

Applied Accelerated Artificial Intelligence
Dr. Tosin Adesuyi
Department of Computer Science and Engineering
Indian Institute of Technology, Palakkad

End to End Accelerated Data Learning
Lecture - 32
Optimizing Deep Learning Training: Transfer Learning Part - 1

I want to welcome everybody to today's lecture and we will be discussing Optimizing Deep Learning Training and under that will emphasize on Transfer Learning. So, today our focus is on Transfer Learning.

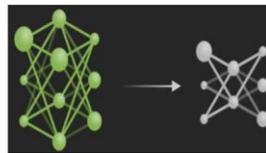
(Refer Slide Time: 00:35)



What is Transfer Learning ?



- **Transfer Learning** is a process of using learned features from a **pretrained model** to train new data. Thus, a new model is produced.
- The **new data** is trained using **weights** and **hyperparameters** from the pretrained model.
- The hyperparameters may be tune to fit desirable accuracy for the new model



Source: <https://openstax.org/r/what-is-transfer-learning/>

Let me proceed I want to tell you that we would not waste much time today it would be more of hands on than the theoretical part. So, first of all we want to know what we mean by transfer learning? So, transfer learning is a process where you have your whole new data and there is an existing pretrained model which you can use your new data to be trained using that existing pretrained model. So, in a simple term we can say it is a process of using learned features from already trained model.

So, that is the process of what transfer learning and now you do that you do that by using the weights and the hyperparameter of the pretrained model to achieve that purpose. Then in order to get a good results or desirable accuracy, so what you do is to also tune

for the hyperparameter of the already trained model which will refer to as pretrained model.

So, this give you the ability not to start building your network layer from the beginning, your deep neural network rather that is you are thinking of ok. How many convolutional layer do I need to have how many pooling layer do I need to have you do not need to bother about that again, because there is an existing pretrained model which you now bring your new data you supply into the existing pretrained model which would give back to the new model with your new data.

(Refer Slide Time: 02:36)

The slide is titled "Why Transfer Learning?" and features three main points, each with an icon:

- Required less data to train:** Represented by an icon of two stacks of green coins, with the shorter stack on the right.
- High learning speed, therefore training time is reduced:** Represented by an icon of an hourglass with a green arrow pointing downwards from the top.
- Save cost:** Represented by an icon of a hand holding two coins with dollar signs.

A source link is provided at the bottom: <https://blogs.mafra.com/blog/2019/02/07/what-is-transfer-learning/>

So, why Transfer Learning? Why do we need to use transfer? And one of the main reason is that you do not need to build your model from the scratch again, because this is very important. Nowadays where there are complex tasks to solve and those complex tasks require large models to be built so and for you to start from the beginning is a lot of labour and a lot of brainstorming.

So, since there are large deep neural network existing which are robust their state of the heart. So, we can leverage on those existing deep neural network by bringing our own new data and using that existing network deep neural network to train our own data and we can harvest our results.

So, one of the major advantage you get is that it requires less data to train you only need to use lesser data based on the complexity of the tasks you want to achieve. And why do you need that? Why do you need to use less data is because the already pretrained model has been trained with large data before.

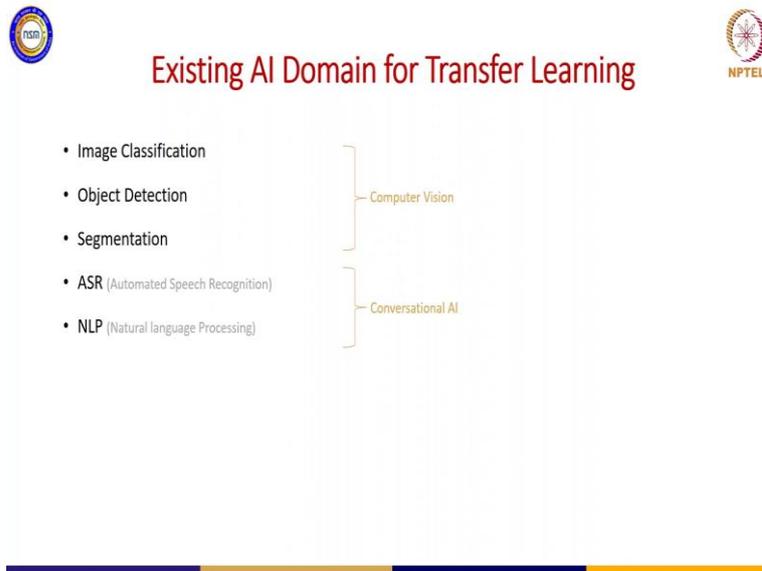
So, the model is as the ability to recognize variation in data and it is it has the ability to lend those futures. So, you already know those futures already. So, when you want to train your own new data you do not need large data which match the complexity of the task you want to train.

So, you only need just a handful of data what I mean handful of data I am not saying, so small data about a sufficient data to level. What you used to train? So, it require less data than the usual one then also you have the time of training of your data will be reduced that is you have speed training time speedy training time, because why some of the things that your data supposed to your model supposed to learn if you are building it from the scratch.

They already pretrained model they already have those features already know how to start those features. So, it does not need to take long for it to start those features, so it learn faster and lastly is that it saved costs. So, especially we look at it from the domain of computer vision, we can see computer vision one of the main challenges there is data is scarce data is expensive based on the tasks you want to perform.

So, if we look at the health domain data there is also scarce. So, when you have a pre trained model which already has those capacity it is been trained with large data before. So, you do not need to use much data, so it saves you cost of buying data that you need you require to train your model.

(Refer Slide Time: 05:53)



We will look at Existing AI Domain for Transfer Learning. What are the domains? Where we can use transfer learning in high we have the image classification they are transfer learning model which exists for that also object detection, there is transfer learning which also exists model exist in that also.

Then segmentation image segmentation transfer limit also exists there. So, those three are classified under computer vision and then there is the automatic speech recognition there is also a model for that and also the NLP Natural Language Processing. So, these 2 also they classify under conversational AI they this domain they have transfer learning model transfer learning is possible in this domain.

(Refer Slide Time: 06:49)



Transfer Learning With NVIDIA TAO



- TAO (Train, Adapt, Optimize)



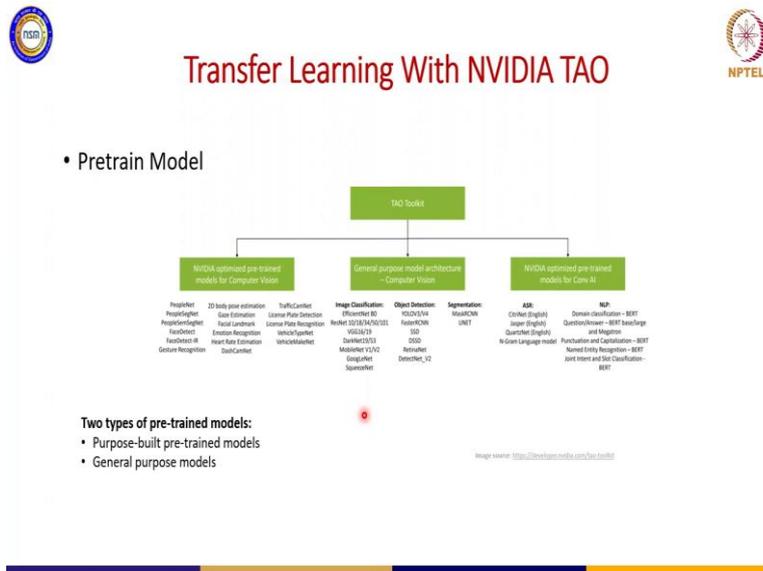
Image source: <https://developer.nvidia.com/tao-toolkit>

We will talk about transfer learning with NVIDIA TAO this is a framework which we can use as for achieving transfer learning and TAO means Train, Adapt, Optimize. So, you can get your pretrained model from NGC I will show you I was talk about NGC as we move further and you will also have your custom data this is your custom data.

So, once you pull your pretrained model from the NGC, from the NGC sorry let me see if that is ok you pull your pretrained model from the NGC everything go goes into TAO inside TAO you can do data preparation and augmentation you can take train inside the toolkits and then you prune after you prune then you have your customized AI model.

So, all of this will run on the CUDA using your container and then the CUDA is there cuDNN for the training, then when you want to inference you can proceed further to optimize using what we call tensorRT and the training can be done on your workstation on DGX on the cloud you can reference, inferences on this platform as well. So, for more information on this you will be able to you can visit this link.

(Refer Slide Time: 08:17)



So, what are the pretrained model that exist are on NVIDIA TAO, we have 2 types of pre trained model that exist there we have the one called purpose built Pretrained Model. Now this pre trained model they are built for you to use on the go.

