Deep Learning for Visual Computing
Prof. Debdoot Sheet
Department of Electrical Engineering
Indian Institute of Technology, Kharagpur

Lecture – 43
Domain Adaptation and Transfer Learning in Deep Neural Networks

Welcome, so today we are going to do an interesting topic. So, you have been learning a lot about Deep Neural Networks, very deep neural networks and like real recent variants. So, we have covered down model switch were as recent as event 2016 and 17 coming down from the recent C. V. Pierce.

Now, one thing which comes into mind is that, all of these models which we were learning down; there were champions of solving a very pertinent problem which is called as the ImageNet challenge, great. So, you had the standard-sized images of 2 to 4 cross 2 to 4, and there were 1000 categories of objects which were present in these images and the whole objective over there was to train down object train down the network which can actually very effectively with a very high accuracy classify down these multiple categories of objects;. So, that was going on pretty good.

Now, the challenge which comes out is that, you have a network which has been designed, it has been trained, you have your weights available; but then till now we have not made use of any of these weights. So, when we were downloading all the models we were downloading just bear models, which are from your model zoo or from your torch vision models library. You are downloading only the architecture; we never downloaded these weights over there directly, it was just the architecture definition which was coming down.

On the other side, we did look into the weight space complexity computations and what is the total time complexity computation as well. And one thing which you would remember is that, if you are looking down into weight space over there and we try to download a model with the weights, then the total download size is also larger. So, that was because the weights have to be downloaded and kept down onto your space and the moment you do it your ram also explodes out quite on a good way.

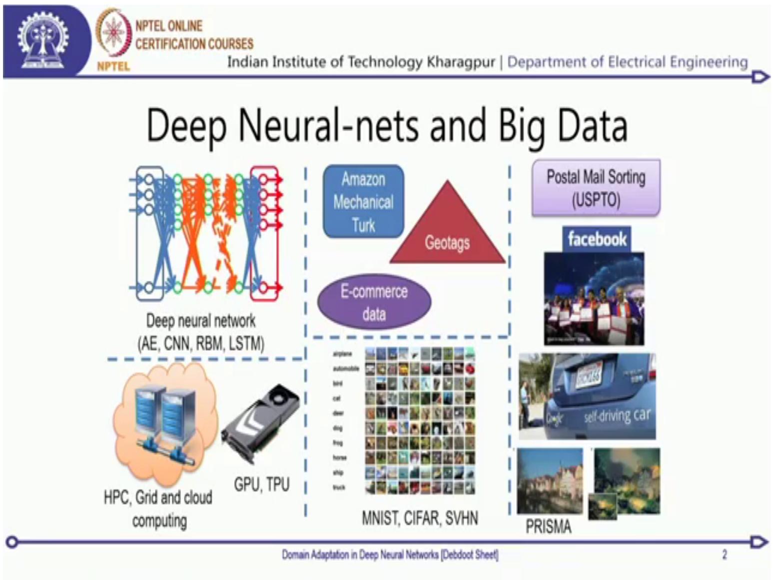
Now, what evidently comes to your mind is that say, we are trying to solve the same kind of an ImageNet problem or real object detection problems. When we were doing it done with smaller size images from c fired just bloating it out, 2 to 4 cross 2 to.4 We realized that the accuracy which we are going up to was not that what is compared to a image net problem; we are still stuck down at about belly 80 percent. Yes, we are doing with lesser number of sample images; we are also doing it for lesser number of classes.

But then is there a way that I can import all the weights which are present down over there for a model, which has been trained for ImageNet problem which were still natural images. And was trained for possibly more number of classes; so, the granularity of training was really high. So, can I actually bring it up and then train a model for lesser number of classes and can I make it faster.

So, that is the whole purpose of something which is called as transfer learning. So, you have a model which was trained to solve one task in one domain and then I am bringing it down to solve another task in another domain and under what conditions can I actually do that.

So, this is what we will be doing in today's lecture which is called as Domain Adaptation and then domain adaptation in deep neural networks is very specifically. So, the whole purpose of transfer learning is to do it via domain adaptation that is what I am going to do. While this itself is a very big field, but we are going to touch upon smaller aspects over there as far as what we do from a very practical point of view.

(Refer Slide Time: 03:16)



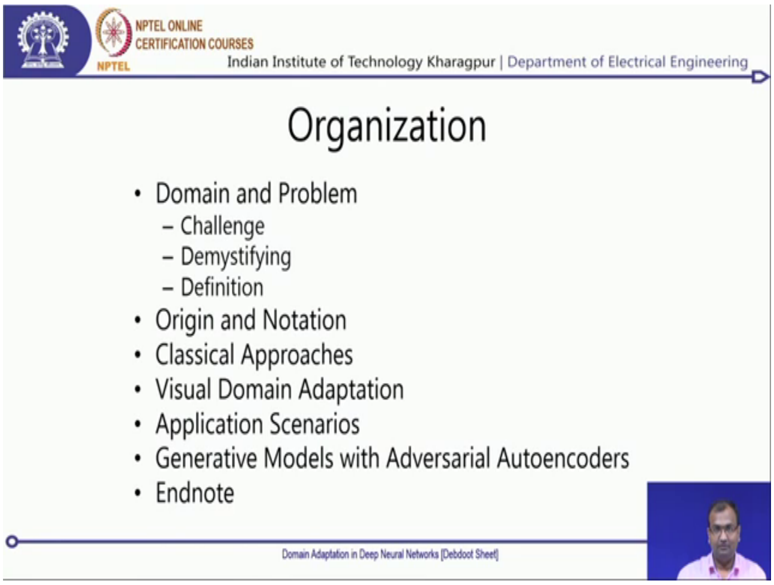
So, if you look into what the situation today is that, you have a real big extensive library of these deep neural networks available. You have your different sources of data, so, they come down from Geotags from Amazon Mechanical Turk and that is how ImageNet itself was created.

