# **Pretrained models in CNN | ImageNET Dataset | ILSVRC | Keras Code**

### **Introduction to Pre-trained Models**

* A **pre-trained model** is a Convolutional Neural Network (CNN) architecture or other neural network architecture that has been created and trained by someone else on a specific dataset.
* These models are often so effective that they can be **reused for different problems**.
* This video aims to explain the concept behind pre-trained models and demonstrate how to use them in **Keras**.
* Understanding pre-trained models is crucial for **Transfer Learning**, a very important topic that will be covered later.

### **Why Use Pre-trained Models?**

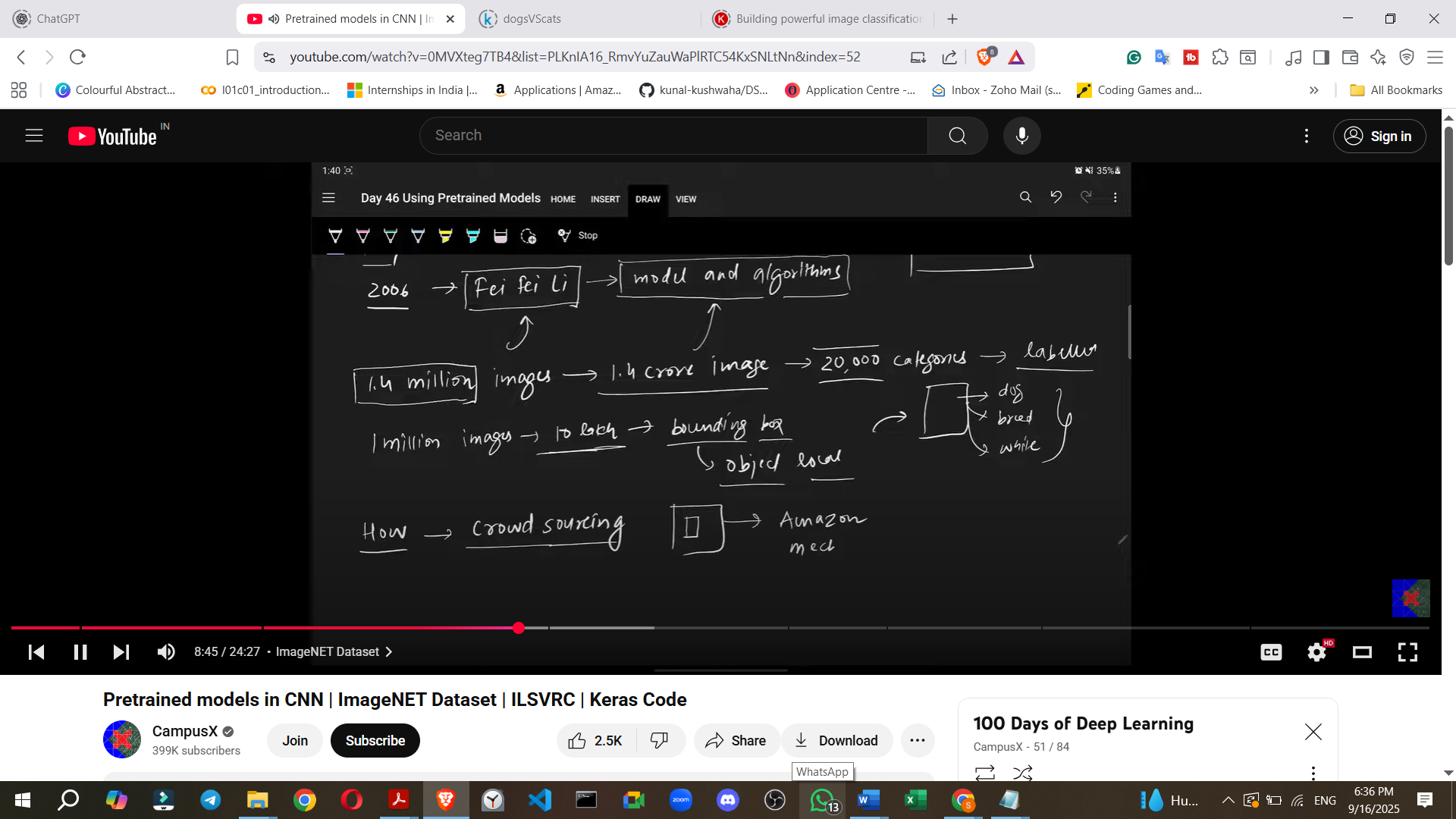
There are two primary reasons why one should use pre-trained models instead of building and training custom models.