You may not even need to supply your neural data they are just built for a particular purpose. Example is the PeopleNet this model recognize persons there is also the 2D pose estimation it recognizes human pose estimation, there is a license plate detection license plate recognition all of these exists. Also it does exist in the NLP domain as well where you have the BERT you have megatron and the likes are there.

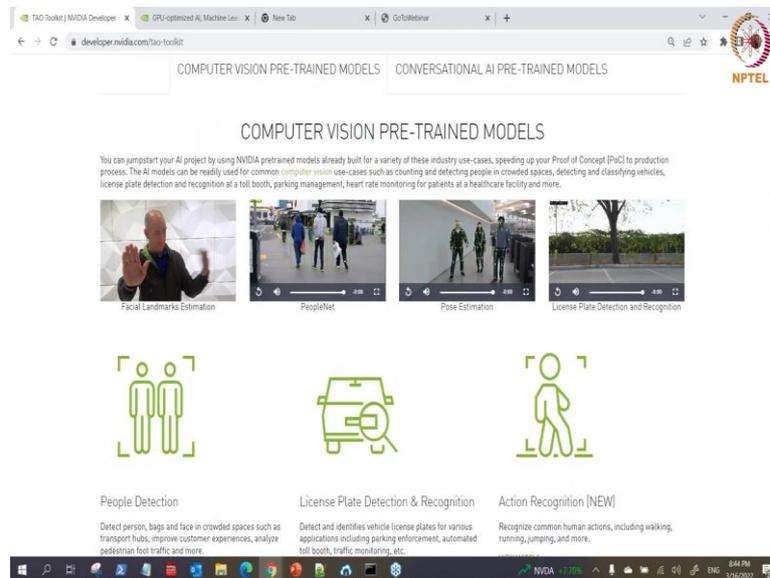
And the second one is the general purpose model that is this general purpose model you can bring in your own data your new data to in for your own specific tasks you can use those model for to achieve that purpose. So, and under that we have 3 categories which are image classification, object detection and segmentation.

So, for under image classification you have the EfficientNet which is there very popular, there is the RestNet 10/18/34/50/101 layers there is the VGG 16 and 19 the DarkNet which was used for Yolo the MobileNet GoogLeNnet and the likes they are there as well.

So, for object detection there is a YOLOV4, YOLOV3, faster RCNN and for segmentation you have the maskRCNN and UNet. So, you can pick this Pre trained

model and take your new data and supply it to this pre trained model and train using that Pre trained model to train your data then you can now have your new model and make use of it.

(Refer Slide Time: 10:26)



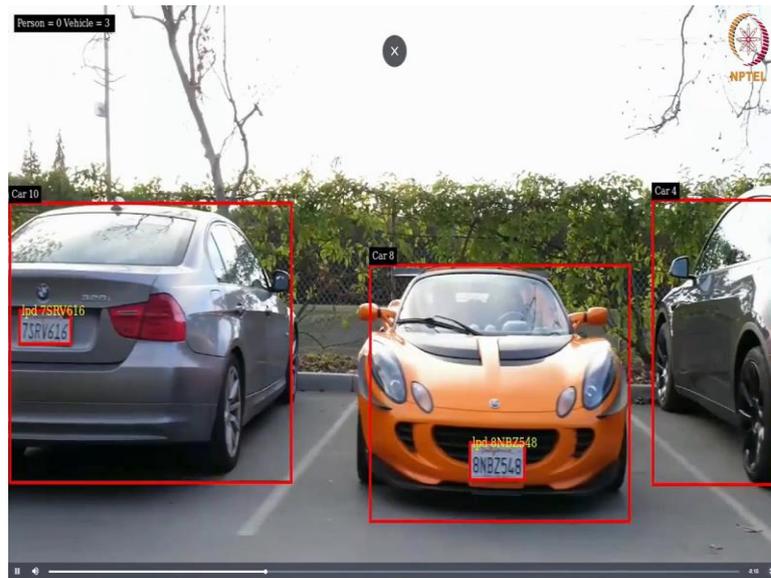
So, examples are what you can see here I will play this you can see.

(Refer Slide Time: 10:32)



So, this is you can see its recognizing the plate number.

(Refer Slide Time: 10:34)



That is a car it show you how many vehicles are on the screen 0 present.

(Refer Slide Time: 10:42)



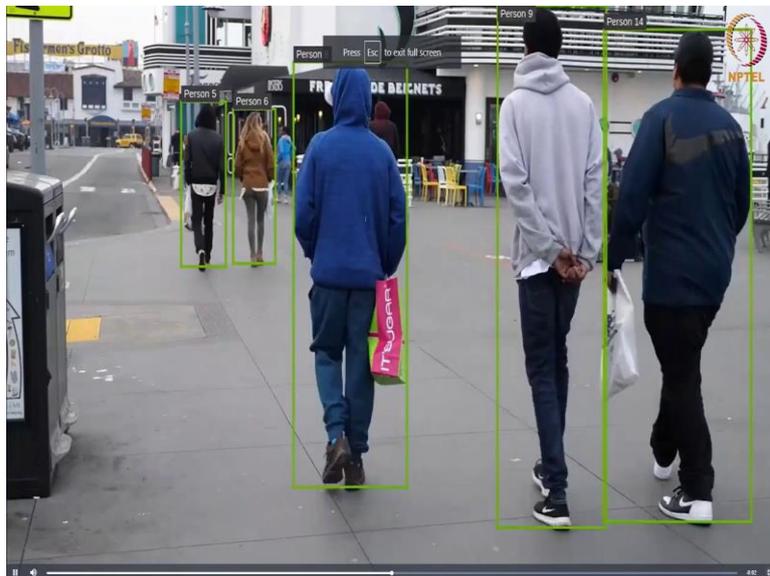
This plate number and you can also recognize what the number on the plate's number as well. So, this are general purpose I mean these are specific purpose model which you can use specifically for license plate, probably somebody who is working on the toll gate. So, this can be used deployed for a toll gate use.

(Refer Slide Time: 11:04)



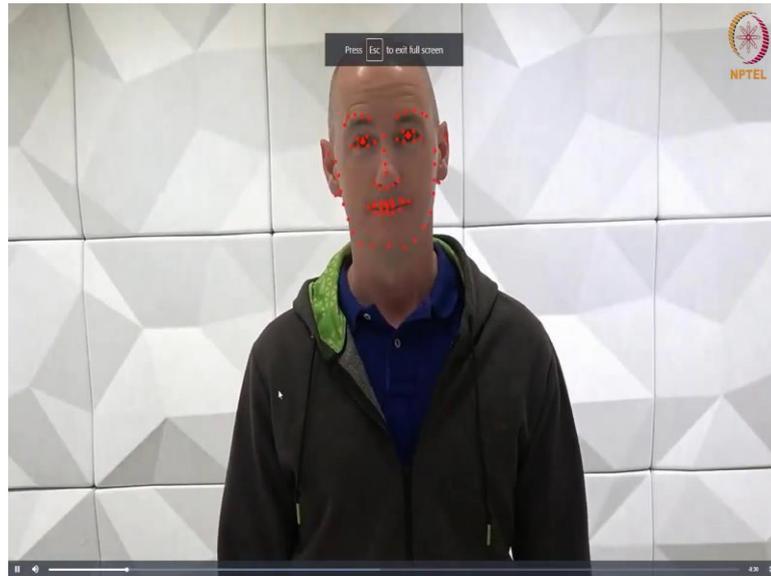
There is also what we call the image pose estimation. So, this one they can use it for action recognition actual recognition that what is that person doing is it dancing, is it the person is it jumping or is it picking something there. So, they can use this as well for that and there is the PeopleNet which I talked about as well.

(Refer Slide Time: 11:26)



So, it recognizes women that is a person and is counting them as well.

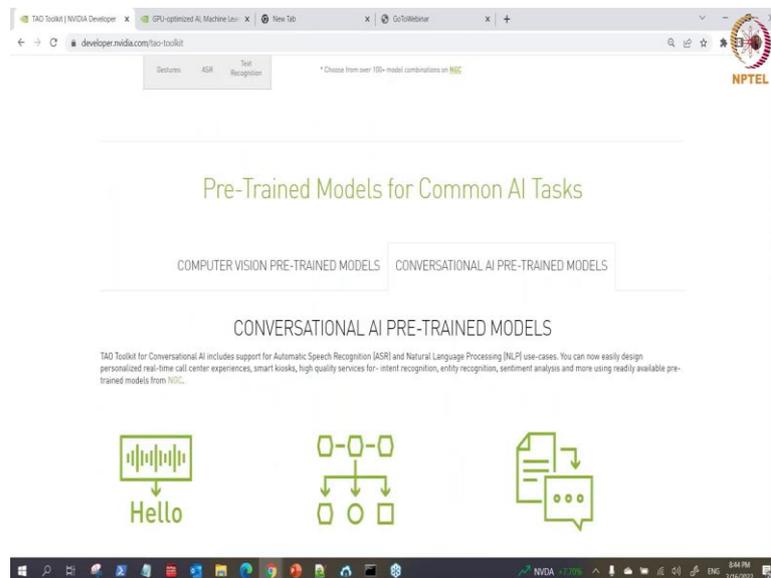
(Refer Slide Time: 11:35)



Then we also have what we call the face marks estimation as well.