You also have your E-commerce data available and coming down; and then on the other side of it you have your real great compute systems. You have your GPU's, you have your Grids and HPC's and everything present down. And then you have very standard data sets which you could use for creating your networks over here.

Now, once we had all of this we did do a lot of interesting work ; so, like we got down self-driving cars, we got down photo tagging on Facebook identification of objects, we got down PRISMA for generating artistic equivalents of these images. We also got down say these mill Postal Sorting systems as in one of this major examples of Hilly Net a very practical example of Hilly Net.

Now, when we are actually missing at this point of time is; so, these were people who solved very specific problems, they really curated and got down a large amount of data. Can I use the same sort of model with lesser amount of data? So, can I use a model which these guys have trained separately for their purpose using a lot of data, which is available to them; can I use that in order to solve my problem, where I have lesser amount of data available. Now, this is what we are looking at today ok.

(Refer Slide Time: 04:46)

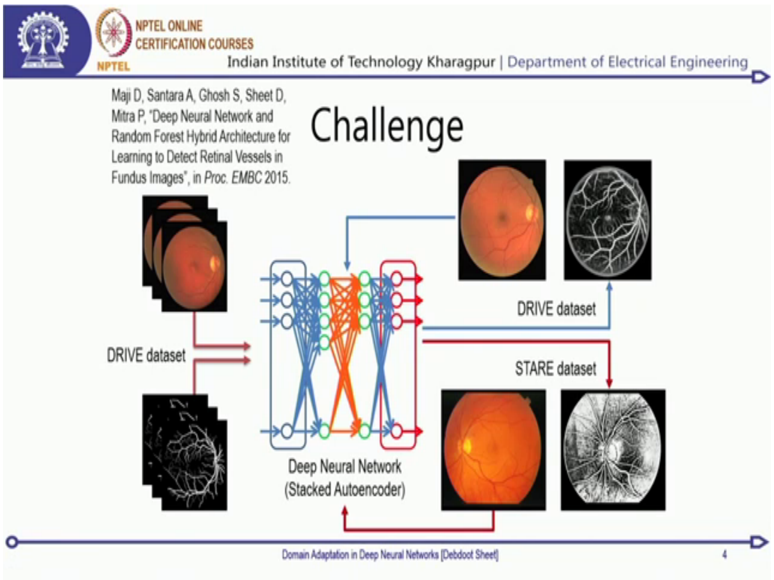


So, let us we look into the organization how it goes down is that you have so, initially I will describe you what a domain is and what our task or a problem is at hand. Then, we get down into certain origin and notations; we go into something called as the classical approaches for domain adaptation, and then get into certain application scenarios and then come down to an endnote.

So, generative models with adversarial autoencoders is which I will be covering down, later on point of time with a very extensive lecture. So, there is one whole series of few lectures which we have in this next week, where we are going to work down on generative models itself and how these neural networks can work as generators.

So, can you give a class label and generate an image out of it. So, that is roughly speaking a very shorter or very sweet and simple definition of a generative model as it goes down ok.

(Refer Slide Time: 05:40)



Now, let us look into the challenge, now I would take out this very specific challenge at hand. And this is a problem which we have already solved out which was the vessel segmentation problem in DRIVE dataset. So, you had your retinal images present over there, and then the whole object was that can eye segment out these vessel using some sort of a deep neural network. And what we had used over there was a fully connected Neural Network in order to do that, ok.

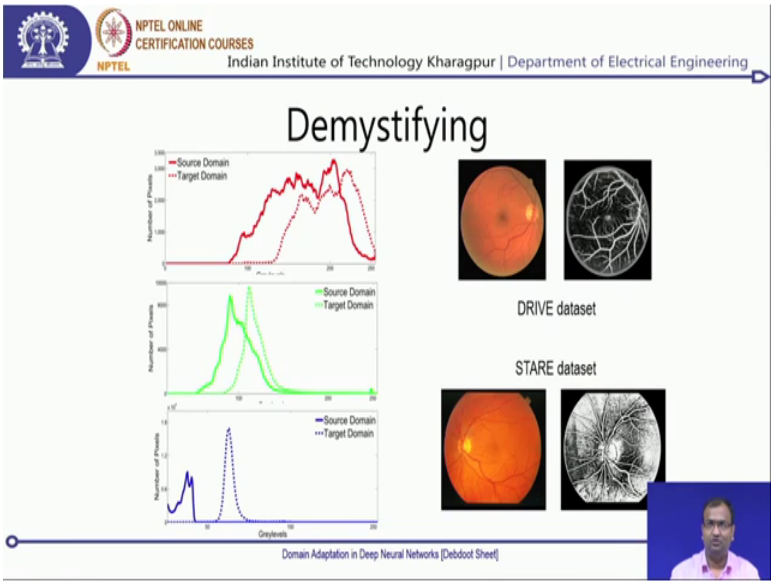
Now, so, if we train the model with that, and then we give an input to it which was from the same data set; we got down a pretty good output coming down. Now, the challenge is that, DRIVE is a big data set and then we use it with trained down a model. Now, say that I get down images from a local hospital which has an Ophthalmoscope, ok. Now, they might not be using the same sort of retinal camera on ophthalmoscope, which has been used for creating drive.