1. **Deep Learning is Data-Hungry**:  
   * Deep learning models require a **large amount of data** to perform well.
   * For **image classification problems**, this means needing a vast quantity of **labeled image data**.
   * While images can be scraped from sources like Google Images, the significant challenge lies in **manual labeling**.
   * For example, labelling 10,000 downloaded images manually is a **tedious, time-consuming, and costly task**, potentially requiring hiring staff, leading to financial loss for companies.
   * Future developments, like companies opening up their data, might ease this, but currently, it remains a difficult process.
2. **Training Deep Learning Models Takes Time**:  
   * Training a model on a large dataset can take **hours, days, or even weeks**.
   * This slow training process can hinder the overall model building timeline.
   * The solution is to **leverage pre-existing, well-trained models**, thereby avoiding the need for extensive data collection and the long training process.
   * Using pre-trained models means **less or even no data is required** for training, and **no time is spent on training**.

### **Origin of Pre-trained Models: ImageNet Dataset**

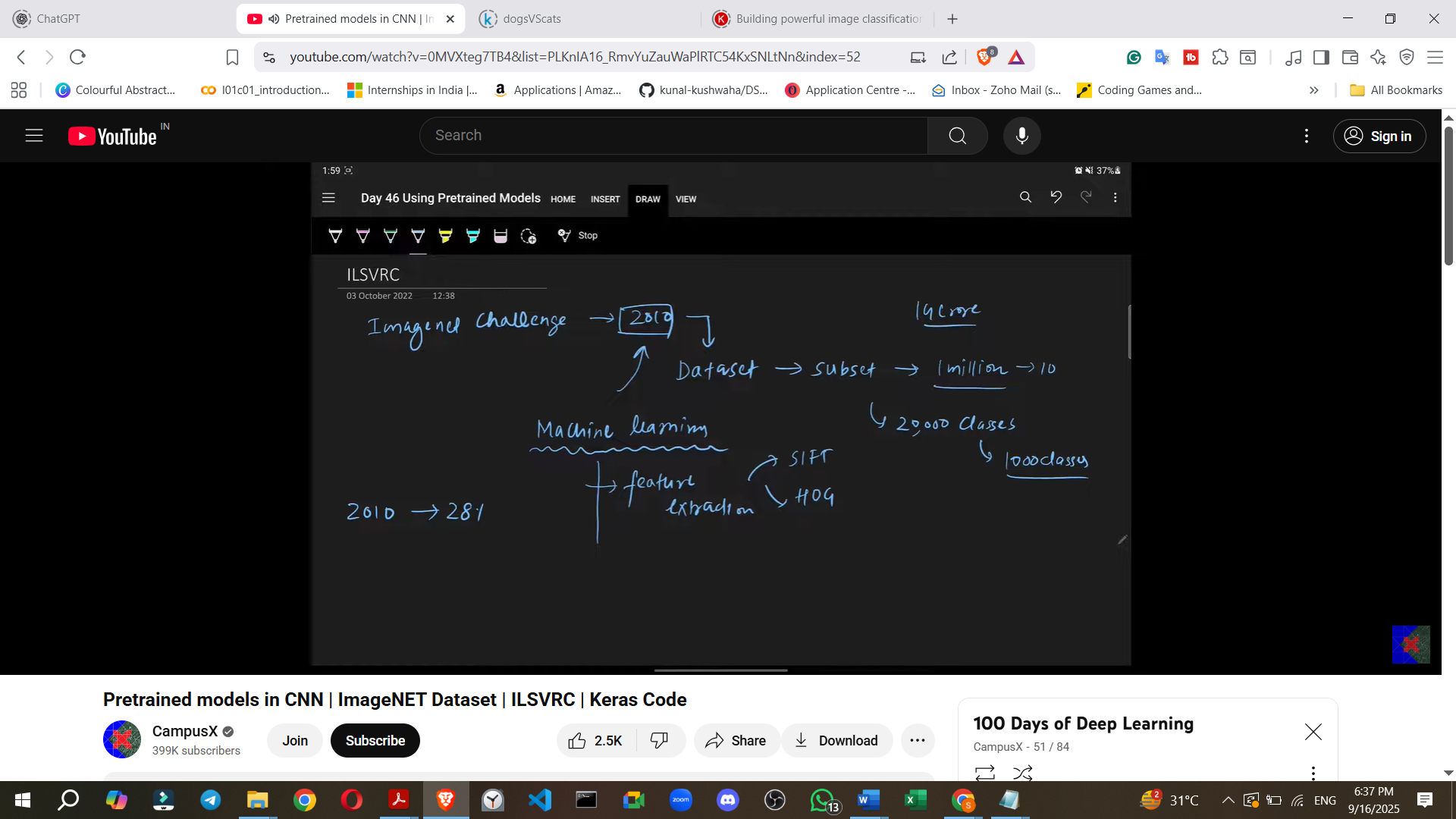
To understand pre-trained models, it's essential to know about the **ImageNet dataset**.

