Sum-IT

Video Transcription and Summarizing application using Whisper and ChatGPT

A brief introduction:

There are innumerable videos on YouTube that are too long for us to watch and process fully. Most of us do not have the time nor the attention span to absorb information from the videos we watch. To solve this problem, we have built a video summarizing application using Streamlit and Whisper-OpenAI and implemented it using ChatGPT's API. This tool provides a short summary of videos that have a very long duration to make it easy for us to learn.

What is Whisper?

Whisper is a feature of the OpenAI GPT-3 language model that allows for the generation of more subtle and nuanced language. Whisper was trained on 680,000 hours of multilingual and multitask supervised data collected from the web. One of the important characteristics of Whisper is the diversity of data used to train it. A third of the training data is composed of non-English audio examples. Whisper can robustly transcribe English speech and perform at a state-of-the-art level with approximately 10 languages – as well as translation from those languages into English. It enables transcription in multiple languages, as well as translation from those languages into English.

How does it work?

The Whisper architecture is a simple end-to-end approach, implemented as an encoder-decoder Transformer. Input audio is split into 30-second chunks, converted into a log-Mel spectrogram, and then passed into an encoder. A decoder is trained to predict the corresponding text caption, intermixed with special tokens that direct the single model to perform tasks such as language identification, phrase-level timestamps, multilingual speech transcription, and to-English speech translation.

What is ChatGPT?

Conversational AI is a central sub-field of Natural Language Processing that makes it possible for a human to have a conversation with a machine. Every time the human says or asks something to the AI, the whole conversation history is sent too, so the AI can have the context in memory and make relevant responses. Modern chatbots leverage conversational AI and can do more than simply having a conversation. For example, they can detect customer intents, search documents, understand the customer tone and adapt their own tone (anger, joy, sarcasm...).

What is gTTS?

gTTS (Google Text-to-Speech) is a Python library and CLI tool to interface with Google Translate text-to-speech API. There are several APIs available to convert text to speech in Python. One of such APIs is the Google Text to Speech API commonly known as the gTTS API. gTTS is a very easy to use tool which converts the text entered, into audio which can be saved as a mp3 file. The gTTS API supports several languages including English, Hindi, Tamil, French, German and many more. The speech can be delivered in any one of the two available audio speeds, fast or slow.

Sum-IT

Enter YouTube Link

Transcribe

Sum-IT

Enter YouTube Link

https://www.youtube.com/watch?v=fSytzGwwBVw

Transcribe

Video Transcription

Stack Quest, check it out, talking about machine learning, yeah. Stack Quest, check it out, talking about cross validation, yeah. Stack Quest. Hello, I'm Josh Starmer and welcome to Stack Quest. Today we're going to talk about cross validation and it's going to be clearly explained. Okay, let's start with some data. We want to use the variables, chest pain, good blood circulation, etc. To predict if someone has heart disease. Then when a new patient shows up, we can measure these variables and predict if they have heart disease or not. However, first we have to decide which machine learning method would be best. We could use logistic regression or K nearest neighbors or support vector machines and many more machine learning methods. How do we decide which one to use? Cross validation allows us to compare different machine learning methods and get a sense of how well they will work in practice. Imagine that this blue column represented all of the data that we have collected about people with and without heart disease. We need to do two things with this data. One, we need to estimate the parameters for the machine learning methods. In other words, to use logistic regression, we have to use some of the data to estimate the shape of this curve. In machine learning lingo, estimating parameters is called training the algorithm. The second thing we need to do with this data is evaluate how well the machine learning methods work. In other words, we need to find out if this curve will do a good job categorizing new data. In machine learning lingo, evaluating a method is called testing the algorithm. Thus, using machine learning lingo, we need the data to, one, train the machine learning methods, and two, test the machine learning methods. A terrible approach would be to use all the data to estimate the parameters, i.e. to train the algorithm. Because then we wouldn't have any data left to test the method. Reusing the same data for both training and testing is a bad idea because we need to know how the method will work on data it wasn't trained on. A slightly better idea would be to use the first 75% of the data for training, and the last 25% of the data for testing. We could then compare methods by seeing how well each one categorized the test data. But how do we know that using the first 75% of the data for training in the last 25% of the data for testing is the best way to divide up the data? What if we use the first 25% of the data for testing? Or what about one of these middle blocks? Rather than worry too much about which block would be best for testing, cross validation uses them all, one at a time, and summarizes the results at the end. For example, cross validation would start by using the first three blocks to train the method, and then use the last block to test the method. And then it keeps track of how well the method did with the test data. Then it uses this combination of blocks to train the method. And this block is used for testing. And then it keeps track of how well the method did with the test data, etc, etc, etc. In the end, every block of data is used for testing, and we can compare methods by seeing how well they performed. In this case, since the support vector machine did the best job classifying the test data sets, we'll use it. Bam! Note, in this example, we divided the data into four blocks. This is called fourfold cross validation. However, the number of blocks is arbitrary. In an extreme case, we could call each individual patient or sample a block. This is called leave one out cross validation. Each sample is tested individually. That said, in practice, it is very common to divide the data into ten blocks. This is called tenfold cross validation. Double Bam! One last note before we're done. Say like we wanted to use a method that involved a tuning parameter. A parameter that isn't estimated, but is just sort of guessed. For example, Ridge regression has a tuning parameter. Then we could use tenfold cross validation to help find the best value for that tuning parameter. Tiny Bam! Hooray! We've made it to the end of another exciting stat quest. If you like this stat quest and want to see more, please subscribe. And if you want to support stat quest, well, please click the like button down below and consider buying one of my original songs. Alright, until next time, quest on!

Transcription Summary

Stack Quest is a website that provides information on machine learning and cross validation. The author, Josh Starmer, explains how cross validation can be used to compare different machine learning methods and to find the best value for a tuning parameter.

Audio Summary