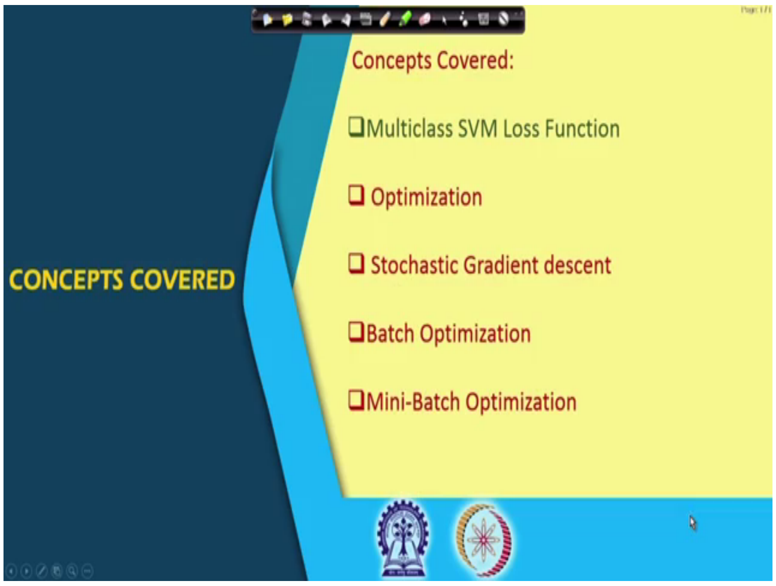
NPTEL ONLINE CERTIFICATION COURSES

Course Name: Deep Learning
Faculty Name: Prof. P. K. Biswas
Department : E & ECE, IIT Kharagpur

Topic
Lecture 16: Optimization

Hello welcome to the NPTEL online certification course on Deep Learning. In our previous class we have talked about the loss function in a multi class support vector machine which is a linear machine.

(Refer Slide Time: 00:43)



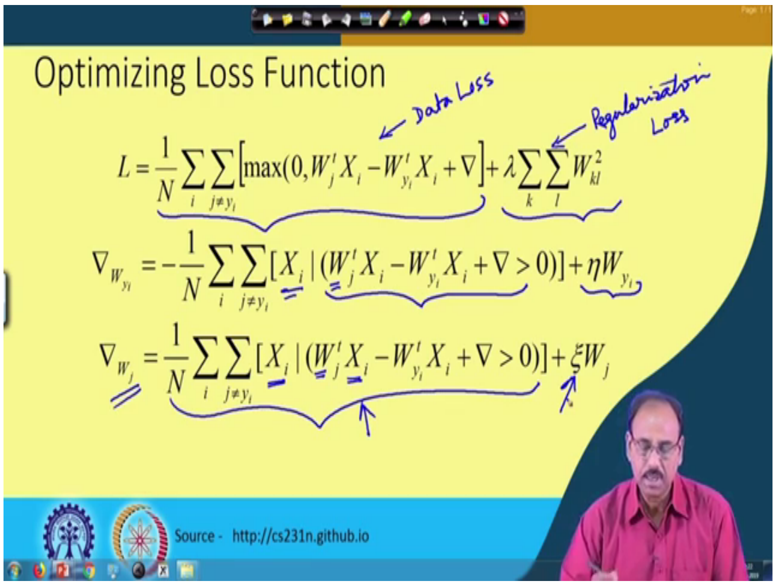
CONCEPTS COVERED

Concepts Covered:

- ❑ Multiclass SVM Loss Function
- ❑ Optimization
- ❑ Stochastic Gradient descent
- ❑ Batch Optimization
- ❑ Mini-Batch Optimization

And we have also seen that how the loss function can be minimized using the gradient descent approach. So, that we get the weight matrix corresponding to the minimum loss on the training vectors. In today's lecture we will continue with the Optimization techniques and we will talk about 3 variants of the optimization techniques which are stochastic gradient descent approach, a batch optimization technique and a mini batch optimization techniques.

(Refer Slide Time: 01:21)



Optimizing Loss Function

Data Loss

Regularization Loss

$$L = \underbrace{\frac{1}{N}\sum_i \sum_{j \neq y_i}\left[\max(0, W_j^t X_i - W_{y_i}^t X_i + \nabla\right]}_{} + \underbrace{\lambda \sum_k \sum_l W_{kl}^2}_{}$$

$$\nabla_{W_{y_i}} = -\frac{1}{N}\sum_i \sum_{j \neq y_i}[X_i \,|\, \underbrace{(W_j^t X_i - W_{y_i}^t X_i + \nabla > 0)}_{}] + \underbrace{\eta W_{y_i}}_{}$$

$$\nabla_{W_j} = \underbrace{\frac{1}{N}\sum_i \sum_{j \neq y_i}[X_i \,|\, (W_j^t X_i - W_{y_i}^t X_i + \nabla > 0)]}_{\uparrow} + \xi W_j$$

Source - http://cs231n.github.io

So, just to recapitulate what we have done in the previous lecture, we had defined a loss function for a multi class support vector machine which is given by L is equal to 1 upon N. Then summation of maximum of 0 and W j transpose X i minus W y transpose X i plus delta plus lambda times sum of W kl square, where W was our weight matrix. And the first term in this loss function which is 1 upon N of sum of this, this is what constitutes the data loss component.