Then these people were from the European origin whose images were taken down, in Indian origin we have a different kind of an image coming up. So, now, how will you do. So, this is a very pertinent product development problem which we have. So, what we trying to do is, these were all healthy images people were healthy they were not supposed to be having any diseases; and that is how they were taken down.

Now, on the other side of it you have this STARE dataset which is from diabetic patients; now, the moment you have this one and you try to feed it through this network this is the kind of an output which it generates. Now, quite clearly you can look into it, that this is a horrible vessel; this is this is really bad. Because, what comes down is that majority of this background region over there, where you do not have any vessel it shows them as white. So, that technically would mean down that all your vessels have ruptured and you have blood flowing down everywhere.

Now, that is not the case; one of the reasons is that that definitely has been a shift in the nature of the data, and because of the shift or the change in the nature of the data is why you are getting down this really bad result. Now, that is this makes it a very pertinent case for domain adaptation for us.

(Refer Slide Time: 07:40)



So, let us look into Demystifying what actually went wrong over there. Now, what came out is that once you have your source domain. So, source domain is this DRIVE dataset over there, and what we are plotting down is the histogram of the pixels which represent these vascular region or the vessels over here. Now, in my DRIVE dataset this solid red line is, what is the histogram of the pixels underneath the blood vessels.

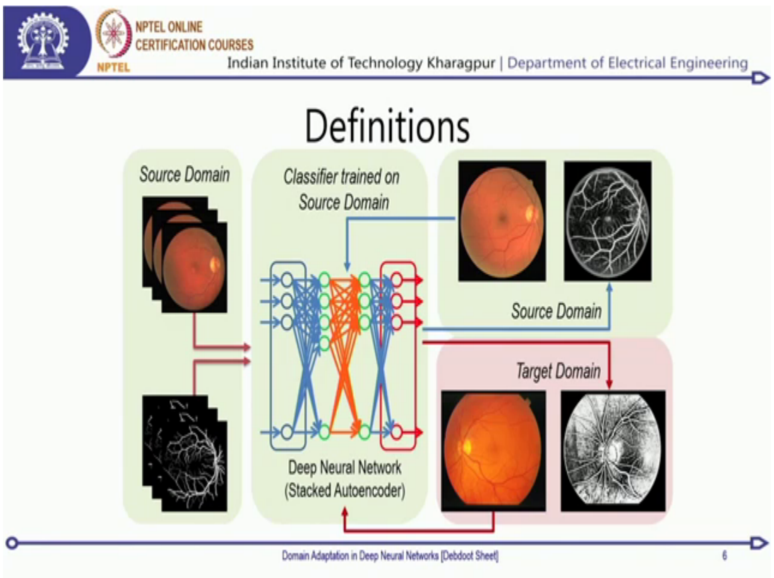
Now, this dotted lines which I have over here is what is the histogram for my target domain, which is for the STARE dataset. So, this is what we call as Target Domain, which is; where we are going to employ our model once it has been trained on this one which is called as the Source Domain from my DRIVE dataset, ok. So, that is the definition of source and target, and it is it is universally accepted unanimously across the field of domain adaptation.

Now, when I look into the histogram of the red channel it is it is more or less overlapping there is some part of a non overlapping area. When I look into the green channel, yes there are major shifts, but not so significant over there. Whereas, when I look into my blue channel, what I would get down is my blue channel is completely offset and drift out. And this is one of the reasons why; when I was working down on the same one.

So, while here it was having lower intensities of blue, here it is having on my target domain higher intensities of blue. And this is one of the reasons why it just drifted out and then went down somewhere over here, in my background regions in order to show that they are the ones.

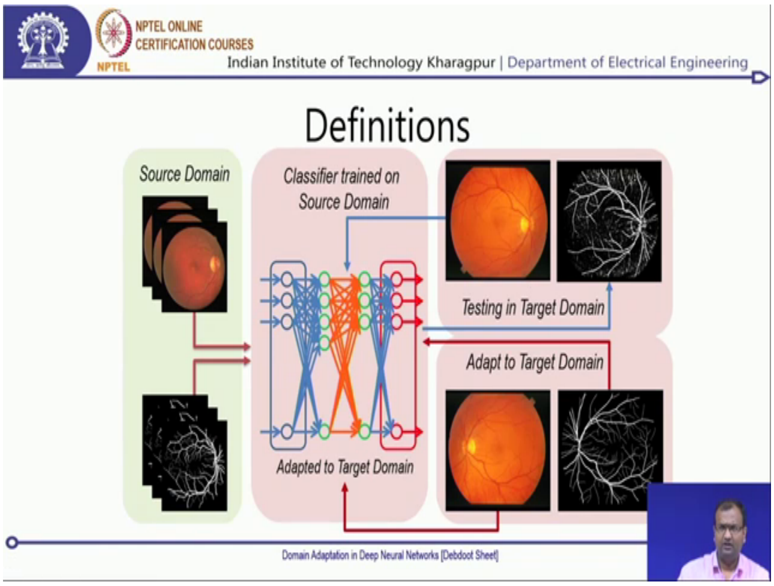
So, this is a very intuitive and simple explanation, but that is not always the case. So, there can be different kind of shifts which attractable, which may not immediately be defined. So, there are multiple ways of actually coming to a solution to this one.

(Refer Slide Time: 09:19)



So, how it gets defined is that the original data on which you are going to train this model or what it has been trained down over here for us it is DRIVE that is the data from that one is what is called as the Source Domain, ok. Now, you have a classifier which you train it down and this classifier is a deep neural network for us and this is what is a classifier trained down the source domain. Now, if I am working on that one, I get down a good result but the moment I get down from a different one which is on my target domain, I get a very bad result coming down.

