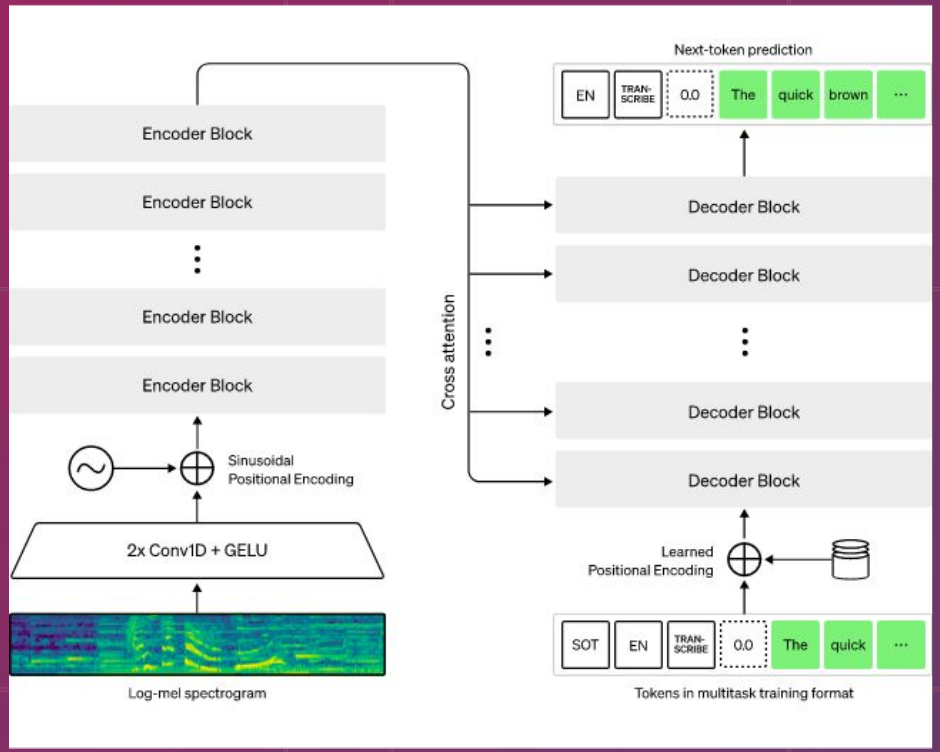
Lab 2
ID2223 / HT2022



Swedish Text Transcription using Transformers

Course Material: Prof Jim Dowling

Source Code for Lab 2

- Source Code Github

- Use Conda or virtual environments to manage your python dependencies on your laptop. See more info on how to manage your Python environment here.

Fine-Tune a Transformer For Swedish Language Transcription

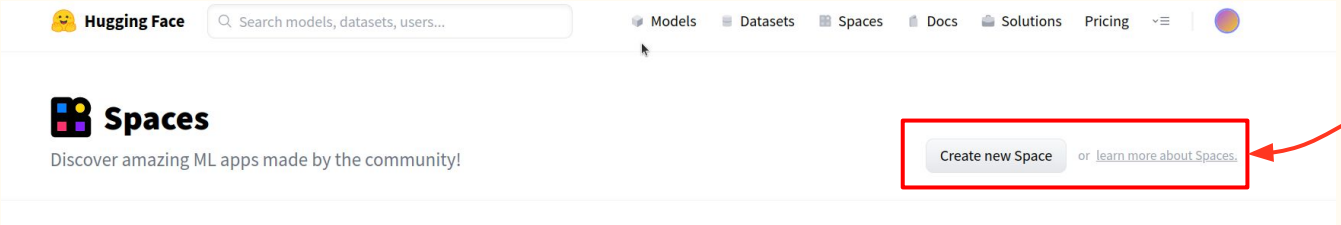
Task 1: Fine-tune a model for language transcription, add a UI

- Fine-Tune a pre-trained transformer model and build a serverless UI for using that model

