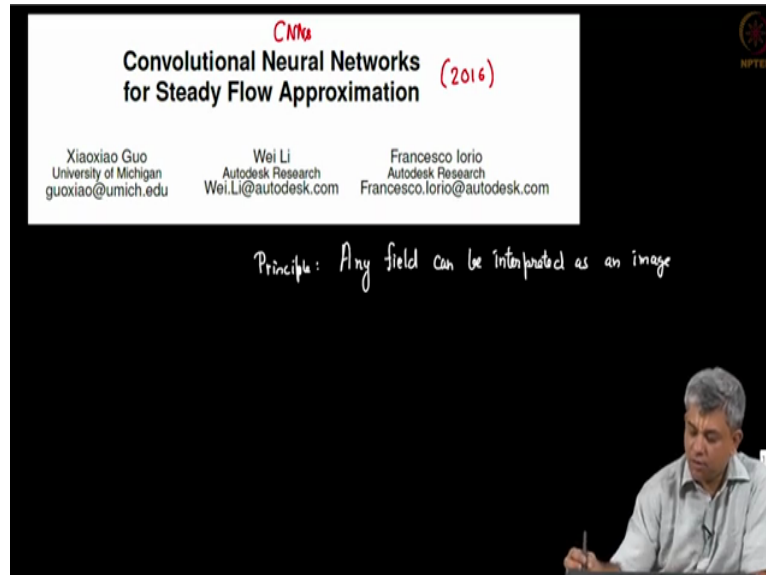


Machine Learning for Engineering and Science Applications
Professor Dr. Balaji Srinivasan
Department of Mechanical Engineering
Indian Institute of Technology, Madras
Application 2 solution

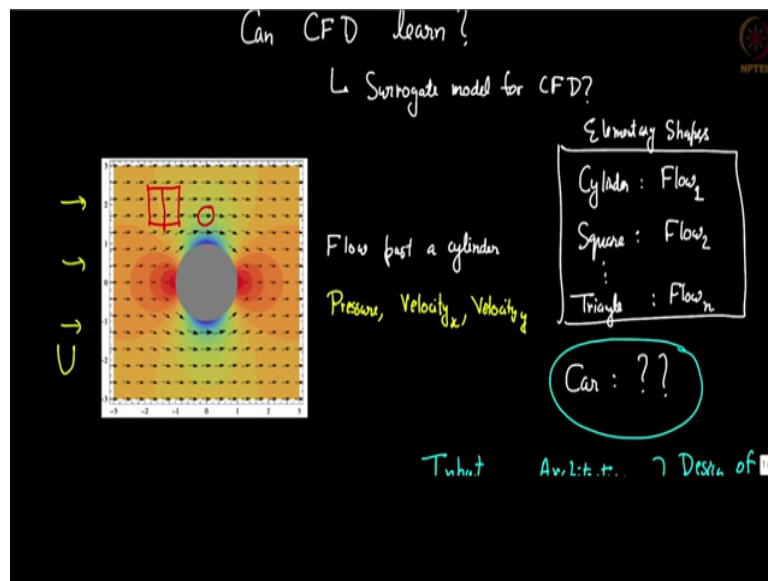
(Refer Slide Time: 0:16)



Welcome back, in the previous video I had set up a problem of trying to find out a surrogate model for computational fluid dynamics. In this video I will talk about one very recent solution, in fact this is a very good paper it is in 2016 created by people in auto desk. Normally I would have like to show you the full paper this paper is available online legally I am just going to describe results from this paper, there are people within my lab also working on similar problems, but this is the original paper I have taken sections from this paper instead of showing you the full paper because it is little bit convenient for me for me to show a few sections from this paper.

So please do read the full paper, it is very well written and the results are actually really really impressive. Okay, so what they used was as you can see in the title CNNs, so the first trick that they have used which I would like you to think about seriously since CNNs have become extremely powerful over the last 5, 6 years is the fact that any field this is the principle, any field can be interpreted as an image.

(Refer Slide Time: 1:45)



So to explain the flow past a cylinder I showed you in the last video, you can interpret this if you just look at the contours that are shown there as an image even the whole thing right now is just appearing to your eyes as an image. To me as a person within CFD this is velocities and pressure, etc but to you all it is in case you are not familiar with CFD it is just an image. So we are actually going to exploit or these people these researchers actually exploited this fact that what looks like physics can be interpreted either as data or as image.