(Refer Slide Time: 09:54)

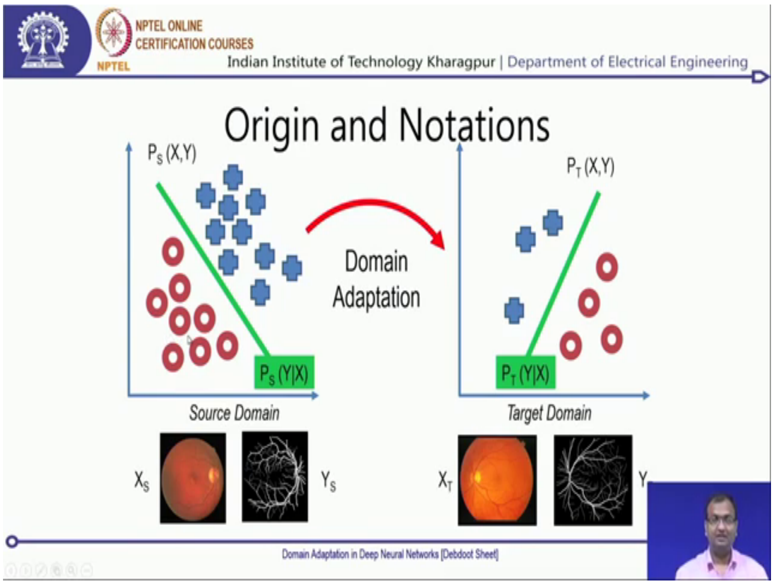


Now, the whole idea for domain adaptation is that you take at least one example, they can be more than one example from the source domain and sorry from the target domain. And feed it to this source domain and try to modify the weights of this source domain.

Now, once you have modified this source domain weights; so, you have weights which are initialized by training it over here, then I am going to retrain. So, do a feedback feed forward and then you find out your error you do a back propagation, feed forward error back propagation; so, these weights are going to update.

Now, you have a model which has been adapted in that way and this is called as a model which has is adopted to the target domain. Now, if you put down example over there you would see that the results are quite intriguingly similar over there, ok. So, this is what comes out ok; well and good.

(Refer Slide Time: 10:39)



Let us look into some of these notations, common notations for this problem. So, the data the input data is typically what is called as X S. So, X has been for my data and Y has been for my class level and the subscript S is just for my source to denote that is from my S source ok.

Now, what I have if I try to look into a joint probability space, then this is what joint probability space will be looking like. So, say I have two different features and then these pluses are what belong to my blood vessel region and the red circles are what belong to my background region over there. So, I have red circles and blue pluses over here.

Now, if that is the case then, together if I plot it down then this is how it would look down if I am just looking at two different features so, this is a hypothetical feature space on which I am looking. Now, any kind of a classifier, what is this; margin which is going to segregate between these two classes, great.

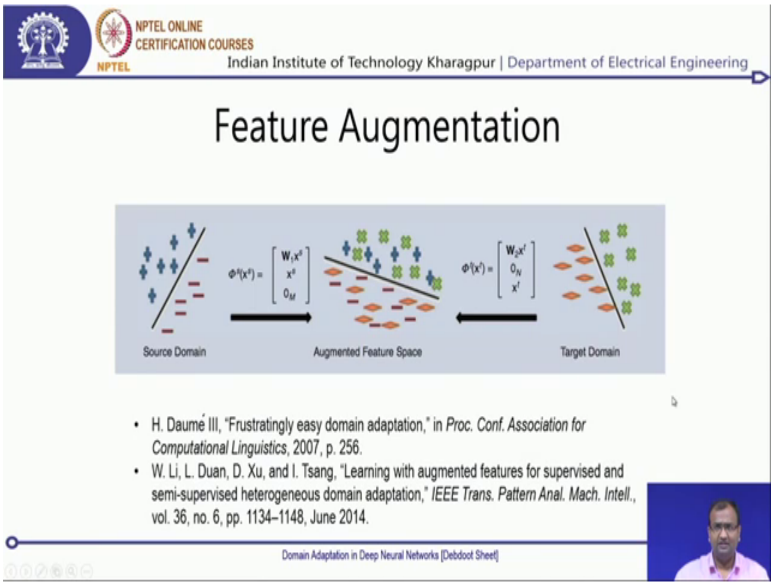
Now, on my target domain when I get it down, what I would be having is a feature space which has a very different topography as such. And obviously, there are less number of pixels available, because you have lesser number of samples available in your target domain.

So, what we said from the definition is that, your target domain may have as less as just one single sample present over there. Now, my whole problem is that using this lesser number of samples I need to draw this sort of a line over here. Now, obviously if I take this classifier line over here and trying to place it over here, so, that will be a line which comes down along this way.

Now, the moment you have that kind of a line coming down, you see that majority of these will be wrongly classified and that is wrong. So, the whole purpose of domain Adaptation is actually to somehow bend this classifier line and bring it over here. So, this is one of the ways in which you are going to modify the classifier.

Obviously, there is another way in which what you can do is, you can modify this data points and do some sort of an affine transformation and place it back onto this domain. So, that is also valid way, but then that is not something which works around with the classifier and we are not going to touch up on that one.

(Refer Slide Time: 12:43)



So, we look into a particular field which is called as Visual Domain Adaptation. Now, this has been there on the field for quite long time and then they are multiple approaches of doing it out. So, one of these ways is what is called as a Feature Augmentation.