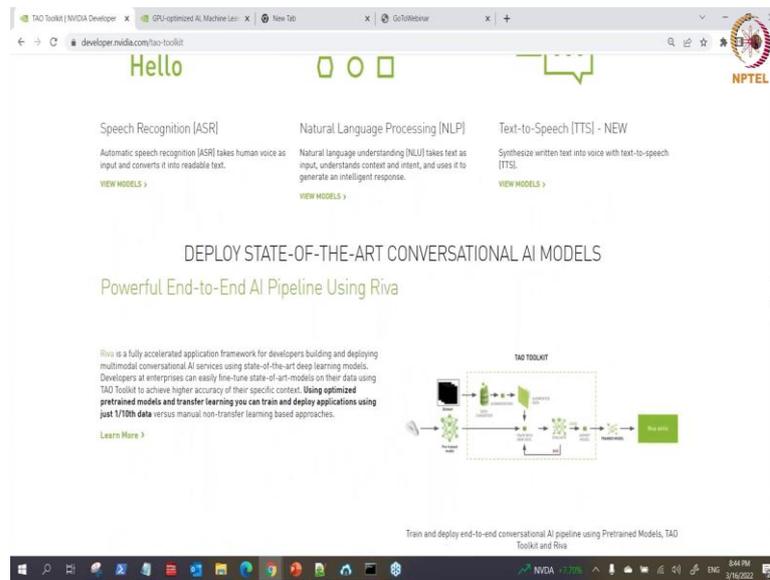
So, it can grab your face it is like a focus, where if you turn to the other side or any side it can get the features of your of your face there. So, there are many of them that exist.

(Refer Slide Time: 11:52)



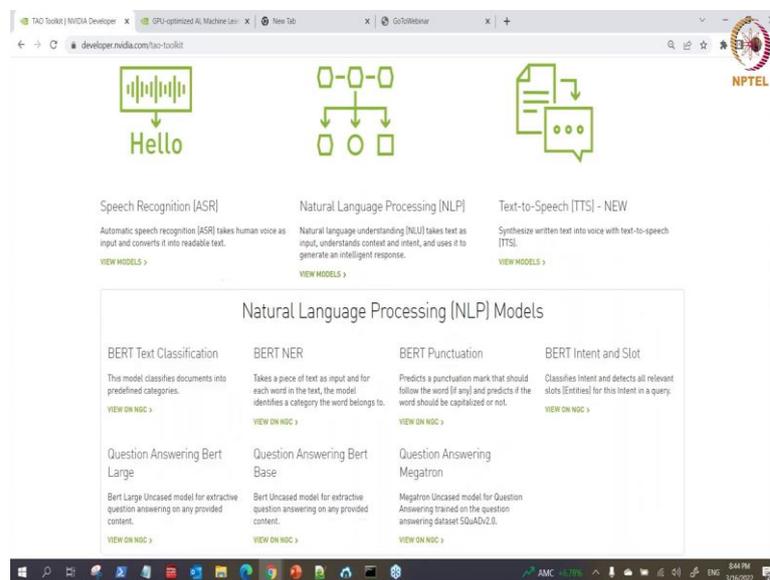
They are also in the conversational AI Domain as well there are many pretrained model.

(Refer Slide Time: 11:55)



Which you can see with NVIDIA TAO there is also text to speech natural language on that you can see these models are already there.

(Refer Slide Time: 12:01)



BERT for text classification and the likes there are so many of them that those exist.

(Refer Slide Time: 12:03)

The screenshot shows the NVIDIA TAO Toolkit website. At the top, there are four columns of BERT models:

- BERT Text Classification**: This model classifies documents into predefined categories. [VIEW ON NDC >](#)
- BERT NER**: Takes a piece of text as input and for each word in the text, the model identifies a category the word belongs to. [VIEW ON NDC >](#)
- BERT Punctuation**: Predicts a punctuation mark that should follow the word (if any) and predicts if the word should be capitalized or not. [VIEW ON NDC >](#)
- BERT Intent and Slot**: Classifies intent and detects all relevant slots (Entities) for this intent in a query. [VIEW ON NDC >](#)

Below these are three more models:

- Question Answering Bert Large**: Bert Large Uncased model for extractive question answering on any provided content. [VIEW ON NDC >](#)
- Question Answering Bert Base**: Bert Uncased model for extractive question answering on any provided content. [VIEW ON NDC >](#)
- Question Answering Megatron**: Megatron Uncased model for Question Answering trained on the question answering dataset SQuADv2.0. [VIEW ON NDC >](#)

The main heading is **DEPLOY STATE-OF-THE-ART CONVERSATIONAL AI MODELS** with the sub-heading **Powerful End-to-End AI Pipeline Using Riva**.

A diagram titled **TAO TOOLKIT** shows a pipeline from **TAO Toolkit** to **RIVA** to **AI Applications**.

Riva is a fully accelerated application framework for developers building and deploying multimodal conversational AI services using state-of-the-art deep learning models. Developers at enterprises can easily fine-tune state-of-art-models on their data using TAO Toolkit to achieve higher accuracy of their specific context. **Using optimized pretrained models and transfer learning you can train and deploy applications using just 1/10th data** versus manual non-transfer learning based approaches. [Learn More >](#)

(Refer Slide Time: 12:07)

The screenshot shows the NVIDIA TAO Toolkit website. It features a diagram titled **TAO TOOLKIT** showing a pipeline from **TAO Toolkit** to **RIVA** to **AI Applications**.

Riva is a fully accelerated application framework for developers building and deploying multimodal conversational AI services using state-of-the-art deep learning models. Developers at enterprises can easily fine-tune state-of-art-models on their data using TAO Toolkit to achieve higher accuracy of their specific context. **Using optimized pretrained models and transfer learning you can train and deploy applications using just 1/10th data** versus manual non-transfer learning based approaches. [Learn More >](#)

Train and deploy end-to-end conversational AI pipeline using Pretrained Models, TAO Toolkit and Riva

Data Generation & Data Annotation Partners

Training AI requires lots of high quality labeled data and we have partnered with several companies to bring data creation and annotation to accelerate training.

- CVEDIA**: Offers end-to-end synthetic computer vision solutions for object detection and image classification.
- Hasty**: Annotation solution using AI to significantly speed up labeling.
- SKY ENGINE AI**: Next generation self learning AI system for image and video analysis applications.

So, we are back here.

(Refer Slide Time: 12:13)



So, let me talk about the overview of the TAO to with respect to computer vision. So, what happened is that you have your data here your raw data so you need to process the data convert it to a particular format. If you want to use transfer learning, so using transfer learning you need a pre trained model for you to be able to use a pre trained model. You must have full details the know-how of the pre trained model what kind of data what is the dimension of data it can accept, what is the format of the data you can accept.

So, you do that here you convert your data augment your data and everything goes into train and where does it train you bring in your world, your pre trained model elsewhere. So, then you train using the pre trained model. So, when you train you evaluate your training results, if it is bad you have to train go back and train and then if it is good you proceed to prune.

Now, pruning is a level of optimization such that it reduces the size of your model and when it reduce the size of your model your model doing inferencing your model will be faster that is the purpose of that it will be faster. But there is a detriment to that it says that because its prune your model your model the accuracy is then to drop a little bit. So, what happened is that after you prune it is advisable to retrain again.

So, that you can get back your accuracy while you have the benefit of small size model with faster inferencing. So, you retrain after you retrain you evaluate the training if the

training the result you are getting is not good as before you prune you go back again and re train. If the result is ok is good then you can export your model and use it forever or whatever you want to use.

And along the line you can also do what we call INT8 calibrations which is also a form of optimization to make your model also run faster. So, this is all what I have explained I have showed you before about those AI Model that are ready you can use.