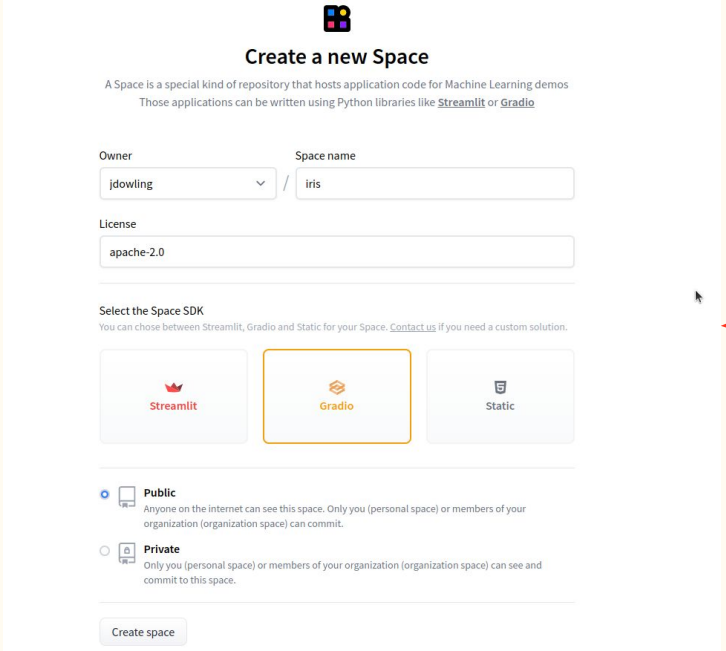
- **First Steps**
  a. Create a free account on huggingface.com
  b. Create a free account on google.com for Colab

- **Tasks**
  a. Fine-tune an existing pre-trained transformer model for the Swedish language, such as Whisper
  b. Build and run an inference pipeline with a Gradio UI on Hugging Face Spaces for your model.