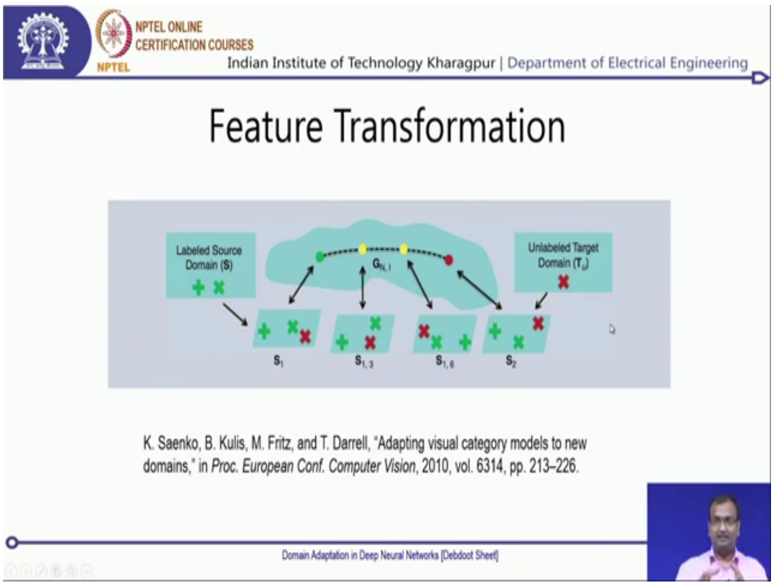
So, the whole idea is that you try to get some sort of a transformation of this input feature space onto a higher dimensional feature space; such that from your source domain to your higher dimensional feature space when you are coming down.

So, you are bringing down your data points as well as your linear classifier or the or the classification margin over there to a higher dimensional space; you do the same thing for your target domain as well. Such that these two classifier margins merge over there in some way and then it makes it easier.

So, you train a classifier with this data over here do this transformation and you get your classifier transformed over here. For a target domain data which comes down over there, you have another transformation which is defined; now, this will get down a different data.

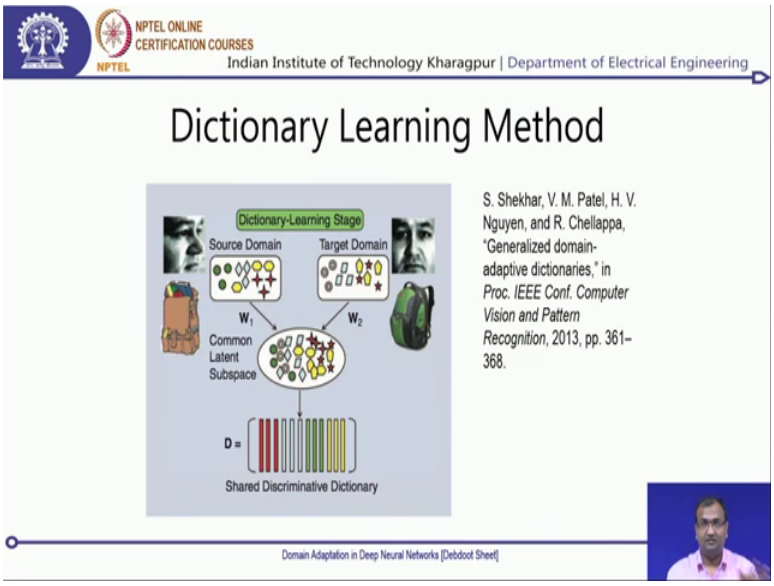
So, this green crosses and orange diamonds which are over here. So, they will fall down in proper way and you have a very high classification. So, for details you can definitely read down through these papers which are present over here. So, this is one of this early and very simple approaches for it.

(Refer Slide Time: 13:47)



The next one is what is called as a Feature Transformation and the whole idea is that, you keeps keep on mixing. Some amount of unlabeled data from your target domain onto your source domain, such that you can have some sort of a gradual transition between these two domains. Now, at a point of time it was a very popular method, but today it is it is no more used except for problems where you have real data scarcity and it makes it complicated to transform whole models and try to learn it out over there.

(Refer Slide Time: 14:16)



Now, one of my personal favorites and what a lot of people actually use today is what is called as a Dictionary Learning Approach. So, the whole idea over here was that say, I had a; let us go down into very simple one. So, say my source domain had side view images of human faces and they had clipart images of bags or some objects over there and my whole purpose was to segregate human faces from bags ok.

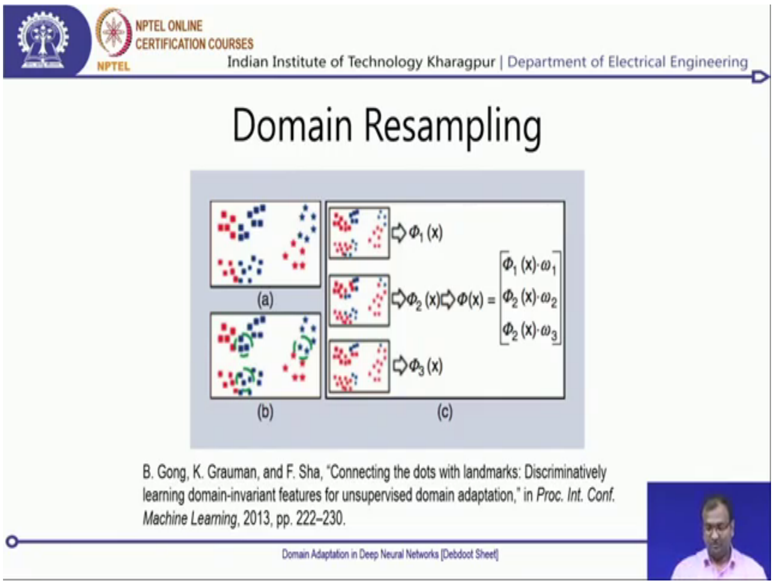
Now, in my target domain what I get is, I get front view images of human faces and I get camera images of backs or from my phone camera I have just taken out and I want to segregate it out.