(Refer Slide Time: 14:37)

The slide features the IITM and NPTEL logos at the top. The main title is "Transfer Learning: Image classification with VGG 16". On the left, a text box contains the following bullet points:

- In 2014 the VGG-16 network came in second place in the ILSVRC competition.
- It is a 16 layers network with a total of 138 Million parameters
- trained for two weeks on 4 NVIDIA GPUs
- Pretrained versions of the VGG-16 model are still often used as a feature extractor for other networks.

On the right, a diagram illustrates the VGG-16 architecture. It shows an input layer followed by two main stages of convolutional layers. The first stage consists of two layers of 3x3 convolutions with 64 filters each, followed by a max pooling layer. The second stage consists of two layers of 3x3 convolutions with 128 filters each, followed by a max pooling layer. This is followed by three layers of 3x3 convolutions with 256 filters each, and another max pooling layer. The final layer is a 3x3 convolution with 256 filters. The output is then passed through two fully connected layers of 2048 nodes each, and finally a softmax layer for classification. A legend at the bottom right identifies the symbols for convolution+ReLU, max pooling, fully connected+ReLU, and softmax.

Now, transfer learning I just want to show you some examples of those model which we use in transfer learning for image classification there is what called the VGG 16 and there is also VGG 19.

So, this model is good for feature extraction. So, people they usually use it in all that big models as well, you can see it can serve as a backbone for some other models the essence is that if you are dealing with a task that need feature extraction. So, this is good to work to help you extract our features.

(Refer Slide Time: 15:16)

The slide is titled "Transfer Learning: Image classification with RESNET-50". It features two logos at the top: NSM on the left and NPTEL on the right. The main content includes a diagram of a residual block on the left, showing an input x passing through a "weight layer" and a "relu" layer to produce $F(x)$. This is then added to the original input x via an "identity" connection, resulting in $F(x) + x$. Below this is a URL: <https://www.pngfind.com/2101210101.png>.

The central part of the slide shows a comparison of two neural network architectures: "Plain" and "ResNet". The "Plain" architecture is a long, linear sequence of layers. The "ResNet" architecture is significantly shorter, consisting of several residual blocks. Below the architectures is a list of bullet points:

- The ResNet-50 network was far deeper and more accurate than the VGG-16 network.
- 2015 Microsoft Research, 50 Layers, 23M params.
- Skip connections were used liberally in the ResNet-50 architecture that won the 2015 ImageNet competition

A URL <https://www.pngfind.com/2101210101.png> is also present at the bottom right of the diagram area.

There is also the RESNET this model this consists of 50 layers the RESNET 50 we have RESNET 50 we have RESNET 18 we have RESNET 34. So, this is example RESENT 50 it makes use of skip connections that is what it makes use of skips connections and it perform better in terms of future attraction than VGG 16.

So, you can also look this hub because this is important, because before you can use this type of model you need to understand the flow out the data format like I have explained before and how the model works internally.

(Refer Slide Time: 15:57)

The slide is titled "Application Focus". It features two logos at the top: NSM on the left and NPTEL on the right. The main content is a list of three items, each preceded by a square checkbox:

- NVIDIA TAO Toolkit
- YOLOv4
- YOLOv5

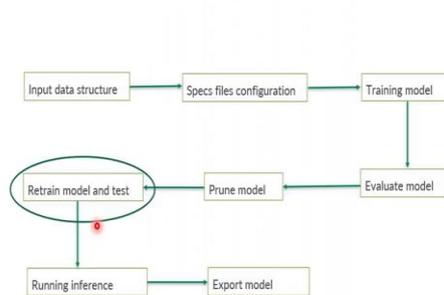
So, our application focus for today so as not to waste time will be NVIDIA TAO tool kit there is also YOLO v4 there is YOLO v5 we do not know if there is not we might not be able to cover YOLO v 4, but I would try as much as possible to also to cover YOLO v5.

But YOLO v5 we want to cover computer vision here now with NVIDIA TAO tool kit we want to cover image classification. So, with this object for YOLO v5 for computer vision for object detection because we might not be able to use camera direct because I am doing using SSH. Let us see how it goes? But we will finally way around that one.

(Refer Slide Time: 16:44)



Overview of TAO image classification phase



So, let us proceed now overview of tao image classification phase. So, if you want to use NVIDIA TAO for your transfer learning to achieve image classification tasks the first thing you need to do is to work to check your input data structure. So, what I mean by this is your data format rather that ok. You must have a particular pretrained model in mind you want to use let us say for example RESNET so you must know ok.

Now, the data my raw data that I want to use it must be in the particular format which RESNET can which can serve as input into RESNET model. So, you consider that after that there is what we call the specs file that is the configuration file, that configuration file is for the pre trained model you have to configure it to suit your new data that you your new data that you are working on you have to configure it to suit that.

So, there are phases in that I we explain into details as we proceed then after you configure your specification file, then you start training your model. After training your model you can proceed to evaluate the model and when you evaluate the model you can prune the model which I have explained before. So, while you prune the model the size is reduced, then you need to retrain if there is an accuracy loss while you prune the model you retrain and test. When you retrain and test you check the accuracy.

If the accuracy is not satisfactory you keep retraining and when it is satisfactory you proceed to inferencing. So, you inference you run your inference and when the inference is good you can now export your model and use it for what you tend to use it for.

(Refer Slide Time: 18:35)

The slide features the NSM logo on the left and the NPTEL logo on the right. The main title is "TAO Application with Resnet 18 model". On the left side, there is a list of tasks and datasets:

- Image Classification Task
- Input Dataset
 - ✓ Kitti
 - ✓ TFrecord

The central part of the slide shows a file explorer window with a tree view of a dataset. The root folder is "data", which contains subfolders for "train" and "val". Each of these subfolders contains a "tfrecord" file. Below the tree view, a grid of image thumbnails is displayed, representing the contents of the dataset folders.

So, let us consider the case of RESNET 18 that we want to use RESNET 18 format so RESNET 18. So, how do you do that now?

Now, first of all you prepare your data set let us say this is our data set we have it already it is and we have splitted it into test train and validation that they are all inside our images. So, want to do image classification. So, in each of these folder where you have test what you will have inside is the folder consisting of the image and the class. So, the image will be inside for example for bus all what would be here were images of bus then you your name the folder with the name of the class. So, each folder consists of distinct images not mixture; distinct images.

(Refer Slide Time: 21:04)



classification_spec.cfg file



```
Model_config {
  arch: "resnet_18"
  n_layers: 18
  # Setting these parameters to true to match the template downloaded from NCC.
  use_batch_norm: true
  all_projections: true
  freeze_blocks: 0
  freeze_blocks: 1
  input_image_size: "3, 224, 224"
}

Train_config {
  train_dataset_path: "/workspace/llt-experiments/data/spilt/train"
  val_dataset_path: "/workspace/llt-experiments/data/spilt/val"
  pretrained_model_path: "/workspace/llt-experiments/classification/pretrained_resnets/llt_pretrained_classification_resnets/resnet_18.hdf5"
  optimizer {
    top {
      lr: 0.1
      decay: 0.0
      momentum: 0.0
      nestorov: false
    }
  }
  batch_size_per_gpu: 64
  n_epochs: 80
  n_workers: 10
  preprocess_mode: "caffe"
  enable_random_crop: true
  enable_center_crop: true
  label_smoothing: 0.0
  mixup_alpha: 0.0
  # regularizer
  reg_config {
    type: "L2"
    scope: "Conv2D_Dense"
    weight_decay: 0.0005
  }
  # Learning_rate
  lr_config {
    step {
      learning_rate: 0.001
      step_size: 10
      gamma: 0.1
    }
  }
}

Eval_config {
  eval_dataset_path: "/workspace/llt-experiments/data/spilt/test"
  model_path: "/workspace/llt-experiments/classification/output/weights/resnet_180.tlt"
  top_k: 1
  batch_size: 32
  n_workers: 0
  enable_center_crop: true
}
```

So, this is the classification-spec.cfg file. So, this file what happens here is used for training. So, inside it you have the model config you have the train config and you have the evaluation config.

