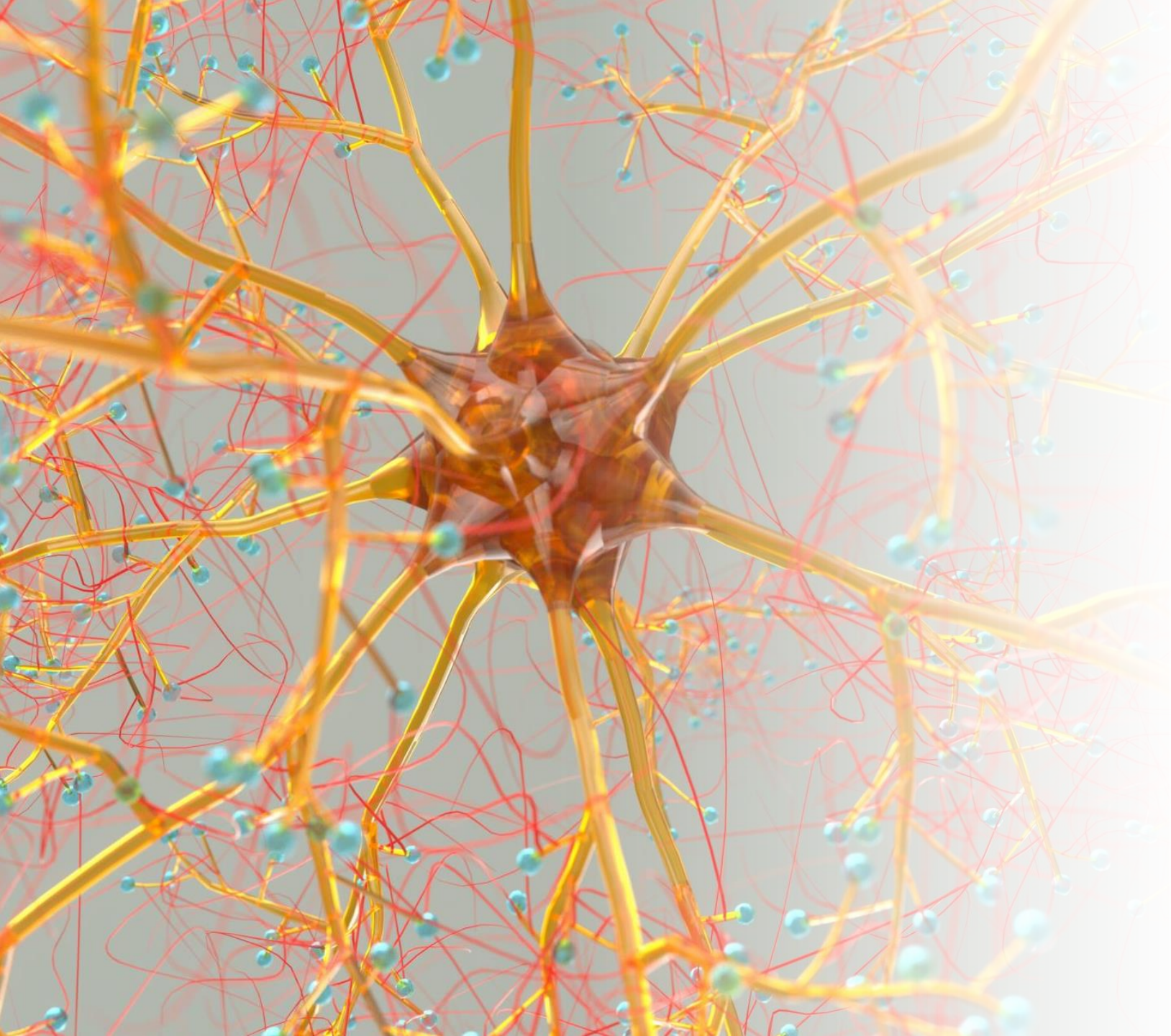
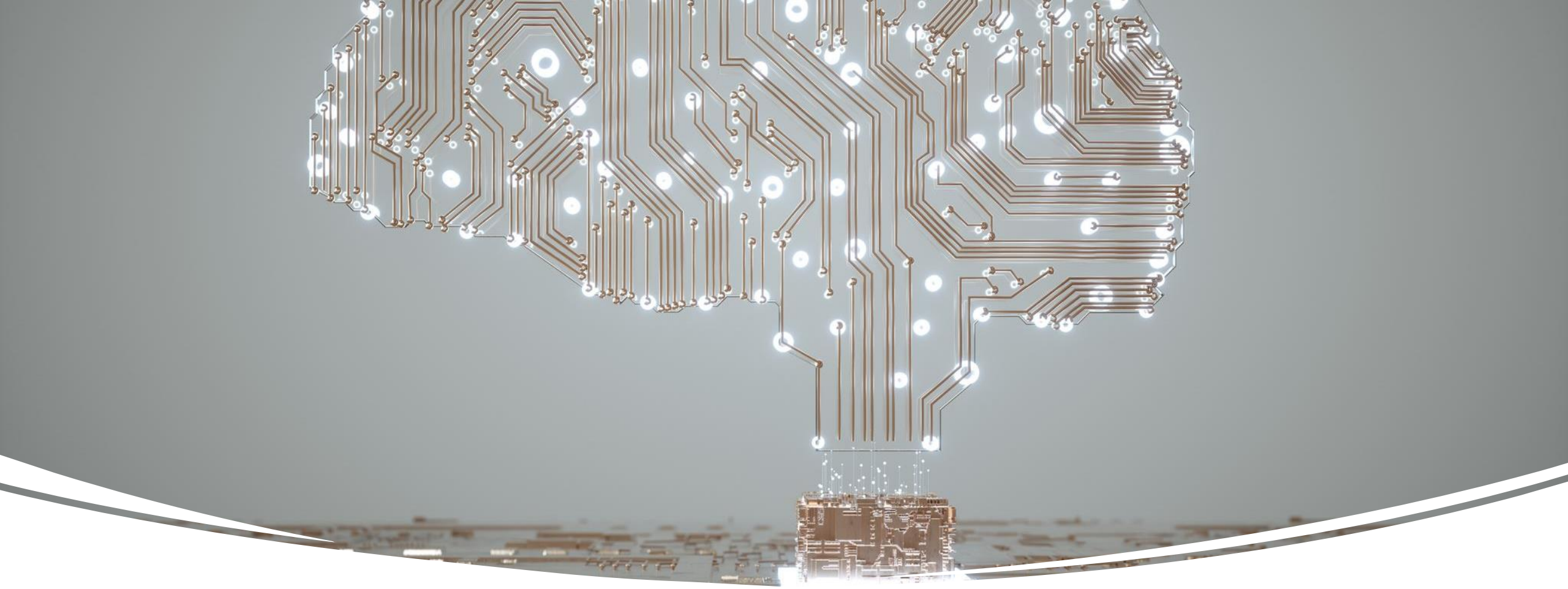