Now, it is quite rational that for this for this network which was trained on the source domain since the nature of the data was probably different it is it is not. So, easy for a two adapt although we as humans have known to adapt it; but we have years of training in order to adapt it.

Now, what will you do over there? So, what typically is done is that; you need to have the source domain data available to you and the target domain data and you would start creating a common latent dictionary. What that would mean is that, you put all of these side view faces, front view faces and everything. So, you train a network which took only side view faces and this clipart images and this is trained and kept down. You also have your source domain data.

Now, you create a newer class of faces, where your side view faces as well as your front view faces; you create a newer class a set of a newer class over there for bags where you have clipart representation of bags as well as you have camera images of bags. Now, you create this common dictionary common discriminative dictionary and then make it discriminate between these two classes. So, that is what comes down in a dictionary learning kind of a method.

(Refer Slide Time: 15:53)



Now, there is also another method called as Domain Resampling, in which what you do is that if you have multiple features over there then you create subspaces of features. And then try to resample out from these subspaces of features, mix them with some different affine transformations; and come down to a common form of higher dimensional space between your source domain and your target domain.

So, that is one part of it which is about mixing down data from different domain then trying to update your model based on it. Now, we will get down into few application scenarios of where it comes down.

(Refer Slide Time: 16:25)

So, one of it is in Face Recognition say that I have faces where, where the challenge is that I can have faces under different poses. So, there can be side view, there can be front view; can I recognize really these ones. Then, I can have people with headgear with uncontrolled background and all of that. I can have say one of these views of my face given a straight view, but then under different emotional expressions over there.

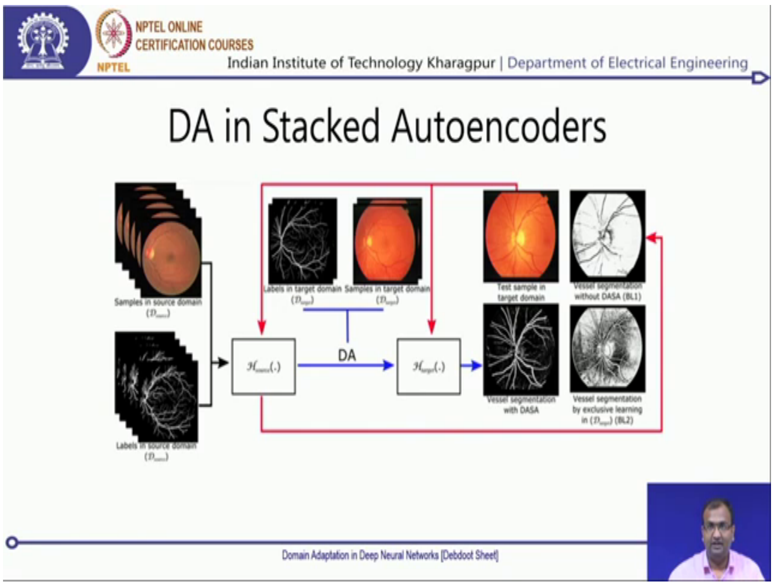
Can eyes; because all of these people are still the same people who are over here. And then I can have sketches. Now, can we match down from sketch to a real image that is a real target problem over here. So, that would be of immense use in forensic sciences or in crime investigations as well. Now, these are practical challenges associated with one of these problems in face recognition itself.

(Refer Slide Time: 17:14)

Then, we have this standard benchmark datasets on Webcam DSLR Object Recognition; where the object is that, you have these very high quality product level images taken down from DSLR as you get down on your Flip kart or Amazon, Snap deal any of these places.

Then you take a camera phone image, you just snapped it out from one of the bags which was lying somewhere, you like somebody's clothes. Now, can I match it exactly and find it up. So, this is another challenge which the community is facing today.

(Refer Slide Time: 17:40)



Now, where I would come down is one of our earlier works, which we call down as Domain Adaptation within Stacked Autoencoders and the whole idea over here was the examples which I was showing down. So, if you have your DRIVE data set on which you have trained it out and then on your STARE data set how will it work on. So, as one extra example which do get pointed out and not many people do refer to it is; a very pertinent question at this point.

So, I have my DRIVE data set on which I have trained data would create; now, I am saying that I want to modify that model which has already been trained on DRIVE onto my STARE data. Now, say if I do not do that what will happen; so, if I say that I have the whole STARE dataset, why not train a whole model which STARE data set and just work it out.

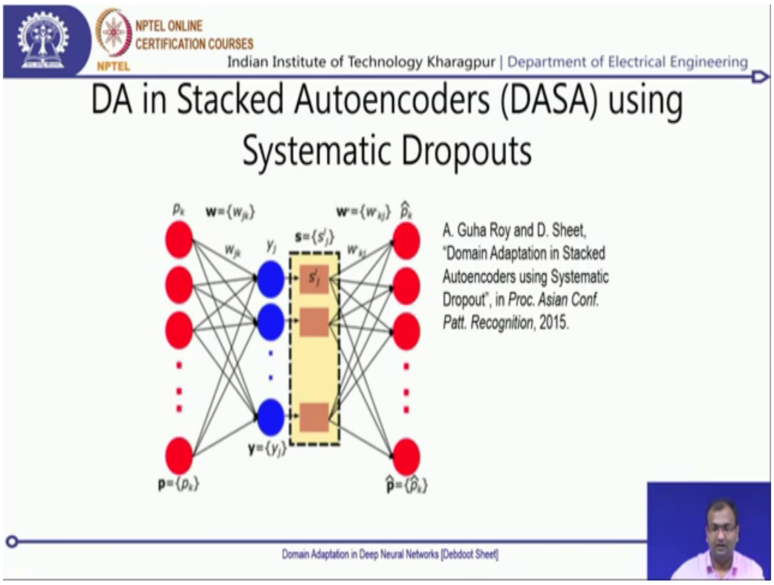
So, that is what we had also tried doing over here. So, what we did is that say you have your model which was trained down over here, which is called as H Source. And then you directly plugged in one of these stress samples onto it and this is what you get down, really bad result over there exactly opposite in some sense over there.