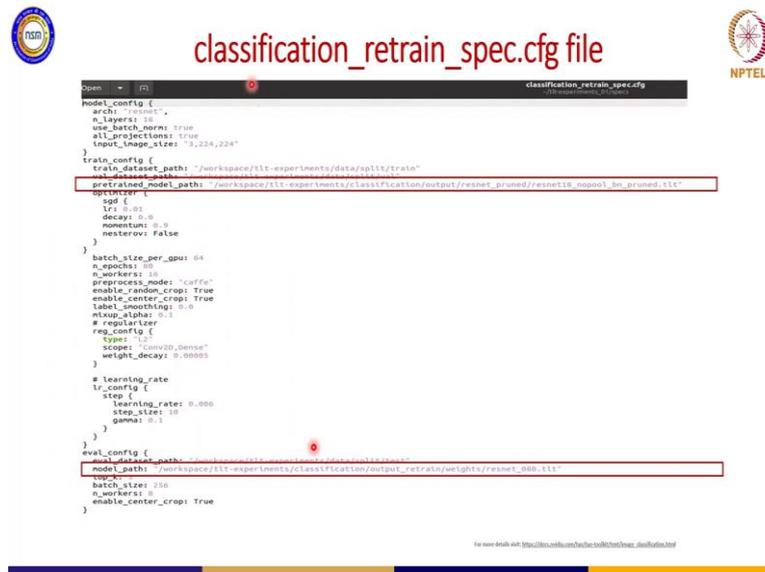
So, for the model config you specify the architect the architecture you want to use which is what we specified that is RESNET we want to use pre trained model pre trained model will be RESNET of what type 18 layers we want to use so the image ok. what is the input size? So, the input size for RESNET the dimension will be what you see three by 224 by 224.

Then we proceed to the configuration the train configuration. So, where would you get access to your trained data set? So, this is train_dataset_path. So, you have to specify the path where your data is and also your validation data path this is val_dataset_path. Where it is? And then your pre trained model the model you want to use to train this is our RESNET you specify the path as well. So, it is save as what resnet_18.hdf5.

Then you look at it ok the batch size per GPU, then the number of epochs which is what 80 you set that then the learning rates also you set that, then when you for the evaluation also the paths of your data set your evaluation data set that is the test data. So, you set the path as well like what we are seeing here, then the model path that is after you finish training where to save it. So, that I can use it for the evaluation you set the path as well because the model will be save in output, so you set the path as well and all that. And so

there are details about all of the features here, but I am just telling you the default ones which.

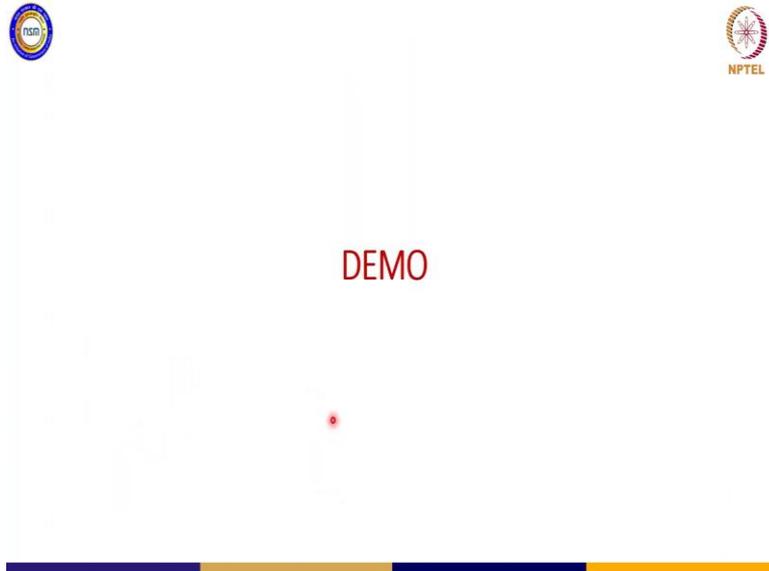
(Refer Slide Time: 23:02)



Then this is now the classification retrained specification file. So, it is similar to the classifications specification file that is the one we use for training. So, the only difference that you have there is the pretrained model paths, that is when you pretrained when you when your model has been trained and when it has been pruned. So, where are you going to get the model to be retrained?

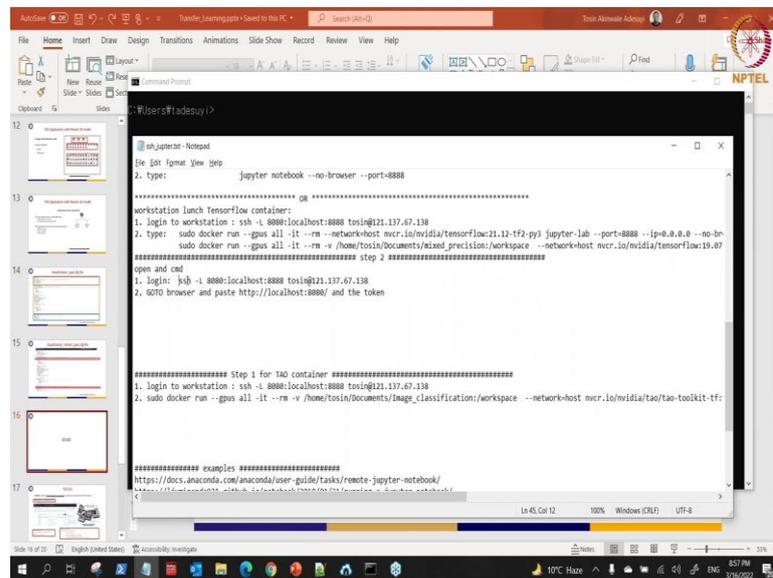
So, you specify the path this way it is also there I will show you the for that structure. So, that you do not get confused also when you want to evaluate as well this is the path. So, you set all these paths these are path that are on your systems. So, it is not compulsory that it must be what you are seeing here exactly. So, it is based on your folder structure on your personal laptop or workstation that you have.

(Refer Slide Time: 24:06)



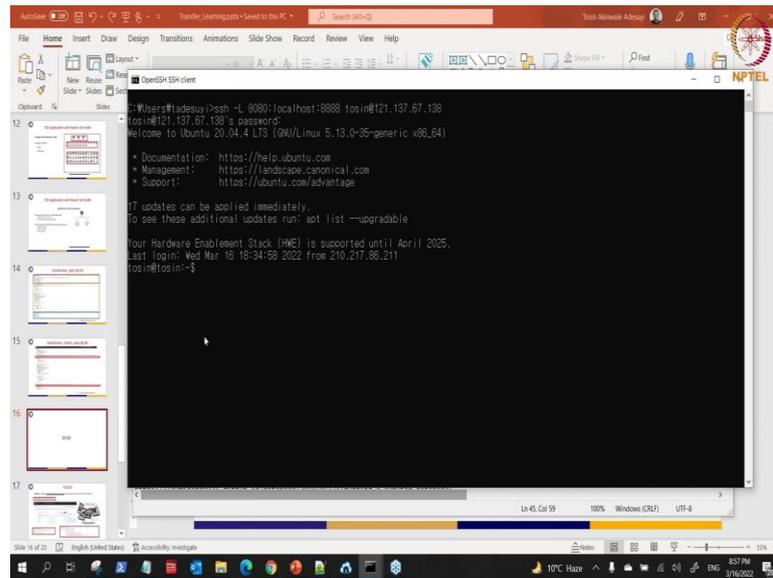
So, quickly let us jump into a demo for that. So, that we know what we can do about that.

(Refer Slide Time: 24:18)



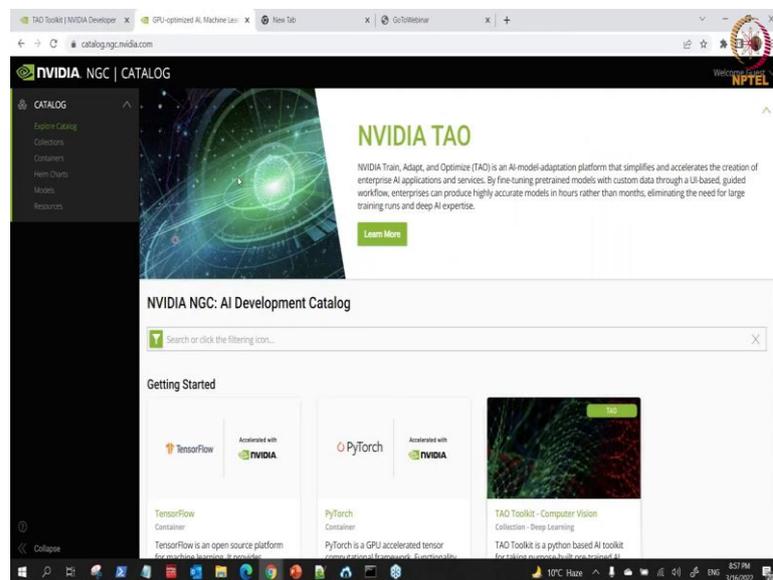
So, what I would do now is I have to connect through my I have to connect to my workstation ok.

(Refer Slide Time: 24:35)



SSH connect to my workstation ok good connected now. So, the next thing to do is to so before you can use the NVIDIA TAO for your transfer learning you need to pull the container.