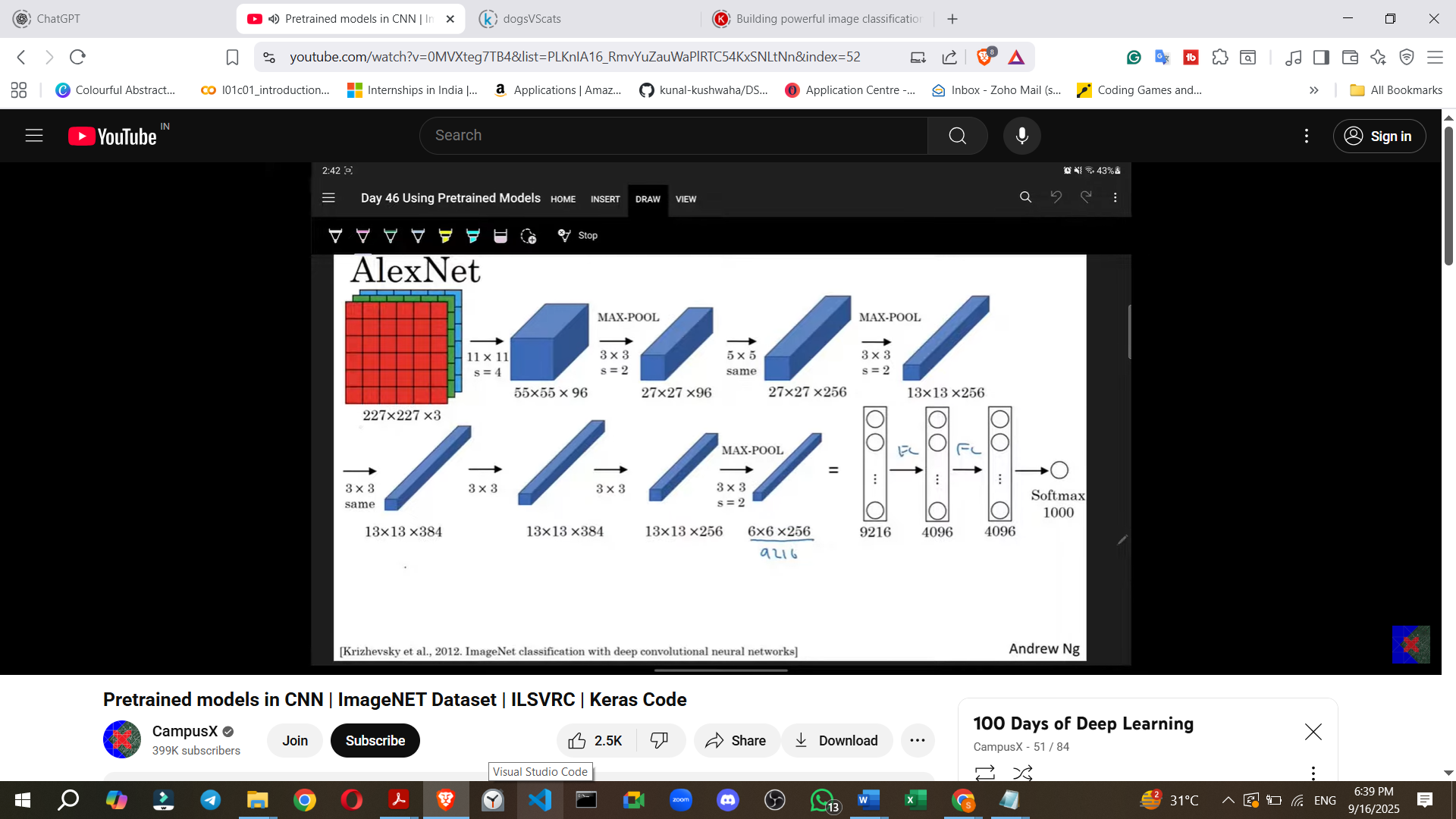
* **What is ImageNet?**
  + ImageNet is a **visual database of images**.
* **Why and How was ImageNet Created?**
  + In 2006, deep learning research primarily focused on **model building and algorithm development**.
  + Professor Fei-Fei Li recognised that for deep learning to advance, **large and high-quality datasets** were absolutely necessary.
  + Starting in 2006, she, along with a professor who created WordNet, began building the **ImageNet database**.
  + It comprises **1.4 million (1.4 crore) images**.
  + The dataset includes images across **20,000 categories** of daily household items and commonly observed objects.
  + All images are **well-organized and labeled**.
    - Labels include specific details, e.g., for a dog image, it would specify the breed and a visual description.
    - It also features **bounding box labeling** for **object localization**, where a box highlights the object's location within an image.
  + The database was constructed using **crowd-sourcing**, employing a service called **Amazon Mechanical Turk** to gather human input on image content.
* **Impact**: ImageNet fundamentally **changed the future of deep learning**.