Now, on the other side of it what I try to do is that, I try to train a model you exclusively using only limited number of samples present in STARE data set and that is really small. And if you look into the number of examples available where you are vessels are match, that as low as four just four a images are present. Now, this is what the performance will come down which is also pretty shabby.

Now, on the other side of it, if I take a model which has been trained on DRIVE and then I just modify it over here. I get down my newer data set which is on my target domain and this is where it comes down.

So, you can see effectively that because it had learnt very nicely to discriminate out vessels and everything, I just needed a bit of modification to match down exactly to the newer domain over here and you get down pretty fine polished out results over there. So, that is why domain adaptation is actually used in a big way in the community, ok.

(Refer Slide Time: 19:35)



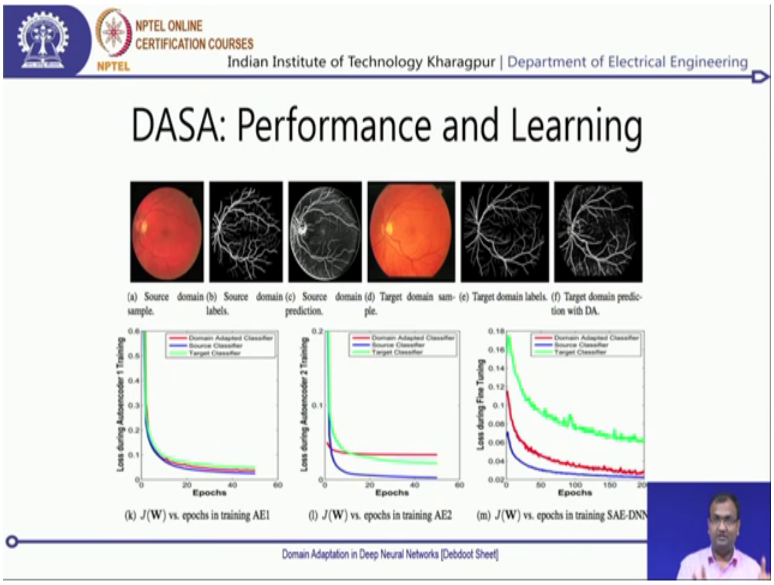
Now, what we had done over there was using a concept called as Systematic Dropouts; now, on one side while you have looked into these aspects of dropouts or randomized dropouts to avoid over fitting. The whole idea over here was to work down on something called as a systematic drop out. And the purpose over here is that some of these neurons; so, so, you have your inputs given down over here and you have your intermediate hidden layer outputs being created.

Now, some of these neurons will be something which will be aid in producing out your output correctly or they are the ones which will be positively reflective of features. Some of these neurons do not care anyway in whatever changes happened down, they are they are just garbage neurons. But there will be some neurons which will negatively impact your classification over there.

The whole purpose over here is to stop down; so, this is a switching layer it is just a 01, 01 kind of a switching layer, which finds out those neurons which negatively impact your classification and just stops it out. And on the back propagation side, it will try to update the weights only of these selective neurons.

So, that is what we have done down in this particular kind of a model for adapting where, deep neural networks for from a source domain to a target domain.

(Refer Slide Time: 20:47)



So, with that what comes out is interesting over here, if you look and the final point which is just for your classification losses. Now, this blue curve is the kind of a curve which you get down on the source domain when the classifier is trained. Now, if you look down into your green curve this is where is the performance of your target domain classifier, which is where I just use 4 images from the STARE data set in order to train the classifier to find out the blood vessels over there

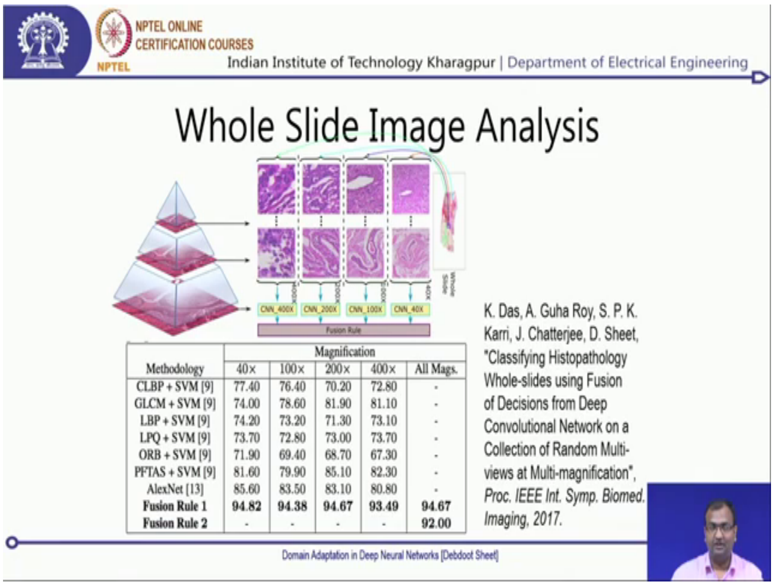
Now, the final accuracy the saturation saturation error is somewhere around 0.7 close to 0.7. That is almost the starting accuracy for my DRIVE data set where, I had a lot of data. So, that is that is; obviously you have more more data more labeled samples so, it is learning out very good.