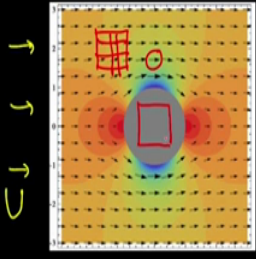
Now before we proceed please remember that this interpretation is a little bit easy to apply in case you have what I will call homogeneous data as in when I see a pressure contour here, this contour or the colors that you see here represent just one physical quantity, I mean it is not as if this portion is pressure and this color is velocity and this color is something else. So when you have heterogeneous data and it is not field data, it is usually useful to use simple ANNs, CNNs are useful when you have images, why is an image useful? Remember the basic idea of a CNN is that spatial closeness actually matters and we want to account for that within our weights or within our kernels or within our filters, okay.

(Refer Slide Time: 3:14)

CAE

Can CFD learn?

↳ Surrogate model for CFD?



Flow past a cylinder
Pressure, Velocity u_x , Velocity u_y


Elementary Shapes

- Cylinder : Flow₁
- Square : Flow₂
- ⋮
- Triangle : Flow_n

Car : ??

Principle: Any field can be interpreted as an image

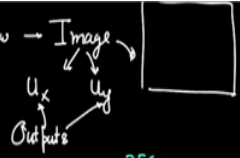
Output: Section of the flow → Image



u_x u_y
Outputs

Input : ? Shape, but how?

Output: Section of the flow → Image



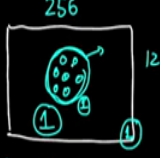
u_x u_y
Outputs

Input : ? Shape, but how?

CNN

INPUT : OUTPUT

One possible input



256 128

0: If inside
1: If outside

So what the idea was that these people used was the input is simply a section of the flow and it is taken as an image and you will see the size of the image later on when I show you the architecture, they took one section of the whole flow, the flow can actually be quite big, there is one portion that you are interested in, you just cut that out and you take that as an image. now this idea of course is very general I will show you another iteration of this idea when we discuss problem 3, but this idea of interpreting a field as an image is actually a very straightforward one, okay.

Now when we take an image please remember the image itself for the sake of this particular paper, they took two images one of the x component of velocity, another one of the y component of velocity both these you can just take as images, okay. Now these are our outputs (I am sorry I should have called this output rather than input) so what we are interested in is actually the velocity of the flow as output, okay.

Now what is the input? This is a little bit more interesting, okay. Now we know that what causes the flow in this case for example is the shape of the body, now the shape is a little bit (())(4:51), now I want to if I change this shape to let us say a square, as a fluid mechanics person I know that I am going to get a different flow, you might know this or you might even see this intuitively even with even if you are not within fluid mechanics. So the shape is what determines the boundary condition, the boundary condition is what determines the flow finally. So if I change the shape the flow changes, so that is the input but how do I give it?

So we know that this is supposed to be the shape, but how? So we have several options let me give you a couple of options actually the people within the researchers within this paper themselves considered it. So one thing is they had decided that they are going to give they are going to use CNNs okay so that much is decided. Once you have decided that you are using CNNs pure CNNs you have to give an image as an input that is also determined, the special thing here is we have an image as output also, it is not a classification it is it is actually a full image just like you saw in segmentation tasks, okay.