(Refer Slide Time: 25:10)



So, let me show you where you can pull the container here. So, you come to this place this is the catalogue dot NGC like I show in the last image dot NVIDIA dot com.

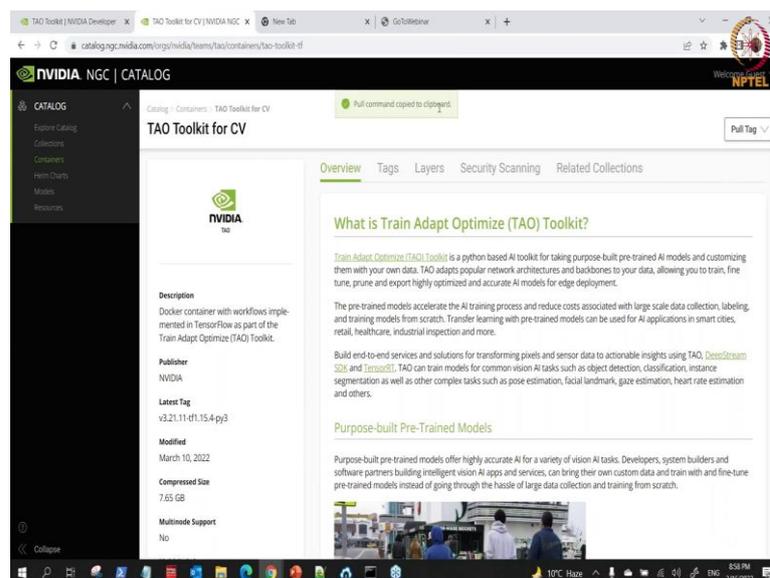
(Refer Slide Time: 25:24)



So, when you get a so what you do is you click you search you can search just type TAO and when you type on TAO you can see this is. If you can see what is here this is TAO toolkit for CV, CV means computer vision.

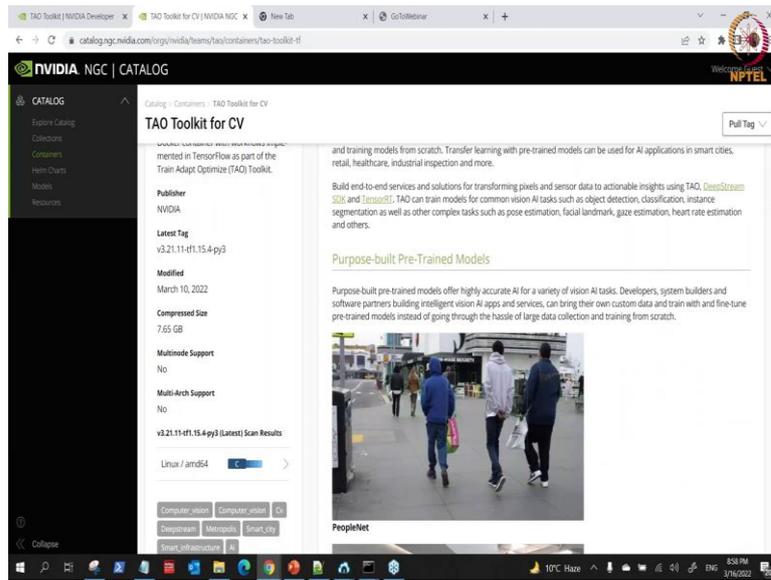
So, there is another TAO for conversational AI that is what you want to deal with. So, what we are dealing with here is this.

(Refer Slide Time: 25:45)

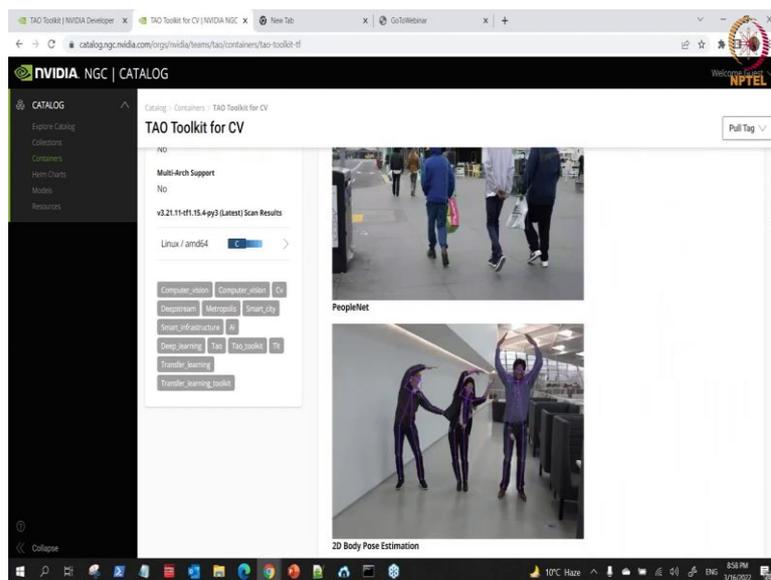


So, we go then these are details that you can read.

(Refer Slide Time: 25:49)



(Refer Slide Time: 25:50)



About it and you see what it can do.

(Refer Slide Time: 25:53)

The screenshot shows the NVIDIA NGC CATALOG interface for the TAO Toolkit for CV. The table lists the following models:

Model Name	Network Architecture	Number of classes	Accuracy	Use Case
TrafficCamNet	DetectNet_v2-ResNet18	4	83.5% mAP	Detect and track cars
PeopleNet	DetectNet_v2-ResNet18	3	80% mAP	People counting, heatmap generation, social distancing
PeopleNet	DetectNet_v2-ResNet18	3	84% mAP	People counting, heatmap generation, social distancing
DashCamNet	DetectNet_v2-ResNet18	4	80% mAP	Identify objects from a moving object
FaceDetectNet	DetectNet_v2-ResNet18	1	96% mAP	Detect face in a dark environment with IR camera
VehicleTypeNet	ResNet18	20	91% mAP	Classifying car models
VehicleTypeNet	ResNet18	6	96% mAP	Classifying type of cars as coupe, sedan, truck, etc
PersonSegmentNet	MaskRCNN-ResNet50	1	85% mAP	Creates segmentation masks around people, provides pixel
PersonSegmentNet	UNET	1	92% MDU	Creates semantic segmentation masks around people. Filters person from the background
License Plate Detection	DetectNet_v2-ResNet18	1	98% mAP	Detecting and localizing License plates on vehicles

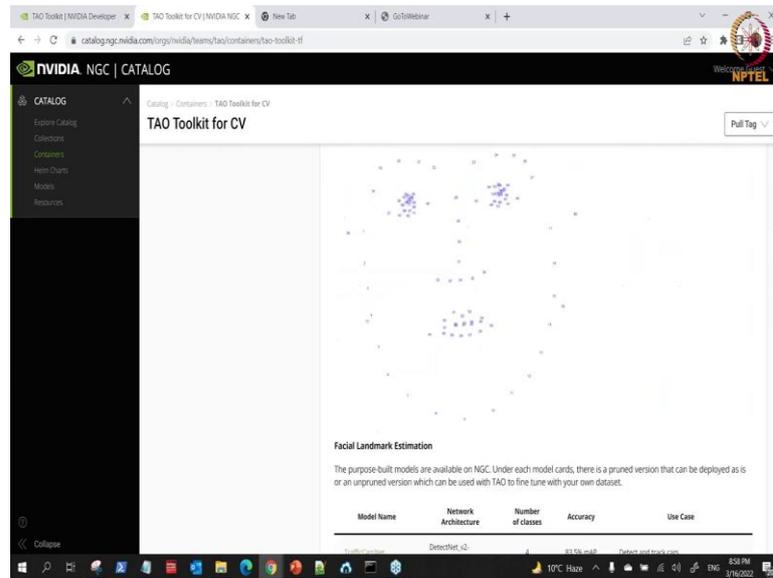
(Refer Slide Time: 25:55)

The screenshot shows the NVIDIA NGC CATALOG interface for the TAO Toolkit for CV. The table lists the following models:

PersonSegmentNet	MaskRCNN-ResNet50	1	85% mAP	Creates segmentation masks around people, provides pixel
PersonSegmentNet	UNET	1	92% MDU	Creates semantic segmentation masks around people. Filters person from the background
License Plate Detection	DetectNet_v2-ResNet18	1	98% mAP	Detecting and localizing License plates on vehicles
License Plate Recognition	Tuned ResNet18	36(U)/6(RC)	97% (U)/99% (R)	Recognize License plates numbers
Gaze Estimation	Four branch AlexNet based model	N/A	6.5 RMSE	Detects person's eye gaze
Facial Landmark	Recombinator networks	N/A	6.1 pixel error	Estimates key points on person's face
Heart Rate Estimation	Two branch model with attention	N/A	0.7 BPM	Estimates person's heart rate from RGB video
Gesture Recognition	ResNet18	6	0.85 F1 score	Recognize hand gestures
Emotion Recognition	5 Fully Connected Layers	6	0.91 F1 score	Recognize facial Emotion
FaceClust	DetectNet_v2-ResNet18	1	85.3 mAP	Detect faces from RGB or grayscale image
2D Body Pose Estimation	Single shot bottom-up	18	-	Estimates key joints on person's body