Register and Create a Hugging Face Space

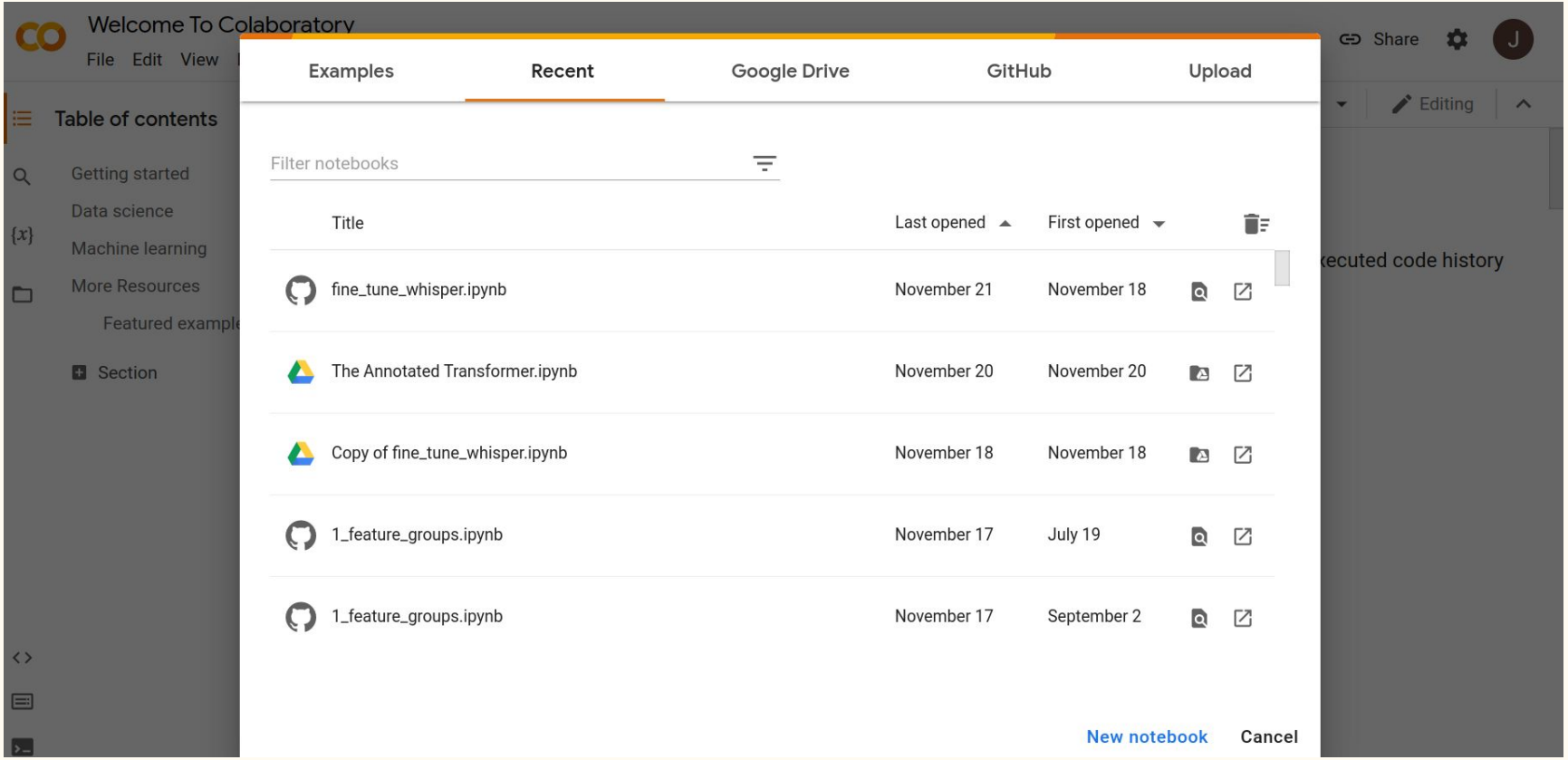


1. Create an account on Hugging Face
2. Create a “Space”



3. Create a Gradio App with the name Iris inside your account

Register and Create an account on Google for Colab



You can use the Whisper model by Hugging Face

- A sample Colab Notebook is available here.
- Here is a blog post explaining the example code

- You should fine-tune the model with either Swedish or your mother tongue. https://huggingface.co/datasets/mozilla-foundation/common_voice_11_0/viewer/sv-SE/train

- We recommend that you train your model with a GPU. Colab provides free GPUs for 12 hours (then it shuts down) - so make sure to save your model weights before it shuts down. Getting a good GPU (e.g., p100) is getting harder on Colab. Alternatively, you could modal.com to train your GPU, but i think they will quickly use up your $30 of free credit. Colab is free. If you have your own GPU, you can, of course, use that. You may need to reduce the number of training epochs or dataset size to train in time.

Communicate the value of your model with a UI (Gradio)

- Communicate the value of your model to stakeholders with an app/service that uses the ML model to make value-added decisions

Example UIs:

- Allow the user to speak into the microphone and transcribe what he/she says (lower grade, as this code is in the example code)
- Allow the user to paste in the URL to a video, and transcribe what is spoken in the video (higher grade)
- Your own creative idea for how to allow people to use your model (highest grade)

Task 2: Improve pipeline scalability and model performance

1. Describe in your README.md program ways in which you can improve model performance are using
   (a) **model-centric approach** - e.g., tune hyperparameters, change the fine-tuning model architecture, etc
   (b) **data-centric approach** - identify new data sources that enable you to train a better model that one provided in the blog post

   If you can show results of improvement, then you get the top grade.

2. Refactor the program into a feature engineering pipeline, training pipeline, and an inference program (Hugging Face Space) to enable you to run feature engineering on CPUs and the training pipeline on GPUs. You should save checkpoints when training, so that you can resume again from when Colab took away your GPU :)
   This is challenging - where you can store GBs of data from the feature engineering step? Google Drive? Hopsworks?

Deliverables

- Deliver your source code as a Github Repository.
- Deliver your description for task 2 as a README.md file in the root of your Github repository
- Deliver a Hugging Face Spaces public URL for the UI for your speech transcription model.

Deadline midnight 9th December.