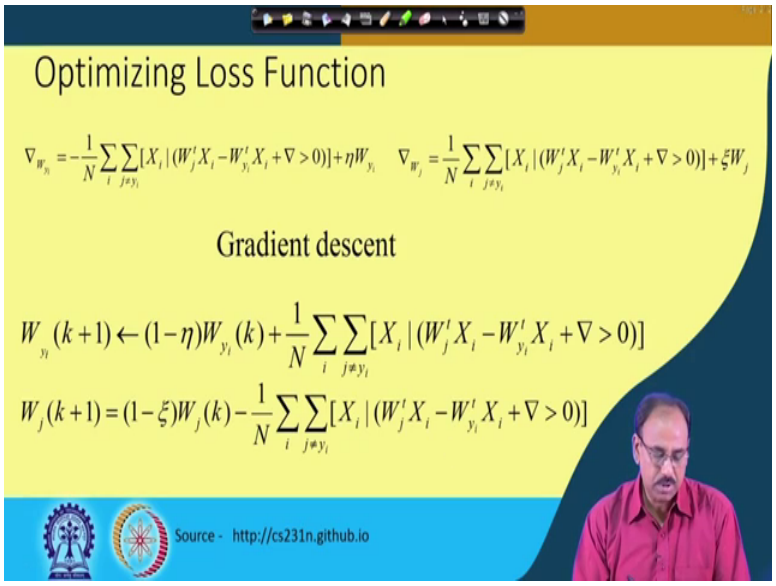
So, it is data loss component and the second term lambda times sum of W kl squared this is what we said is a regularization loss. So, in order to minimize this loss using the gradient descent approach we have to take the gradient of this loss function with respect to W y i and the gradient of this loss function with W j. So, the gradient with respect to W y i is given by this it is minus 1 over N sum of all those X i all those training vectors for which W j transpose X i minus W y transpose X i plus delta is greater than 0.

So, you find that for those training samples for which this component this term W j transpose y i minus W y i transpose W j transpose X i minus W y i transpose X i plus delta is greater than 0 such X i, such training samples are not correctly classified or not satisfactorily classified by the weight vector W. So, for these samples we have to correct the weight vector W. So, that is the reason you take the sum of X i only for those samples which are not correctly classified and this is the correction term that comes from the data loss component.

Similarly, the other term that comes from that regularization loss component which is eta times W y i. In the same manner when you take the gradient with respect to W j again I have a component 1 our 1 over N sum of X i where W j transpose X i minus W y i transpose X i plus delta is equal to is greater than 0 and such X i is; obviously, not satisfactorily classified by that W that I have at that iteration step. So, you take the summation of all those X i which are not satisfactorily classified, I do not take any corrective measure for attaining sample X i which is satisfactorily classified by W.

So, this is a term again coming from the data loss component and the other term that comes from the regularization component is zeta times W j.

(Refer Slide Time: 05:04)



Optimizing Loss Function

$$\nabla_{W_{y_i}} = -\frac{1}{N}\sum_i \sum_{j \neq y_i} [X_i \mid (W_j' X_i - W_{y_i}' X_i + \nabla > 0)] + \eta W_{y_i} \qquad \nabla_{W_j} = \frac{1}{N}\sum_i \sum_{j \neq y_i} [X_i \mid (W_j' X_i - W_{y_i}' X_i + \nabla > 0)] + \xi W_j$$

Gradient descent

$$W_{y_i}(k+1) \leftarrow (1-\eta) W_{y_i}(k) + \frac{1}{N}\sum_i \sum_{j \neq y_i} [X_i \mid (W_j' X_i - W_{y_i}' X_i + \nabla > 0)]$$

$$W_j(k+1) = (1-\xi) W_j(k) - \frac{1}{N}\sum_i \sum_{j \neq y_i} [X_i \mid (W_j' X_i - W_{y_i}' X_i + \nabla > 0)]$$