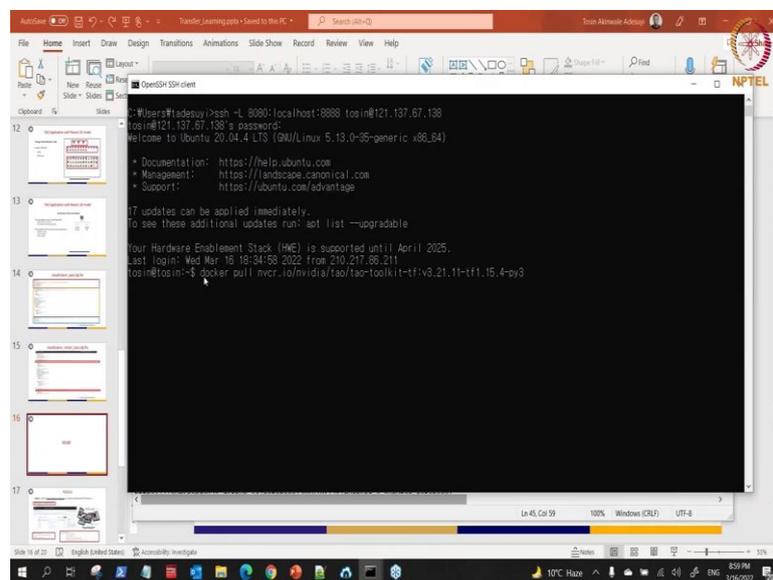
So, and this are the list of all the models that are inside the TAO up there.

(Refer Slide Time: 25:58)



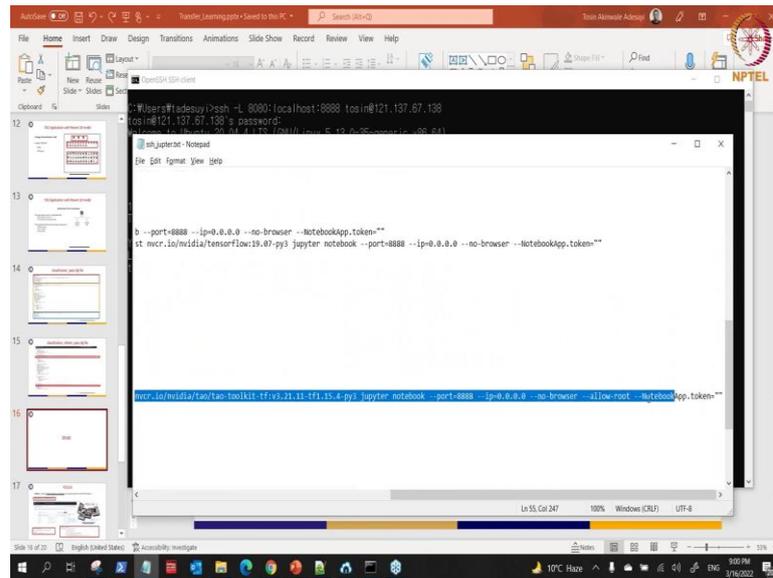
So, you come to this place you click on this tag it will come down. So, you select one of them. So, if I select this you can see command is pulled up.

(Refer Slide Time: 26:18)



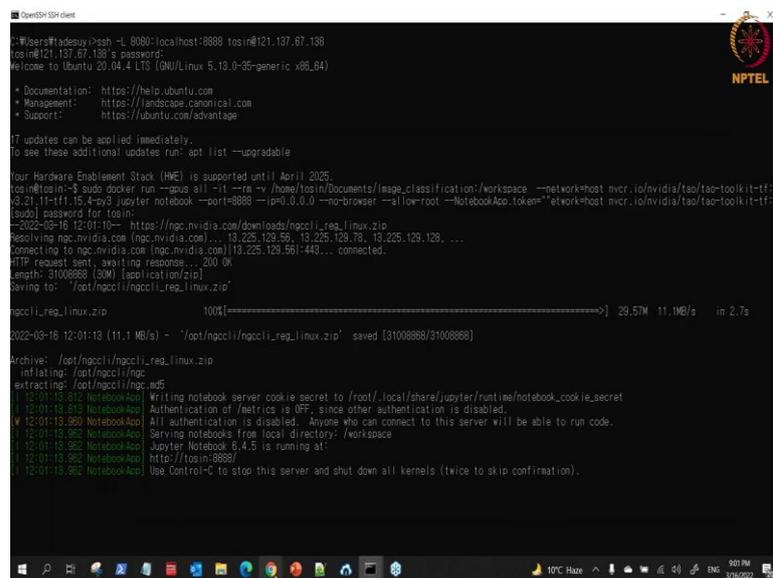
So, you can now go back to and you can come here and say docker pull this I have already pulled this already I have done that. So, once you pull this container then next thing to do is to what is to run the container you run the container. So, you do not need to cram anything here everything is explanatory, let me see you can also install them somewhere. So, that you do not because they are very long I must say.

(Refer Slide Time: 26:55)



So, I will explain this part to you will explain the parts coming with..

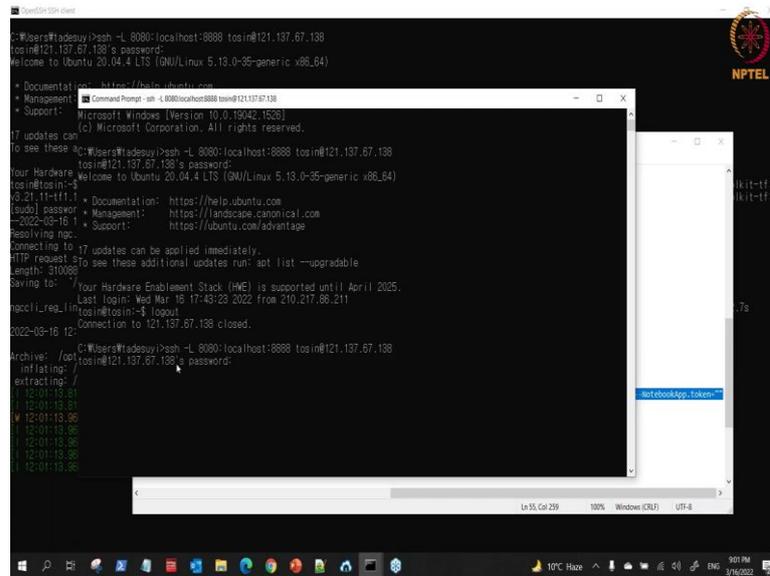
(Refer Slide Time: 27:26)



So, I come here then I paste it here then let me expand this what I have here. So, what you can see here is that ok. So, this is pseudo docker you can run the GPU all this use all the GPU that are available, then make it in interactive mode you remove it when you are not in use this is my project work part. What I will be using this is the path for my projects that are mapping it into the workspace this a folder inside the container.

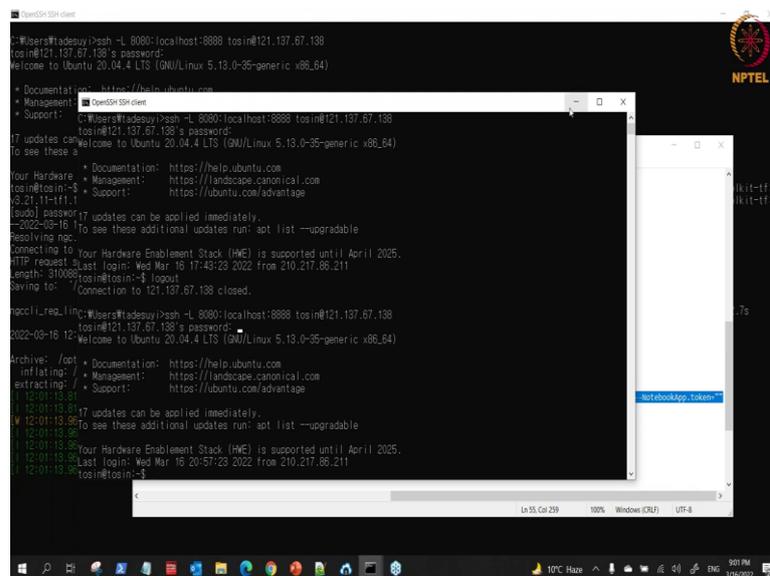
And I am calling the container here what you can see this is the name of the container to be launched inside a, we launch inside the notebook with these ports. So, once I run that see so it is going on and so yes the notebook is running out. So, what I need to do next is.