But then, what you would be interested to look at is that, instead of training this one with limited number of samples. If I take a model which was trained on DRIVE and then I modify it first here, then might starting error over there is itself low. That is much lower than the starting error for directly starting over there. And then this final error is somewhere which is located very close to my DRIVE data set.

So, this is what you saw that if you are trying to modify network which has been trained with large amount of data, more number of images on a source domain. And then just used for a target domain, then you can actually come down to much lower errors over there; and this is quite good. Because now, this would reduce for most of your actual engineering problems you can actually make it faster within very few number of epochs, you can modify your network.

You do not need to keep on running for 100 epoch, 200 epoch or even 1000 epochs. You can now just modify your network within extra 5 epochs, 10 epochs or maybe maximum till 100 epochs and still get down, a very good performance coming down. So, this was one of these examples with retinal images.

(Refer Slide Time: 22:45)

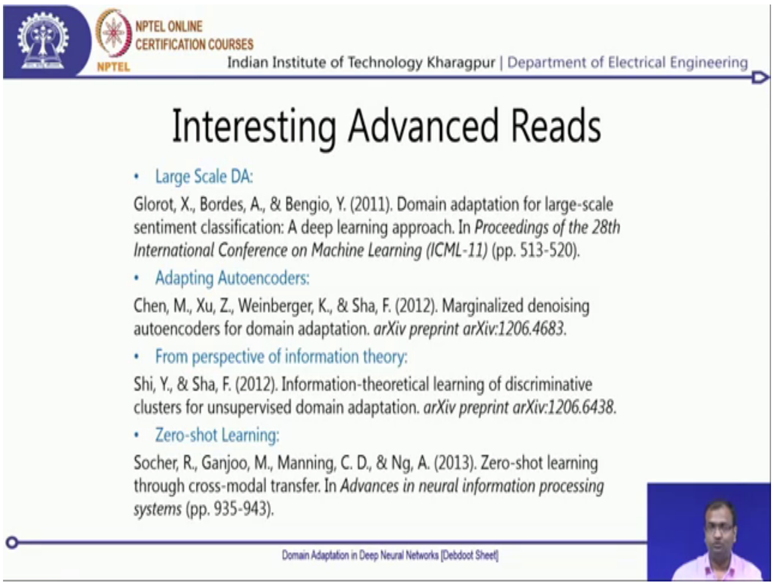


Another example is working at on digital pathology this where, the whole idea which we had put down was to take a network which was trained on standard image net kind of problems. And once you have those networks what you need to do is; so, there was obviously, a voting and a multi-view rule applied over there.

But then nonetheless these networks over there, which were already pretend for just imaginary problem which is natural image classification. And what I am showing over here, are medical grade images. These are histopathology images, it takes pathologist years of experience to in order to actually come down to understand patterns, very visible and pertinent patterns of (Refer Time: 23:19) on these slides of a tissue samples over there.

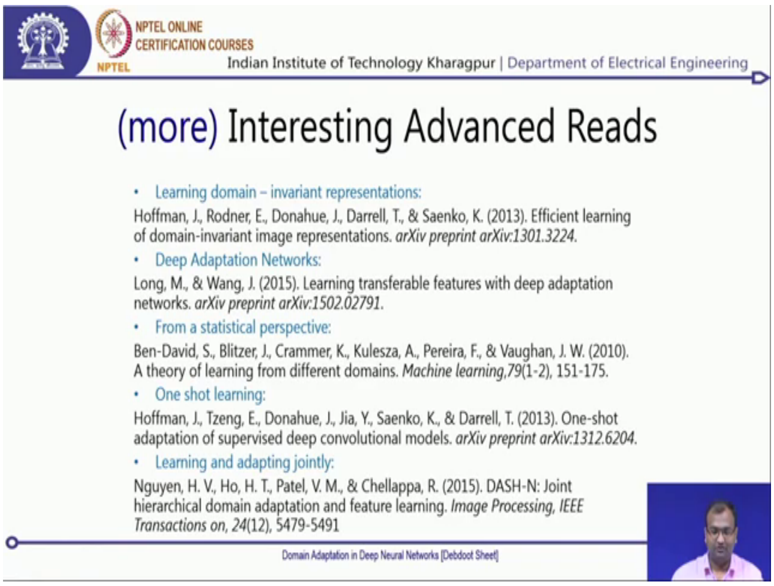
Now, we could modify very easily just within less than 10 epochs, to make a network workout to actually be a pathologist eye equivalent. So, that is a real case of using it down. So, I can refer you down to this particular people from is be 2017 where we had details of it, not really given down on how to modify it down for your work and purpose, ok.

(Refer Slide Time: 23:47)



So, if you are more of interested in looking down into other issues of domain adaptation, how to solve it out; this is the whole list.

(Refer Slide Time: 23:55)



Since, the slides will be available so, I am not just going to stick down onto this one; you can have your look at it at a later on point of time. Stay tuned in the next subsequent ones; we will be actually looking into adapting a classifier for our purpose.

So, we will download a whole train model over there and then see how domain adaptation actually comes down to end, convergence and compare and contrast it with a standard model where you are not using your domain adaptation to train it. So, these are interesting points which will be doing on the lab sessions. So, still then stay tuned and;

Thanks.