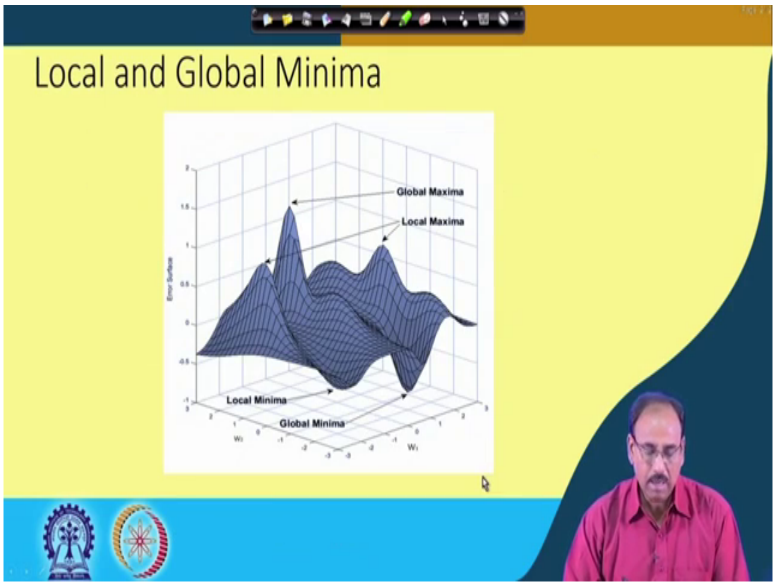
Source - http://cs231n.github.io

So, using this gradient you go for gradient descent approach and your weight updation rules becomes W y i k plus 1 gets 1 minus eta times W y i k plus 1 over N sum of all those X i which were not satisfactorily classified. Similarly W j is modified as W j k plus 1 is equal to 1 minus zeta times W j k minus 1 over N sum of all those X i which are not satisfactory classified. So, that is what gives you the gradient descent approach for updation of the weight matrix and once you reach the minimum of the loss or on convergence the W matrix that you get that gives you the support vector machine or the linear machine which now can be used for classification.

And, again you remember that all this X i which have being used for training the support vector machine or for training the linear machine. Once the training is complete that is once we get and weight which satisfies our criteria of correct classification then we can forget about all those training vectors or X i's which were used for training the support vector machine or which you are used for obtaining the value of weight matrix W which is now to be used for classification.

So, once this part of learning is complete we can simply discard all those training vectors and what I need is only this weight matrix W for classification of any unknown samples or any unforce N samples right.

(Refer Slide Time: 06:58)



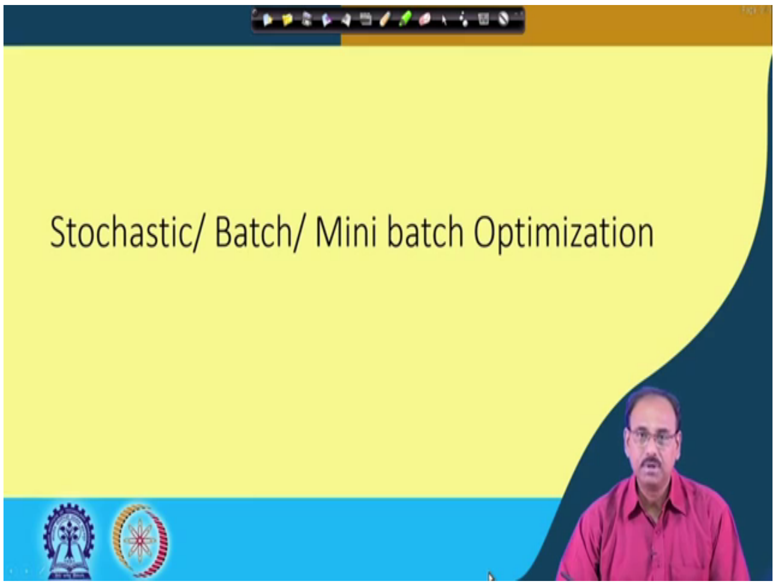
Local and Global Minima

So, coming to the nature of the optimization functions or the loss functions, earlier in case of support vector machine we have seen that the loss function is convex which is always we want that the loss function to be convex. Because then I do not have the problem of local minima local minima or global minima, but in general we will see later that we can have situations where the loss function is not necessarily convex. So, I can have a loss function which is given over here the nature of the loss function is somewhere over here.

So, here you find that this loss function has a number of maxima's and it also has a number of minima's right. So, a minima which is minima overall the minima's that is what is the global minima, similarly a maxima which is maximum over all the maximums that is the global maximum. So, we have local minima and global minimum and we have local maxima and global maxima. So, the target of any optimization technique is to reach the global minima; however, there is a possibility that you may be trapped into local minima.

So, in case of machine learning in machine learning applications even a solution leading to a local minima is acceptable if the loss or the error given by that is not too much. However, as we said that it is possible that you may be trapped into local minima and obtaining a solution corresponding to a global minima it may not be possible always.

(Refer Slide Time: 08:51)



Stochastic/ Batch/ Mini batch Optimization

So, this is in general is what is the loss function and as I said that we always target to have to achieve that global minima. Now, I talk about 3 different optimization variants, one is the stochastic optimization, the one is batch optimization, the other one is mini batch optimization. So, if you look at the previous optimization techniques is somewhere over here. So, you find that whenever I compute the loss function or let me come to the definition of the loss function yeah.