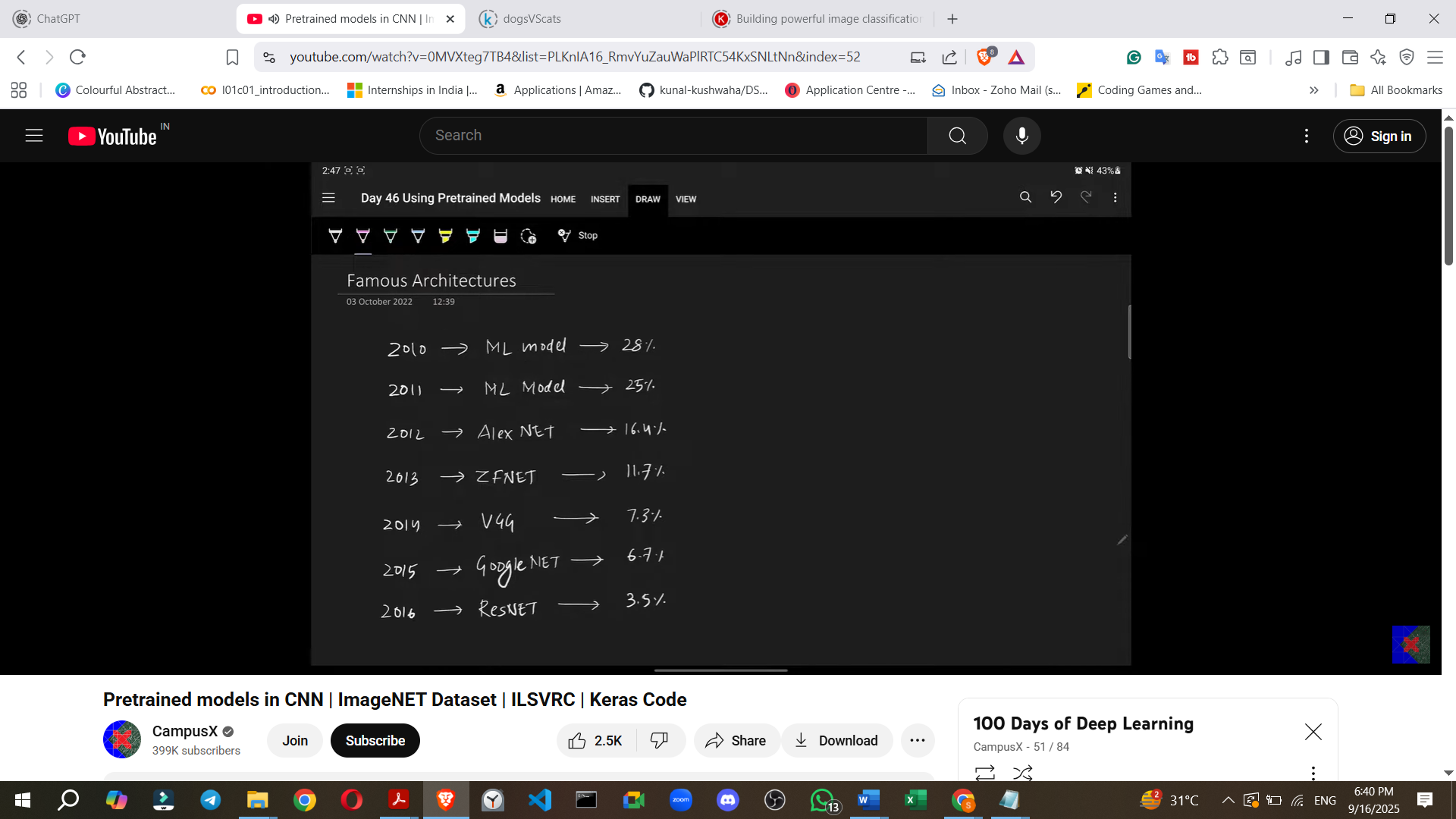
### **The ImageNet Challenge (ILSVRC)**

After creating the ImageNet database, researchers decided to use it for a **competition**.

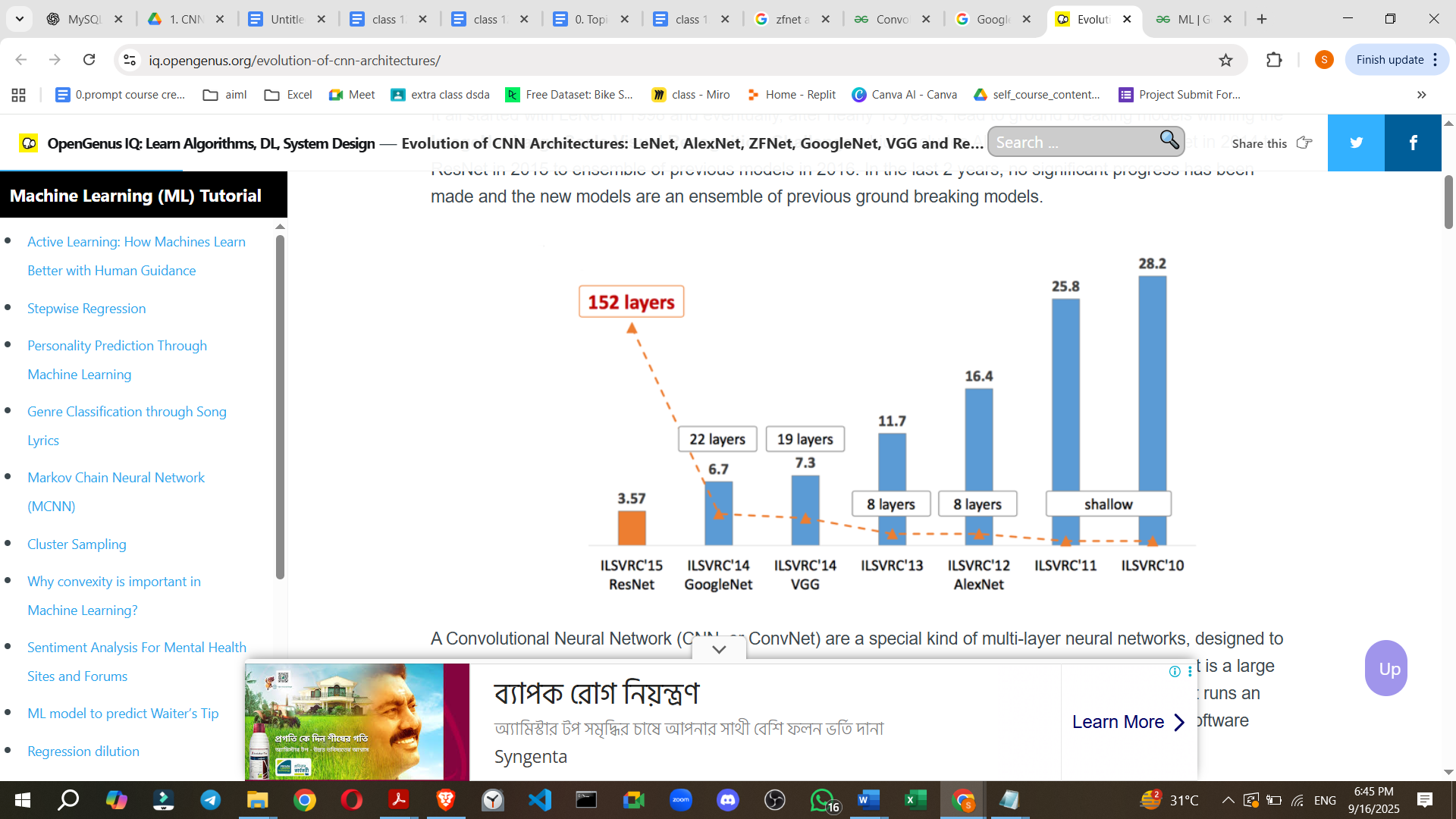
* **What is ILSVRC?**
  + **ImageNet Large Scale Visual Recognition Challenge** (ILSVRC), or simply **ImageNet Challenge**, started in **2010**.
  + Its goal was to identify and highlight the **best image classification models**.
* **Dataset Used**:
  + The challenge used a **subset** of the original ImageNet database.
  + This subset contained **1 million images** and **1000 classes**, a reduction from the original 20,000 classes to simplify complexity.
* **Early Years (2010-2011)**:
  + The initial competitions featured **Machine Learning-based models**.
  + Participants relied on **manual feature extraction**.
    - 
  + **Error rates** were relatively high:
    - **2010**: Winner's error rate was **28%** (28 mistakes out of 100 images).
    - **2011**: Error rate improved slightly to **25%**.
* **The Deep Learning Revolution (2012): AlexNet**
  + The real shift occurred in **2012** with the introduction of **AlexNet**.
  + AlexNet was the **first Deep Learning (CNN) model** to win the competition.
  + It incorporated **ReLU activation layers**.
  + AlexNet dramatically reduced the error rate to **16%**, a significant improvement of over 10% compared to the second-place model.
  + This victory **captured the attention of the entire tech world**, making people realise the immense potential of CNNs and deep learning.
* **AlexNet Architecture (Example)**:



* + Input images were **227x227 pixels and coloured**.
  + The architecture included:
    - First layer: **96 filters of 11x11**.
    - Max pooling (3x3 size, stride 2).
    - Subsequent layers with **5x5 filters (256)**, more max pooling, and **384 filters**.
    - Finally, three **fully connected layers** with 9216, 4096, and **1000 units** (corresponding to the 1000 classes).
* **Continued Deep Learning Dominance (2013-2016)**:
  + After AlexNet, deep learning models consistently won the challenge.



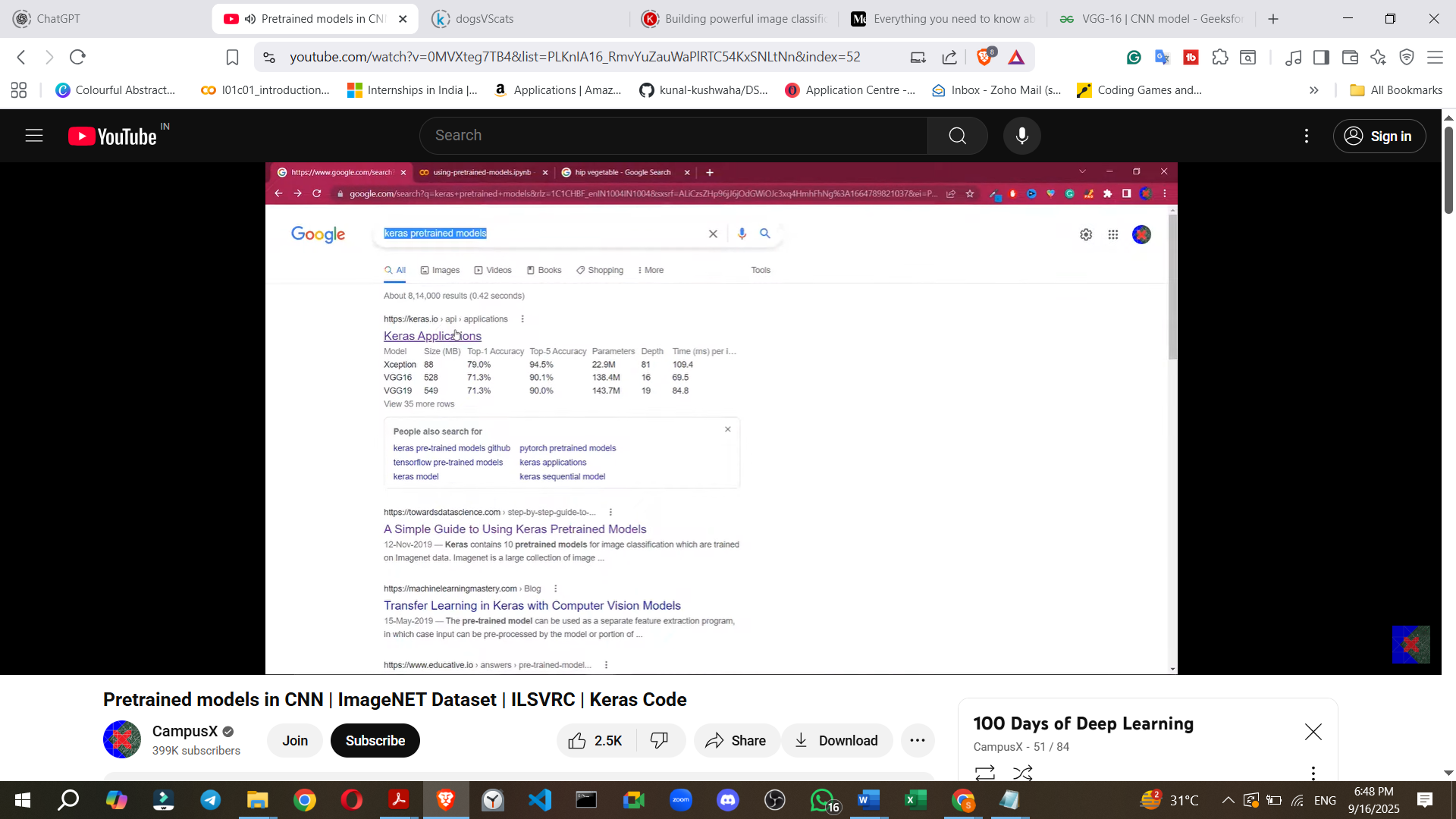
* + **Error rates continued to decrease** rapidly:
    - **2013**: **ZFNet** achieved **11.7%**.
    - **2014**: **VGGNet** (Visual Geometry Group) achieved **7.3%**. VGGNet became very famous and is widely used.
    - **2015**: **GoogleNet** achieved **6.7%**.
    - **2016**: **ResNet** (Residual Network) achieved an unbelievable **3.5%**.
  + **ResNet surpassed human-level performance**: The human error rate for this dataset is approximately 5%, meaning ResNet performed better than humans.
  + A common pattern observed was the **addition of more layers** and increased complexity in CNN models over the years, leading to a reduction in error rates.
  + It is recommended to study these architectures (AlexNet, ZFNet, VGG, GoogleNet, ResNet) and attempt to implement them in Keras as an exercise.