(Refer Slide Time: 28:41)



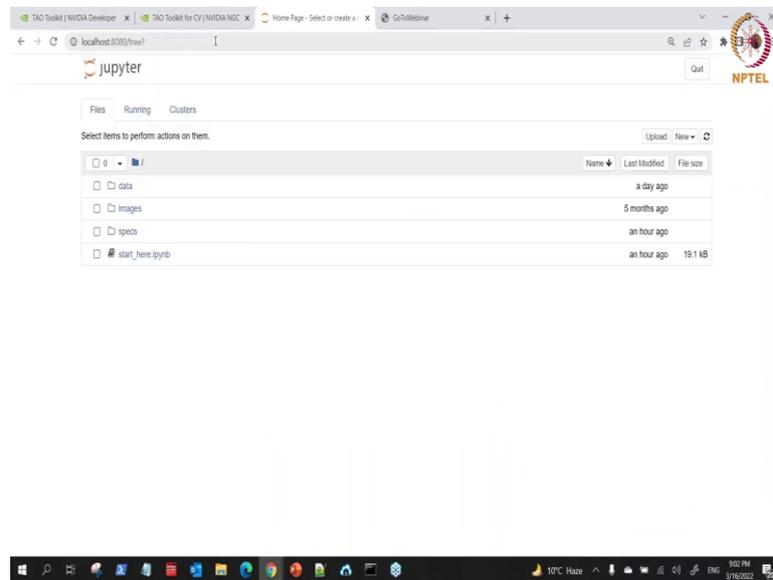
Because I am connecting remote leaders while I am doing all this configuration for your own personal system, you might not need to go do this long go too long this way.

(Refer Slide Time: 28:56)



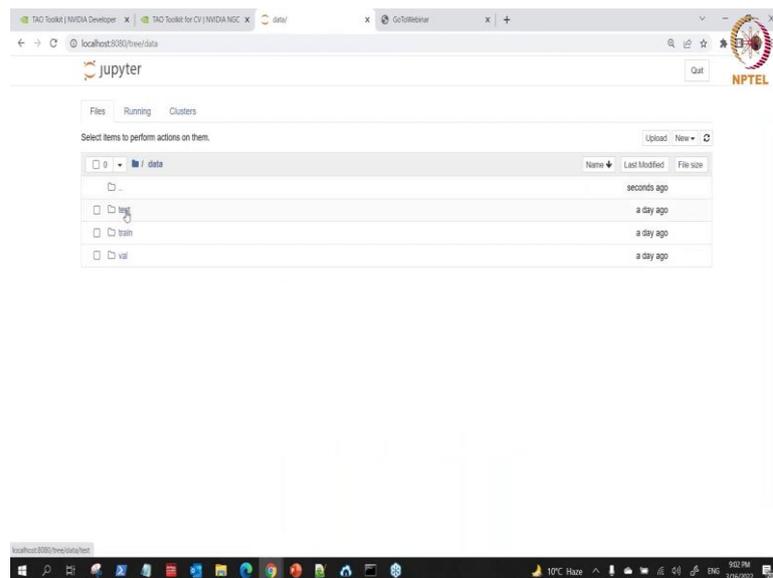
So, if you are working directly on your workstation you do not need to do this part. So, what happen is after you have launched your container like this so you just go to your browser.

(Refer Slide Time: 29:15)



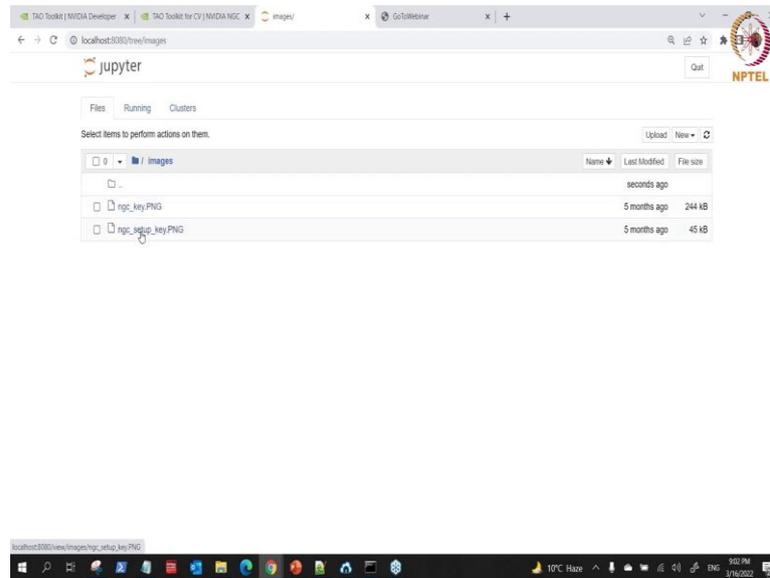
So, you go to your browser then you next thing you do is what you just say local host and this and it will run. So, this is the project path. So, what I have here is my data.

(Refer Slide Time: 29:23)



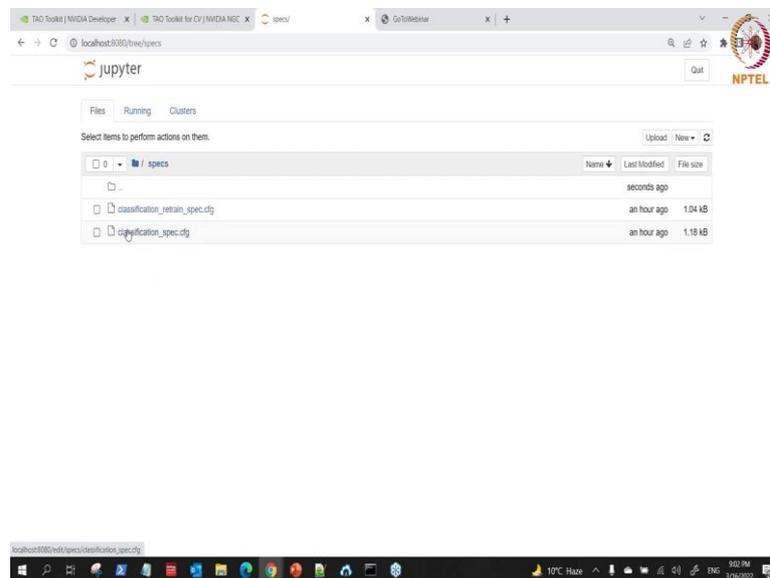
This is the data I told you to test val train and validation these images there is nothing there is just an image that I used to for this notebook.

(Refer Slide Time: 29:35)



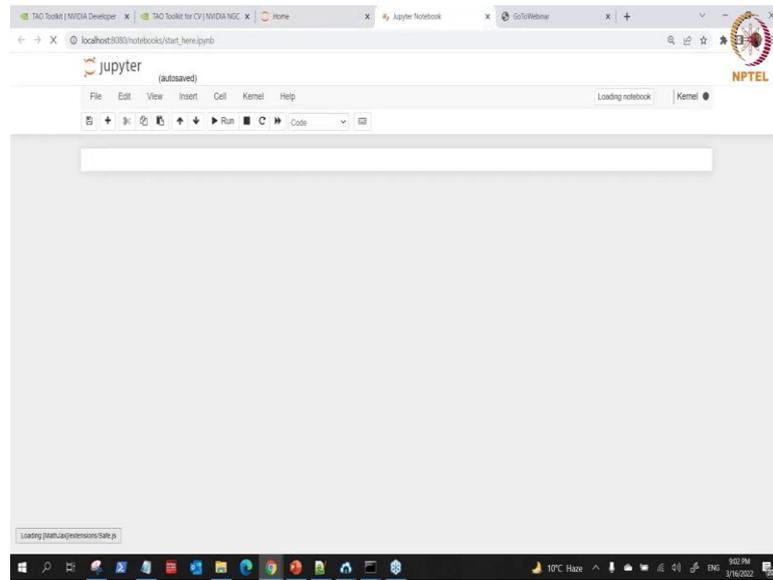
That is what is there it is not a paths of the project. So, it is just what I use for this notebook.

(Refer Slide Time: 29:43)



Then there is a specification file which I told you this is the one for training and this is the one for retraining.

(Refer Slide Time: 29:52)



So, we get back here and what click on this is our notebook.