So, whenever I come to or try to define the loss function the loss function is defined in terms of all those samples all those training samples which are not correctly classified or which are not satisfactorily classified by your classifier or by the support vector machine. So, I have to collect all those samples which are not satisfactory satisfied satisfactorily classified; that means, for my training algorithm to work I must have access to all the training samples in one go right.

So, I must have access to all the training samples and then you have to try in the classification algorithm with all the training samples we have to identify all the training samples which are not correctly classified. And, then we have to compute the loss function for each of those individual training samples which are not correctly classified and then you have to combine all such loss functions to get the overall loss function and that is what this expression tells you.

So, wherever this term is greater than 0 that is w transpose W j transpose X i minus W y i transpose X i plus delta, if this term is greater than 0 then I assume that this corresponding X i is not correctly classified and this leads to an error term. So, I have to compute this error term all those training samples sum them up and take they take the average to compute the overall loss function. And, that is in some cases problematic if your training sample size is very very large and this is the approach which is known as batch optimization technique because you are considering all the training samples together.

On the other side the so, one of the problem of batch optimization technique is that I must have access to all the training samples, I have to identify all the training samples which are not correctly classified and then I had to compute the loss function which is the sum of loss functions of all individual training samples the loss function contribution of all individual training samples which are not correctly classified. So, that needs huge amount of memory and sometimes the computation is also not inefficient so, the other variant of that is stochastic optimization techniques.

So, in case of stochastic optimization instead of considering all the training samples together you take the training samples one by one and the moment you find that that training sample is not correctly classified immediately if you compute the loss function corresponding to that training samples and use that loss function to update your weight vector or weight matrix immediately. If the training sample is correctly classified at by a given w then as the training sample is correctly classified so far as that training sample concerned is concerned I need not update W at that moment.

So, you take the training samples one by one, every time our training sample is correctly classified simply skip that sample go to the next sample. If the sample is not correctly classified immediately you update the weight vector or the weight matrix and then you take the next training sample which has to be tied with this updated weight matrix. So, if the next training sample is correctly classified by this updated weight matrix then you skip that sample go to the next one, if the next one is again correctly classified skip that sample again go to the next one, if that sample is not correctly classified. Then immediately using the corresponding loss function following the gradient descent approach you update your weight vector or the weight matrix.

And, it continues like this until you come to a situation that your error is less than certain acceptable limit or in a single pass over all the training samples all the training samples are correctly classified and this is what is known as stochastic optimization or stochastic gradient descent. When you come to batch optimization or batch gradient descent it is somewhere in between a sorry the mini batch optimization or mini batch gradient descent it is somewhere in between.

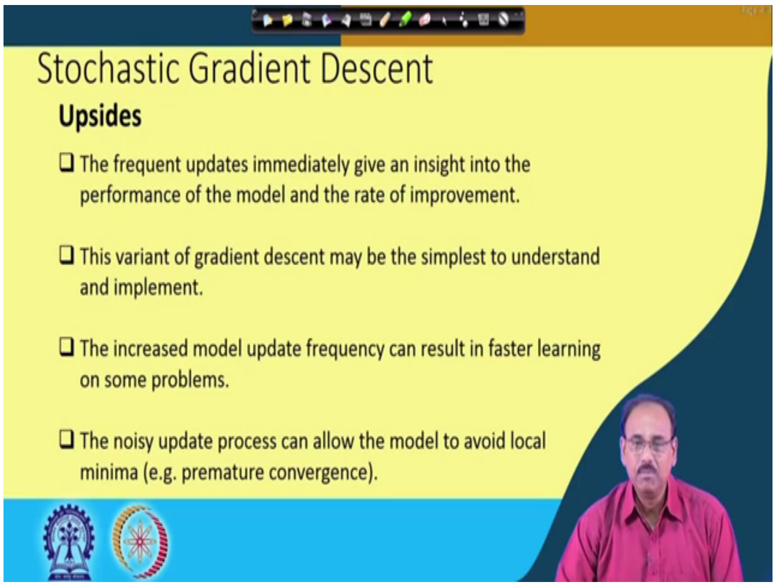
So, in case of batch optimization we have considered all the training samples together and then identified all the samples which were not correctly classified, computed the error function, then accumulated all those error functions together to give you the overall error function and then you go for optimization technique to reduce the error. In case of stochastic optimization you have considered the training samples one by one so that is one other extreme, in one extreme in case of batch optimization you consider all the training samples in one go together, in case of stochastic optimization which is the other extreme you are considering one training sample at a time. If it is correctly classified you skip, if it is not correctly classified you update the weight vector.

In case of mini batch optimization you have to take the training samples in mini batches. So, there have to decide that what should be my batch size whether I should consider 10 training samples at a time or I should consider 100 training samples at a time which is my mini batch size. So, once I have these mini batches then the approach that I have taken for batch optimization I do the same approach for this mini batch. So, if a mini batch contains say 50 training samples I have to consider all those 50 training samples together identify that out of this 50 what are the training samples which are incorrectly classified.