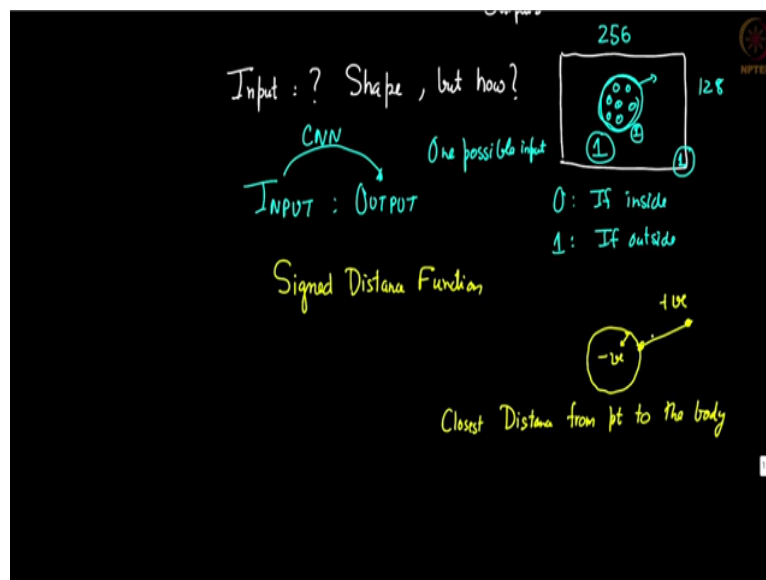
But the input is also an image, so how do we give an image as input? So let us say I want to give the image of a circle as an input, there are a few ways of doing it, one very simple way is to say let us say this is 256 cross 128 pixels and within each pixel I will label it as 0 if it is inside the body and I will label it as 1 if it is outside the body, okay so is this clear? 0 if inside the body, 1 if outside the body, this is one possible input.

Now as it happened when people did this, when the researchers did this the results were kind of okay but not so good, okay. So remember when I have this kind of problem what I am saying is for a specific input I am going to somehow map it using some CNN architecture to an output, image to image, okay. I am not going to talk about the architecture right now, I will talk about it in a few minutes.

But somehow if you are able to map this input to an output, they saw that if you label the input as 0's and 1's it does not work very well. Now why does it not work very well? Here is where your domain knowledge comes (into place) into play, okay. Now we know from fluid mechanics that it matters how close you are to the body? Perhaps you can see this even whether even if you do not know fluid mechanics, the further and further I am away from the body, the lesser and lesser it will matter what the shape of the body is, okay.

Similarly if I am close to this body and I label that pixel or location 1 and I am far away from that body and I still label that pixel or location 1 then it seems like some portion of the physics of the problem I am completely throwing away, I am saying that this portion of the body or this portion of the flow is as important as this portion of the flow as compared to the shape of the body which is not true.

(Refer Slide Time: 8:30)

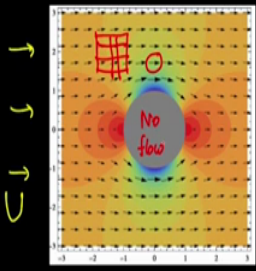


So these people actually came up with a slightly modified idea it is an intelligent idea it is called a signed distance function, okay what is a signed distance function? It works as follows inside the body it will be negative, outside the body it will be positive and the value will be the distance closest distance from the point to the body. So suppose you are at this place they

will find out the closest distance from there to the body and if it is outside they will label it positive, if it is inside they will label it negative and this way you have something that accounts to the for the fact that points that are closed to the body are actually more important than points that are further away from the body as far as the boundary conditions are concerned, okay. Let me show you a contour for this.

(Refer Slide Time: 9:36)

↳ Surrogate model for CFD?



Flow past a cylinder
Pressure, Velocity_x, Velocity_y

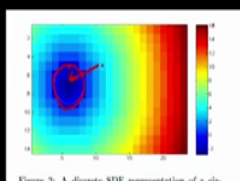
Elementary Shapes

- Cylinder : Flow₁
- Square : Flow₂
- ⋮
- Triangle : Flow_n

Car : ??

