**Open Source Models with Hugging Face – Younes Belkada, Marc Sun, Maria Khalusova**

Thanks to open-source software, if you want to build an AI application, you might be able to grab an image recognition component here, and an automatic speech recognition model there, and then LLM somewhere else, and then string them together very quickly to build a new application.

Hugging Face has been transformative for the AI community in terms of making it easy for anyone to do this by making many open-source models easily accessible. This has been a huge accelerator for how many people build AI applications.

In this course, you will know how to do this, and build cool applications yourself, possibly faster than you might have previously imagined would be possible.

For example, you use models to perform automatic speech recognition (ASR) to transcript speech into text. And also, text to speech (TTS) model to go the other way to convert text into audio. These models combined with LLM give you the building blocks you can use to build your own voice assistant.

You will also see how to use Hugging Face’s transformer library to quickly pre-process as well as post-process outputs of machine learning models. For example, pre-processing audio like controlling the audio sampling rate in the ASR or TTS examples mentioned before this, as well as pre-process or post-process data such as images and text.

The notion of grabbing open-source components to build something quickly has been a paradigm shift in how AI applications are built.

In this course:

* You will create your own chatbot with open source LLMs.
* You will use an open source LLM from Meta, the same code can apply to more powerful open source LLMs when you have access to more powerful hardware.
* You will use open-source models to translate text from 1 language to another, summarize documents, and calculate sentence embeddings in order to compare similarities between 2 sentences.
* You will use transformers for processing audio. It needs to know what audio task it needs to perform when you ask for It, it wakes up when you say its name – Classification. It converts your speech to text to look up your request – Automatic speech recognition. It replies to you – Text to speech.
* You will classify arbitrary sounds, transcribe speech recordings, and generate speech from text.
* You will learn how to detect objects in images and segment images into regions called semantic areas. For example, you can apply this code to detect that a puppy exists in an image and also segment the part of the puppy that makes up its ears.

After you have learnt to handle text audio and image tasks, you can combine these models in a sequence to handle more complex tasks. For example, if you want your app to help someone with a visual impairment by describing an image to them, how could you implement that.

In this course, you will apply object detection to identify the objects, image classification to describe those objects in text, and then speech generation to narrate the names of those objects. You will also use a model that can take in more than 1 data type as input – multimodal models. You will also use the Gradio library to deploy an AI application to Hugging Face spaces so that anyone can use your application to perform tasks by making API calls to the internet.

The goal of these examples isn’t just for you to be able to build these specific examples, it is so that you will learn about all these building blocks so that you will be able to combine them yourself into your own unique applications.

**Lesson 1: Selecting Models**

Hugging Face Hub: An open platform that hosts models, datasets, and machine learning demos that are called Hugging Face Spaces.

**Models**

Model Page:1) Filter by task, language, permissive licence (allows you to use model for commercial use). 2) Sort by trending, most downloads

Each model specific page:

* Model cards (like a readme file)
* Models can have checkpoints with varying number of parameters. These types of models come in different sizes. Checkpoint refers to the saved model, including the pre-trained weights and all necessary configurations. Load a model = Load a model checkpoint. Depending on your hardware, you may not be able to run the largest checkpoints.

Rule of thumb to estimate how much memory I will need for a model.

* Files and versions -> Find a file called pytorch\_model\_bin
  + This file stores the trained weights of the model, and you can easily see its size.
* Multiply the size of pytorch\_model\_bin file by 1.2: This is approximately how much memory you will need to run this model.

Task page: Find task first then models

To load the model you want from the Hugging Face Hub, you can use the Transformers Library. (Use in Transformers button) -> Find code snippets to know how to load model.

The pipeline object offers a high-level abstraction to solve tasks. It also takes care of complex pre-processing of inputs to match the model’s expectations. For example, some audio models expect the input audio to come in the shape of a “logmel” spectrogram. Text typically needs to be converted into so-called “tokens”, and images often need to be properly resized and normalized.

With the pipeline, you won’t need to do any of these pre-processing steps by hand.

**Lesson 2: Natural Language Processing**

In this lesson, you will build your own chatbot using an open-source model built by Meta.

See code.