So, you compute the loss function using those incorrectly classified training samples out of your mini batch and using that loss function you go for updation of the weight vector or the weight matrix. So, in case of mini batch optimization I still take the samples in batches, but not all the samples together I divide the samples the training samples into smaller batch size and I go for optimization using those smaller batch sizes that is what is mini batch optimization.

So, the mini batch optimization is somewhere in between your batch optimization and stochastic optimization. So, given this now let us try to see that what are certain advantages or disadvantages of these different approaches of optimization techniques.

(Refer Slide Time: 16:52)



Stochastic Gradient Descent

Upsides

- ❑ The frequent updates immediately give an insight into the performance of the model and the rate of improvement.
- ❑ This variant of gradient descent may be the simplest to understand and implement.
- ❑ The increased model update frequency can result in faster learning on some problems.
- ❑ The noisy update process can allow the model to avoid local minima (e.g. premature convergence).

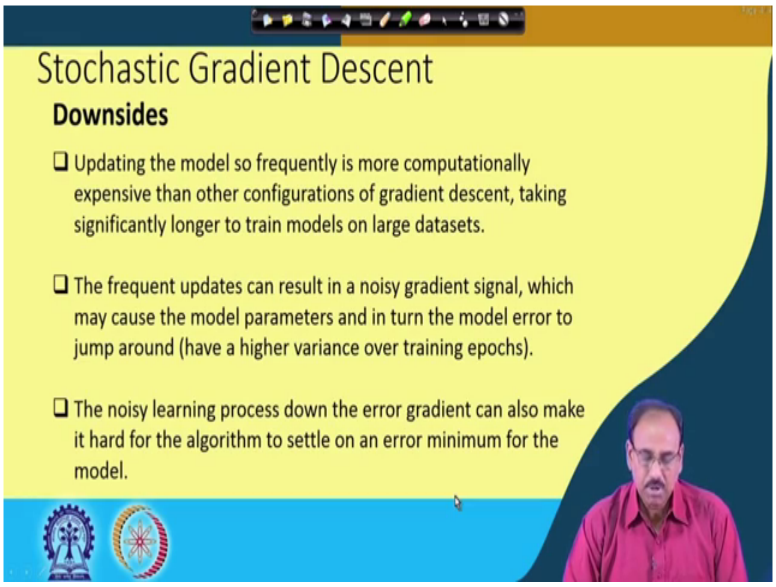
So, coming to you are stochastic gradient descent approach the advantages are you are frequently updating the weight vector or the weight matrix. So, as I said that any moment as the a sample vector is misclassified immediately you go for updation of the weights or updation of the weight matrix. And, this gives you an insight into the performance of the model, how the model is performing and how the improvement in the loss function is taking place.

So, that tells you an insight that gives you an idea of the rate of learning of the model. And this gradient descent is the simplest one to understand also the simplest one to implement, because I do not have to accumulate all the training vectors together or I do not have to identify all the misclassified samples together and combine them collect them and then compute the lost function.

So, the implementation and understanding of this stochastic gradient descent approach is also simpler than in case of batch gradient descent approach. As we are updating the weight vectors quite frequently every time a sample is misclassified you are updating the weight vector or the weight matrix. So, as a result you are learning or updating of the weight vector or learning of the algorithm that may become faster, but this is not always guaranteed, but in some cases it may become faster.

At the same time since you are update process is noisy in the sense that with one training example which is misclassified I can be on one side of the optimization surface the error surface with the other one I may simply go to the other side of the error surface over while doing that you may simply overstep the global minima or local minima. So, as a result it may be possible that you may be able to avoid the local minima and convergence maybe at in the global minima. These are the advantages of the stochastic gradient descent approach; however, it has got disadvantages also.

(Refer Slide Time: 19:50)



Stochastic Gradient Descent

Downsides

- ❑ Updating the model so frequently is more computationally expensive than other configurations of gradient descent, taking significantly longer to train models on large datasets.
- ❑ The frequent updates can result in a noisy gradient signal, which may cause the model parameters and in turn the model error to jump around (have a higher variance over training epochs).
- ❑ The noisy learning process down the error gradient can also make it hard for the algorithm to settle on an error minimum for the model.

Say for example, as the model is updated quite frequently so, your computation may be more expensive than what you do in case of batch optimization. See here this optimization techniques it involves a number of steps for optimization. The first step is for that training samples I have to compute whether it is correctly classified or it is misclassified, if it is misclassified then I have to compute the loss function. Once I compute the loss function then I have to compute the gradient and once I compute the gradient I have to update the weight vector.

So, the complexity depends upon what is the dimensionality of the feature vectors or the dimensionality of the weight vectors. It depends upon the number of training samples that I have because for every training sample I have to compute whether that a sample is correctly classified or the training sample is not correctly classified. The complexity also depends upon the number of iterations over which the computation has to be made and updation has to be made. Now one of the problem of the stochastic gradient descent is though we have said that it is a very very simple approach very simple to understand very simple to implement, but one of the problem may be that suppose for a k th sample vector I find it is correctly classified by the weight vector at a certain in a certain iteration.