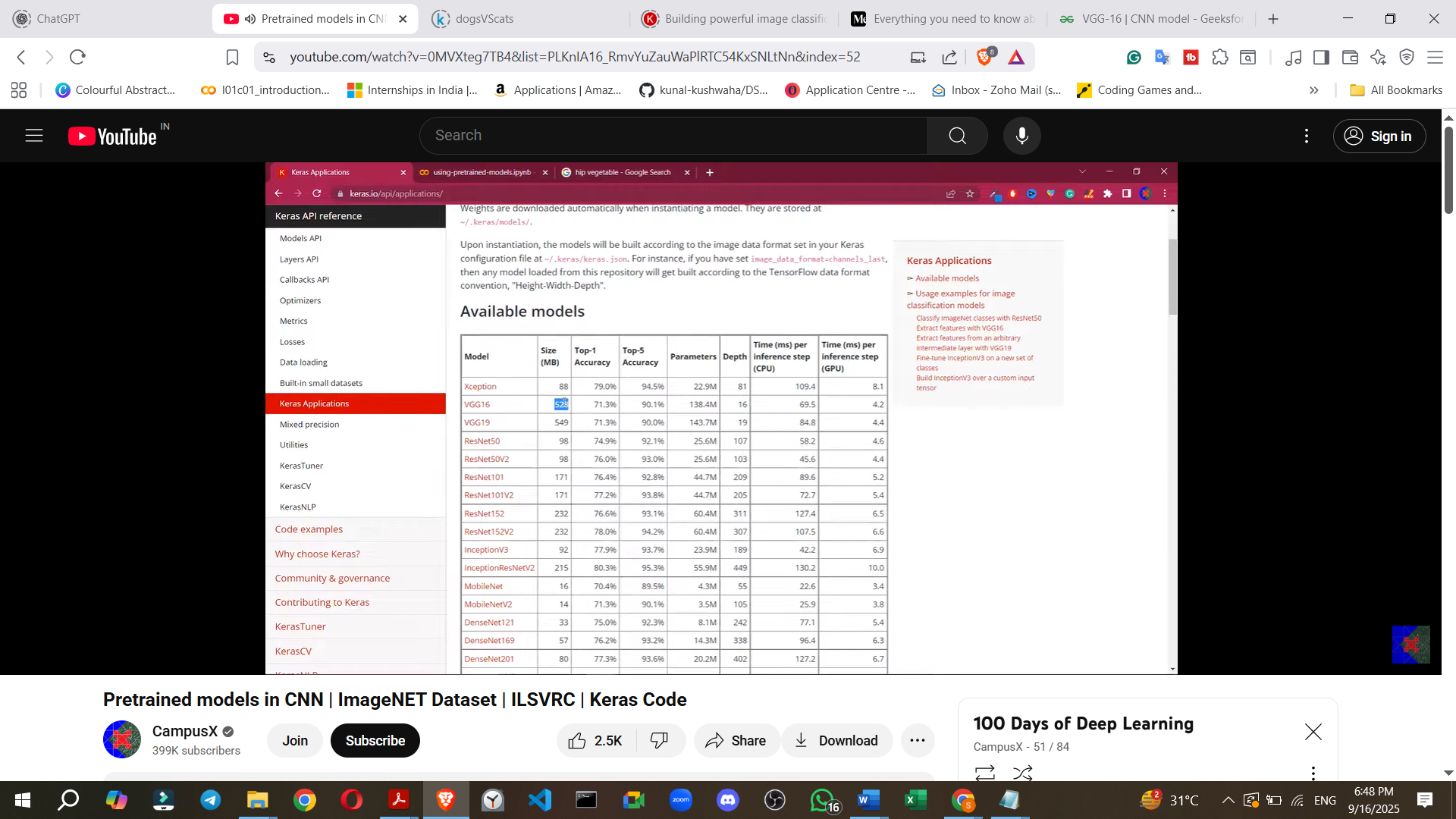


### **Using Pre-trained Models in Keras**

Keras provides a convenient way to access and use many pre-trained models.

* **Benefits Revisited**:
  + **No need for extensive training data**; can work with very little or even zero data.
  + **No time spent on training**, as the models are already trained.
* **Keras Applications Library**:
  + Searching "Keras pre-trained models" leads to the keras.applications documentation, which lists available models.
  + Commonly available models include: **Xception, VGG16, VGG19, ResNet50, ResNet50V2, InceptionV3, MobileNet**, among others.
  + The documentation provides details for each model, such as:
    - **Size** (e.g., VGG16 is ~528 MB, ResNet50 is ~98 MB). The large size is due to the stored weights (parameters), e.g., VGG16 has ~13.4 million parameters.
    - **Number of parameters and layers**.
    - **Top-1 Accuracy**: The model's ability to make an exactly accurate prediction.
    - **Top-5 Accuracy**: The model's ability to have the correct prediction among its top five outputs.
    - **Inference time** on CPU and GPU.
  + **Famous Architectures**: Xception, InceptionV3, and MobileNet are also prominent.





* **Example: Universal Classifier using ResNet50**
  + The goal is to build a classifier that can predict the content of any image.
  + **Steps**:
    - **Import** necessary modules from Keras.
    - **Load the ResNet50 model**: Crucially, specify weights='imagenet' to load the weights pre-trained on the ImageNet dataset.
    - **Load and preprocess an image**: Example images used were a dog, bread, tomato, and chair. The image is loaded, converted to a batch, and preprocessed.
    - **Make predictions**: Use model.predict(X) where X is the preprocessed image.
    - **Decode predictions**: Use keras.applications.resnet50.decode\_predictions to interpret the model's output.
  + **Results**:
    - **Dog image**: Correctly predicted "Labrador retriever" and "Golden retriever" (specific breeds).
    - **Bread image**: Predicted "French loaf".
    - **Chair image**: Predicted "dining chair".
    - **Tomato image**: Showed some confusion, predicting "strawberry" and "hip" (a vegetable-like item). This illustrates potential slight misclassifications but generally shows good understanding of image content.
  + **Conclusion of Example**: Without building or training any custom architecture, a **"universal classifier"** was created by simply using a pre-trained ResNet50 model.

### **Future Learning**

* The concepts of pre-trained models are foundational and will be reused in **Transfer Learning**, which is the next topic to be covered.