Input Architecture } Design of
Output Loss Function } DL

Signed Distance Function



Closest Distance from pt to the body

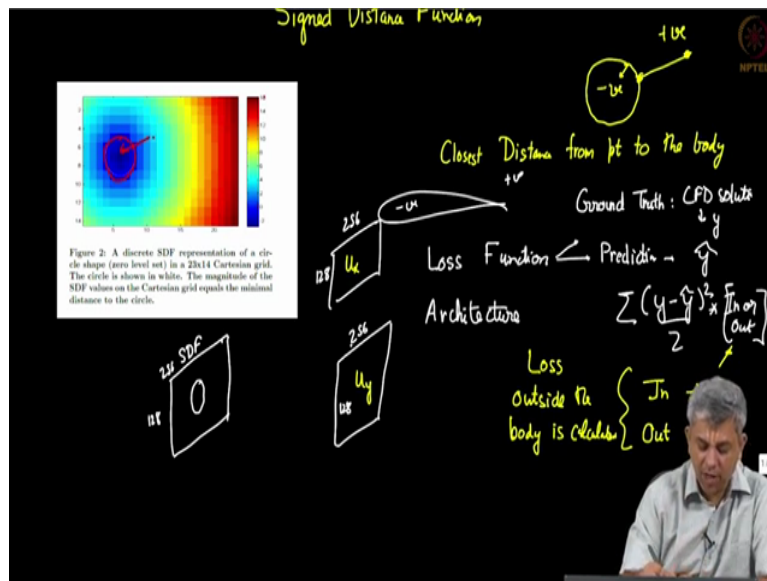
Ground Truth : CFD solve

Loss Function ← Prediction →

Architecture $\sum \frac{(y - \hat{y})^2}{2} \times \begin{cases} In \\ Out \end{cases}$

Loss

outside the body is ok $\begin{cases} In \Rightarrow 0 \\ Out \Rightarrow 1 \end{cases}$



So you can see here once again they have drawn this signed distance function exactly for a circle as I have shown here, you can see that 0 distance is dark blue so somewhere here is where the body is, further and further away is more and more positive, inside it is negative and you can see this body, actually they have shown the outline, this is actually the body, okay. So the body is here and this is how they have given the signed distance function.

Now for any object you can give a new signed distance function let us say it is an (\cdot) (10:09) once again negative here, positive here and then find out the shortest distance from any point to the surface and then label it positive or negative. So this is how they have given the input and once again here is where we have used some amount of domain knowledge, okay. Now that we have decided the input, we have decided the output, we need to decide two more things, we need to decide the loss function and finally and in some sense most importantly you have to decide what is the architecture, what map, what model are you going to give mapping this image to the image of the flow.

Okay, so let us first discuss the loss function, the loss function is very simple for each flow I have my ground truth, this is ofcourse in the training set, the ground truth is the CFD solution. So you have some code they used something called LBM [Lattice Boltzmann Method] in order to generate lots of flows, so they found out CFD solutions and you have your prediction, as usual let us call this y , we will call this \hat{y} .

So the loss function you have several choices, I will just mention one that you could use $y - \hat{y}$ square by 2 ofcourse this is our least square loss function except with a small change, we will multiply it by one small number I will just call it in or out. We obviously

know that the flow within the body here even if the CNN gives the prediction that is actually useless.

So this here we have no flow at all. So even though you are still since you are (imagic) interpreting the whole thing as an image you are going to get a loss inside the body that part we will not count, okay so (in means we will use 1) sorry in means we will use 0 we will not count the loss, out means we will use 1 so that is only loss outside the body is calculated.

Another way to say it is if the signed distance function is negative you will assume that there is no loss and if the signed distance function is positive we will actually calculate the loss and incorporate it (into the architecture) into the algorithm, okay. Now the last step is we want to find out what the architecture is that maps input to output, so let us see that. I am going to give a rough architecture now and then I will show you the architecture that they used. Now remember we have an image as input, this image was the signed distance function we call it SDF this was a 256 cross 128 image, we also have an image as output again 256 cross 128, now you have two possible images that you have to give, one of these will be the x velocity field and one of these will be the y velocity field, okay.

Now once again it is our domain knowledge that tells us that there are effects of two things, one is the effect of physics which is the PDEs and another one is the effect of the geometry which is given by the boundary conditions.

(Refer Slide Time: 14:24)

Figure 2: A discrete SDF representation of a circle shape (zero level set) in a 256x128 Cartesian grid. The circle is shown in white. The magnitude of the SDF values on the Cartesian grid equals the minimal distance to the circle.