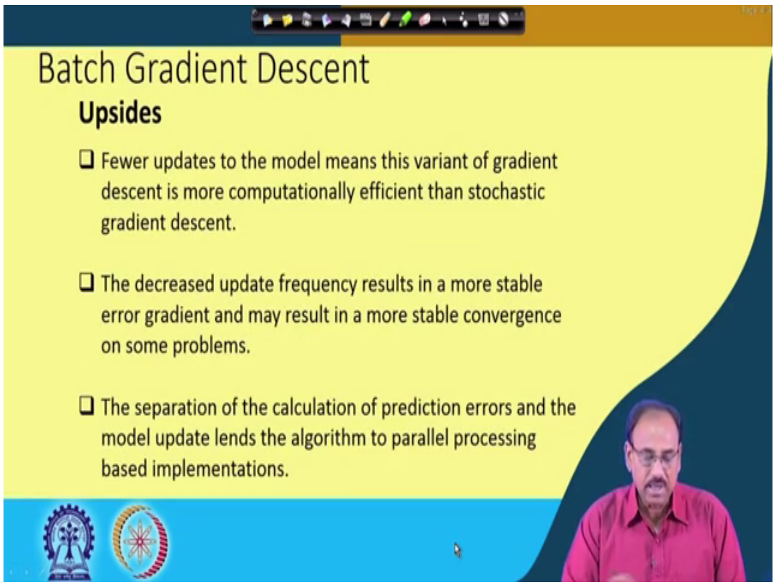
But as the weight vector is being updated for every sample which is misclassified this k th vector which in one instant I have found to be correctly classified by the weight vector. But, in the next pass the same sample may be misclassified by the updated weight vector because the weight vectors are being continuously updated and that may lead to increase in computational complexity.

Not only that once I find an weight vector once I find a sample vector is misclassified by a weight vector in a particular iteration step immediately you are updating that weight vector. And using that updated weight vector you go to you go for testing the next training vector and this continues. And finally, you have to come back to the same weight vector using which you had updated the weight vector once and now you may find that this weight vector again is not correctly classified by the (Refer Time: 22:33) weight vector.

So, the same operation has to be done once more and this may repeat many times and that may lead to the computational complexity or increase of computation time of the gradient descent approach and this may be significant longer to train a model on a very large data set. The other problem may be that frequent update can result in noisy gradient signal and that may cause the model parameters and in turn the model error to jump around and this is what I said that, I may overstep the minima which I want to reach because I am frequently updating the nor model parameters.

The other problem may be that the noisy learning process down the error gradient can also make hard the algorithm to settle on an error minimum for the model it is again for the same one that. Even if certain training vector a particular training vector is correctly classified by on weight vector at a particular iteration in the next iteration the same training vector the sample vector may be classify misclassified by the updated weight vector and that does not allow the weight vectors to settle to a minimum of the error. So, these are the problems of the stochastic gradient descent.

(Refer Slide Time: 24:09)



Batch Gradient Descent

Upsides

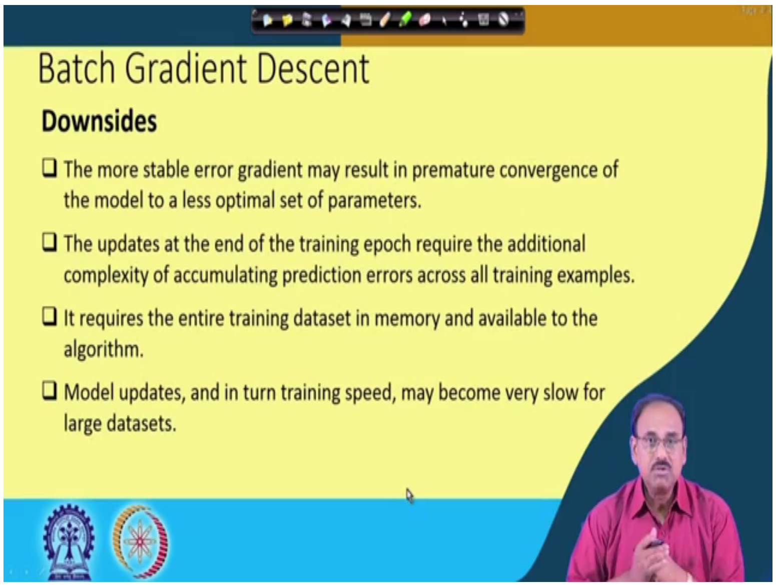
- Fewer updates to the model means this variant of gradient descent is more computationally efficient than stochastic gradient descent.
- The decreased update frequency results in a more stable error gradient and may result in a more stable convergence on some problems.
- The separation of the calculation of prediction errors and the model update lends the algorithm to parallel processing based implementations.

Now, against this when you go for batch gradient descent, the batch gradient descent approach again has got certain advantages, it also has certain disadvantage. The advantages are that as your updates are not that frequent or if a fewer number of updates because here for every sample we are not updating the weight vector rather we are collecting all the samples which are misclassified then computing well expansion and then we are computing the weight vector. So that means, this is usually computationally more efficient than the stochastic gradient descent approach.

And this decreased update frequency results in a more stable error gradient and may result in a more stable convergence on some problems, again you note that term that it is on some problems it is not guaranteed that in every case this may not be true. And the next one is as your calculation of errors and the update this were two separate processes for calculation of error or the loss function, what we are doing is, we are collecting all the samples which are misclassified then you are computing the error. And, once your error computation is complete then you are using that error to update your model parameters.

So, these two processes are separate and as that two processes are separate they can be implemented in parallel as well in a pipeline right. So, these are the advantages of the batch optimization or batch gradient descent approach.

(Refer Slide Time: 26:00)



Batch Gradient Descent

Downsides

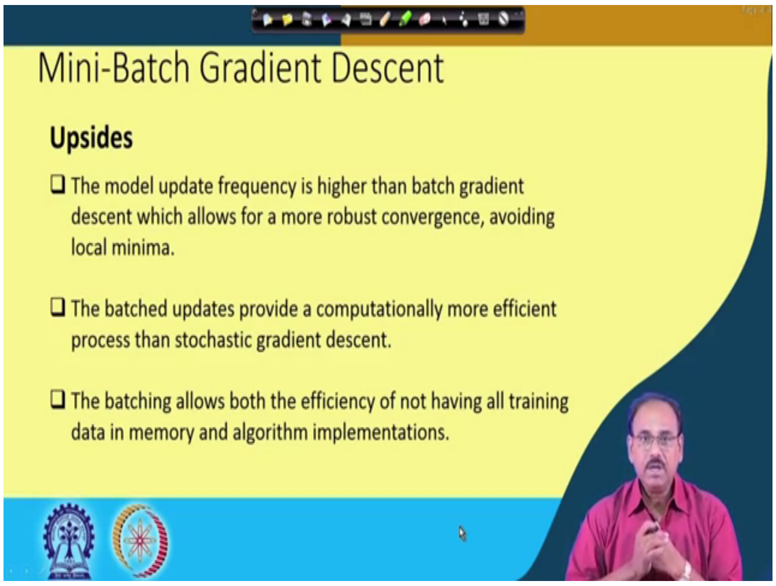
- ❑ The more stable error gradient may result in premature convergence of the model to a less optimal set of parameters.
- ❑ The updates at the end of the training epoch require the additional complexity of accumulating prediction errors across all training examples.
- ❑ It requires the entire training dataset in memory and available to the algorithm.
- ❑ Model updates, and in turn training speed, may become very slow for large datasets.

On the downside or coming to that these advantages that as the error gradient is very stable this may result in a premature convergence in the sense that there is a high risk of converging to a local minima, because of stable gradient. The other disadvantage is that the updates at the end of every training epoch require additional complexity of accumulating prediction errors across all training samples. And, this is what I said that I have to identify all the training samples which are misclassified and using these misclassified training samples I have to get the overall error so, that is also a disadvantages.

It also requires that the entertaining data must be available in memory and to the algorithm for processing, which requires huge amount of memory and as in case of machine learning applications or deep learning applications the number of training samples is use of the order of maybe lakhs of samples are used for training your machine learning algorithm. So, this is a problem that for batch optimization techniques are bad batch gradient descent techniques I require that all the training samples must be available in memory and must be available to your objection algorithm. And of course, as we are working on all the data at a time the model updates and the training speed may become very slow for the datasets we are very very large.

So, these are the advantages and disadvantages of the batch optimization or batch gradient descent approach.

(Refer Slide Time: 27:58)



Mini-Batch Gradient Descent

Upsides

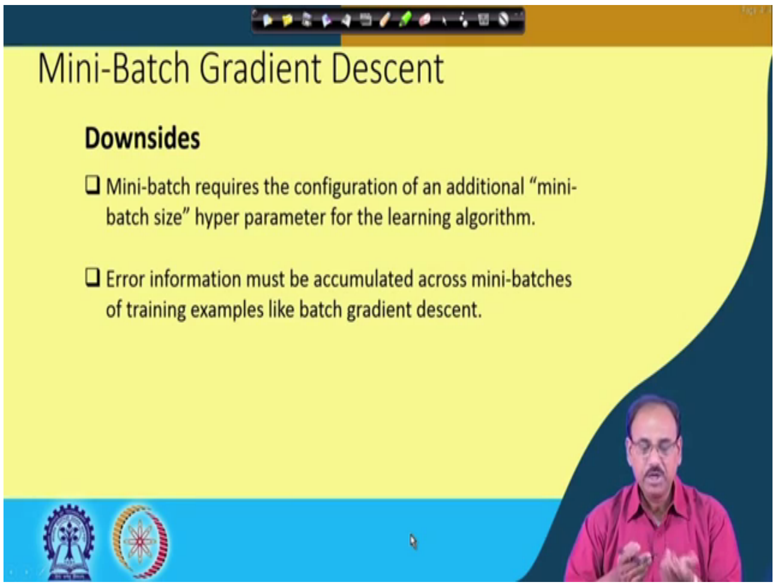
- The model update frequency is higher than batch gradient descent which allows for a more robust convergence, avoiding local minima.
- The batched updates provide a computationally more efficient process than stochastic gradient descent.
- The batching allows both the efficiency of not having all training data in memory and algorithm implementations.