Ground Truth: CFD solute y

Loss Function \leftarrow Prediction \rightarrow

Architecture $\sum \frac{(y - \hat{y})^2}{2} \times \begin{cases} In \\ Out \end{cases}$

Loss Outside $\begin{cases} In \Rightarrow 0 \\ body \text{ is closed} \\ Out \Rightarrow 1 \end{cases}$

Encoder-Decoder

Encoder \rightarrow Geometry \rightarrow Common to u_x & u_y

Decoder \rightarrow Physics $\left\{ \begin{array}{l} u_x \\ u_y \end{array} \right\}$ 2 decoders

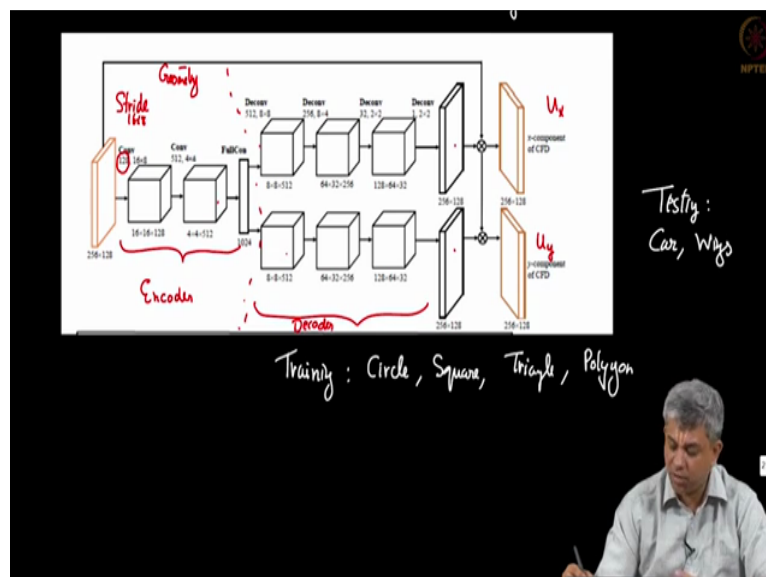
So what we will do is we will use first thing, remember I want to go from an image to an image. So typically this is very similar to what you did in segmentation tasks we will use

what is known as encoder, decoder structure you would have seen this several times during the CNN lectures. An encoder, decoder structure does the following, it takes an image squeezes it into some essential information and then expands that essential information, it is sort of like zipping a file and then unpacking it, you just remove the unnecessary stuff, pack it into something and then expand it.

Now what it does is essentially only the important pattern stay and the other stuff goes out, we will do the same thing encoder, decoder back into the full image, okay. Now an important thing is should you use two encoders and two decoders or one encoder, two decoders, etc because remember we are trying to predict two things and here is where our physics knowledge comes into play once again, we know that when I am encoding what I am encoding is effectively the geometry, this is not strictly speaking true, but it is true enough that I will just leave it like this, is what I am saying is regardless of whether it is the U velocity or the V velocity your geometry has to be squeezed.

So therefore this can be common because it only pertains to the geometry and not to the particular velocity field. And the decoder actually tries to incorporate physics and then you can split it into u_x and u_y , 2 separate decoders are used and ofcourse the corresponding loss function, okay.

(Refer Slide Time: 16:20)



So let me actually show you the structure that they used in their paper, here is the CNN architecture that the people used in this paper. Now notice what is happening here, this portion is the encoder until here, took a 256 cross 128 image, squeezed it and got a (fully

convolutional) fully connected layer of 1024, now this portion is a decoder portion and you have split it into two sections, one is for U_x and one is for U_y , okay.