Then coming to the mini batch gradient descent as we said that mini batch is somewhere in between your stochastic optimization and batch optimization. So, the advantages of mini batch optimization is that the model update is frequencies higher than that in case of batch descent optimization which allows for more robust convergence avoiding local minima.

So, in case of batch optimization we have seen that as your error rate as the training rate is very very slow, so is more stable. So, there is a risk of convergence to a local minima in case of batch optimization that reach that risk is less and this batch updates provide more efficient process than in case of stochastic gradient descent because in stochastic gradient descent we had to work every sample at a time in batch optimization we are taking mini batches.

So, a here the computation is more efficient than in case of batch optimization and the other advantage of our batch optimization is that in case of mini batch I do not need all the training samples to be available in memory and the algorithm rather I need all the training samples in the mini batch to be available in the main memory, whose size is much lower than the size of all training samples.

(Refer Slide Time: 29:34)



Mini-Batch Gradient Descent

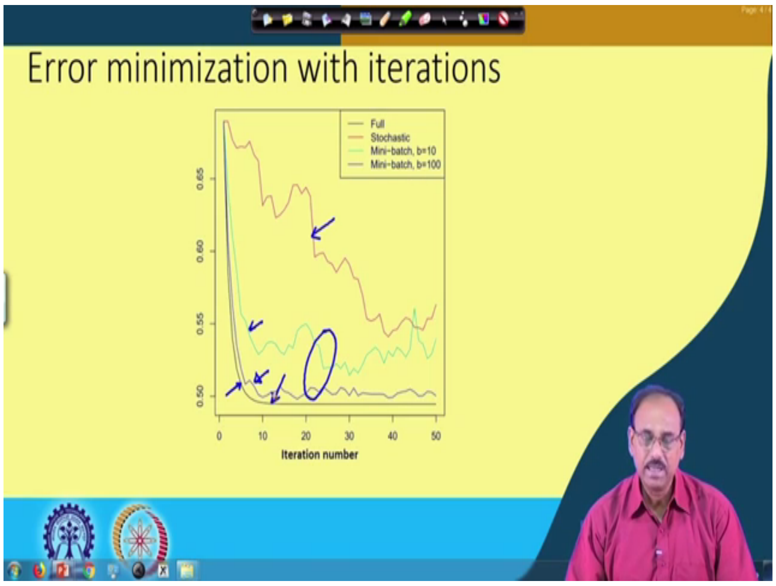
Downsides

- ❑ Mini-batch requires the configuration of an additional "mini-batch size" hyper parameter for the learning algorithm.
- ❑ Error information must be accumulated across mini-batches of training examples like batch gradient descent.

And coming to the disadvantages, here again how do you decide what should be the mini batch size that is mini batch request configuration of an additional hyper parameter which is the size of the mini batch. And, the other advantage disadvantage which is of course, not as severe as we had in case of batch optimization that the error information must be accumulated across batch mini batches of training samples for batch gradient descent.

So, I can have these 3 variants of the optimization techniques or gradient descent techniques, the batch optimization, the stochastic optimization and mini batch optimization.

(Refer Slide Time: 30:19)



Error minimization with iterations

And, just as a comparison of the performance of these 3 you find that this shows a set of curves on certain examples for batch optimization, mini batch optimization and stochastic gradient descent. So, here fine you find that this black curve this is for the batch optimization and here you find that this curve is very smooth, on the other extreme we have this stochastic gradient descent, but every sample is taken one at a time ok.

And, here you find that the error reduction over iteration is very easy (Refer Time: 31:05) and using the batch optimization techniques we have performance in between which is as expected and depending upon the mini batch size your rate of convergence is different. So, this is the one which tells you where mini batch sizes stained and this is the one which tells you when which gives us the performance when mini batch sizes 100 and you find that interestingly that if you in go on increasing the mini batch size finally, it becomes a batch optimization technique.

So, with increase of mini batch size or in other words with increase of number of samples to be considered to together for batch optimization techniques you find that gradually you are moving towards the performance of your batch optimization technique. So, in today's lecture we have discussed about the gradient descent and the 3 different variants of gradient descent, the batch gradient descent, the mini batch gradient descent and the stochastic gradient descent. So, with this I complete this lecture.

Thank you.