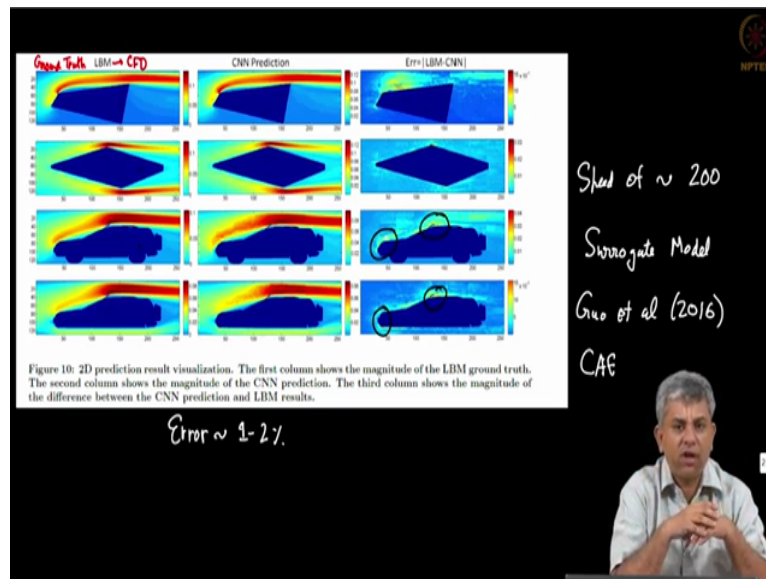
Specifically also see that you have 128 in the very first layer, 128 filters each of size 16 cross 8 which is why this 256 cross 128 now got squeezed into 16 cross 16 cross 128 as a simple exercise please think about what is this stride, the stride as you will see is actually 16 cross 8, so what it is doing is it is taking a block, it is not simply sliding, takes a kernel, it jumps, jumps and comes down. I will leave this to you as a simple exercise Doctor Ganapathy had discussed this in great detail, but this is just so as to avoid too many filters and you know too much computation, okay.

So this is a simple structure, similarly the next step is to use 512, 4 cross 4 filters and finally there is a fully connected portion and I like I said possible to interpret do not quite accurate to think of this as squeezing the geometry, the reason it is not quite accurate is we are actually doing back prop throughout, so some effect of U_x and U_y is actually going to go through. Okay, after this they have a decoder structure, you have transpose convolutions, you know you sort of expand this back right back to the same original size of 256 cross 128 you would have seen this several times during the segmentation tasks.

Finally we take the loss function, you get x component of CFD, you get y component of CFD and then you train this, okay. So once you do this the results are actually quite surprising and I would like you to see these results right now. Now this work by Guo and co-workers what they did was they used simple shapes the kinds that I had described earlier for which CFD tends to work a little bit fast for several technical reasons, they used circles, square, triangle, etc some polygonal shapes, okay and oriented them at different angles and found this out and found out various flow fields and they actually gave far more complex shapes such as for testing, so this is for training and what is remarkable is unlike our usual CNN examples where training and testing actually in some sense look similar, they tested on far more complex shapes.

If you know CFD, you know that from here to here mapping is actually quite hard, you cannot really say something quite obvious about this, so cars, wings, etc.

(Refer Slide Time: 20:05)



Now I am going to show you their results again I would refer you back to the paper, it is a very good paper it is available for free for academics as well as non-academics on the web please search for it, okay. So here you have their results, what is written as LBM is essentially CFD we can take this as ground truth and here is the prediction from CNN, you can see that you know picture to picture the match is actually remarkably good, this is actually extremely good, their error was around 1 to 2 percent.

So you can see the error drawn here, now as a person in CFD we know that you know these things can be somewhat worrying, so there is some error near the boundary which is actually important for people within CFD, but nonetheless overall this is actually a remarkably good prediction, the speed up is actually quite huge, so speed up of order a few hundreds depending on whether they use CPUs or GPUs their speed up rates were quite different.

So this is actually an excellent surrogate model, why is that? Because you can get up till 2 percent of the answer atleast in whole flow field you can get upto 2 percent of the answer about 200 times faster. So you can imagine if your design cycle is predicated mostly on CFD, it is not always, but let us say 60 to 70 percent is actually dependent on CFD you can actually get a speed up of a factor of 10 or 15 in your whole design cycle, something that takes an year can take about 20 days.

So you can see that just by using CNNs you can get really rapid this is ofcourse this is for some limited cases we are not there yet in terms of full 3D and all sorts of complexity flows but this is a great beginning. So I would refer you all to this original paper by Guo et al

whether you are interested in CFD or not? Ofcourse I spent along interminable amount of time talking about CFD and I will briefly introduce the next two problems because they are somewhat related but in general a CAE problem CAE being computer aided engineering, CAE problems in general can gain tremendously you know people even within chip manufacturing companies all of them do some computation or the other and CNNs have tremendous applications there, I expect more and more papers and indeed many papers have come out over the last year, this was 2016, 2018 to 19 many papers have come out and by the time we finish this course and by the time we finish this year is you will probably see the literature flooded with such examples. And I encourage you to just look at CNN application of problem x wherever x is a field problem, thank you.