Neural Networks

Rodrigo Nogueira & Jawer Echeverry

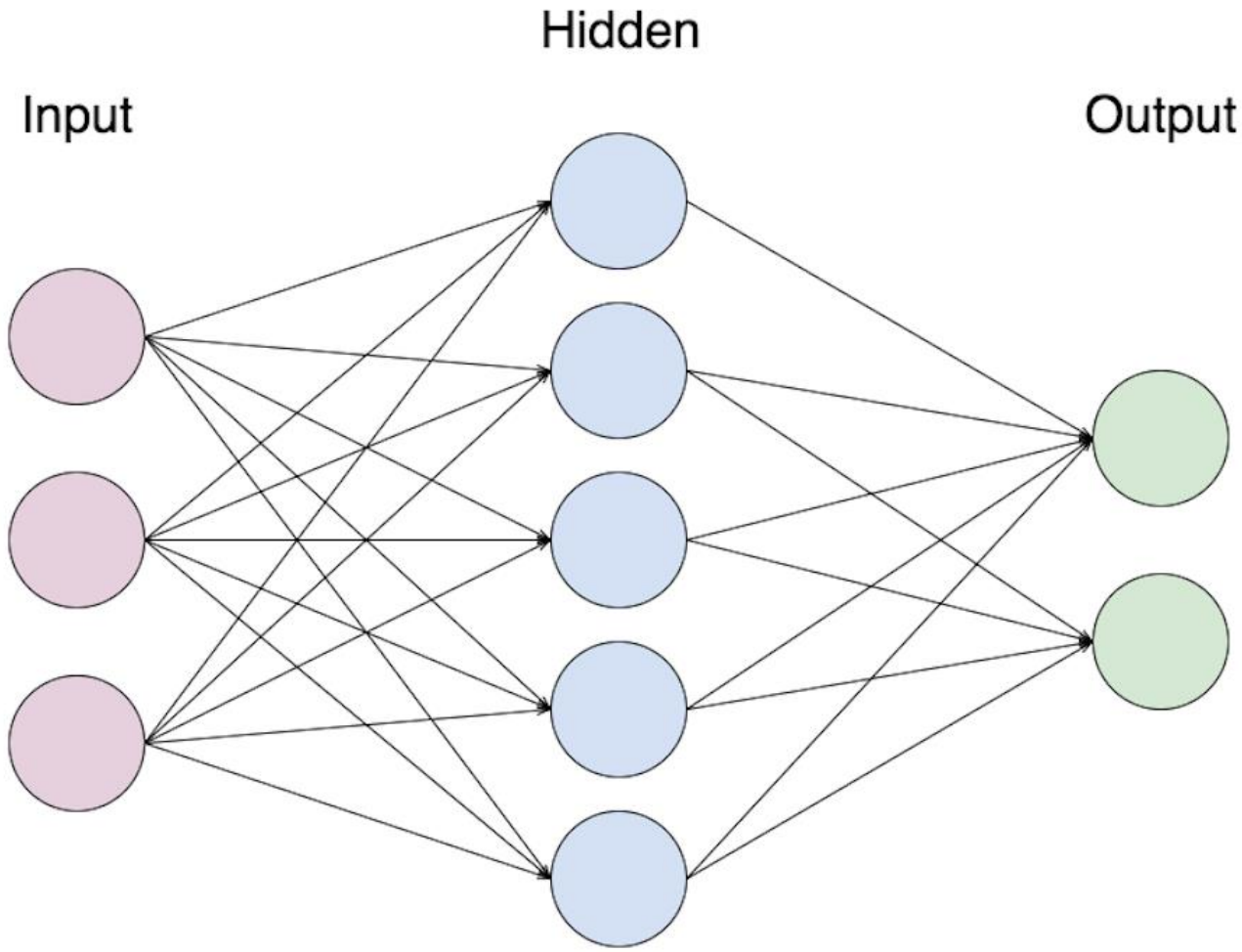


What is a Neural Network

- Neural networks are machine learning algorithms modeled after the human brain
- They consist of layers of nodes that process input data, with each layer extracting more complex features
- The connections between the nodes are modified during training to improve prediction accuracy

Neural network with three layers:

- Nodes (neurons) are represented by circles, and connections between them are represented by lines
- Input data is fed into the input layer, processed through the hidden layer, and produces an output in the output layer.



How They Are Used

Neural networks are used for:	They are also used for:	They improve decision making and automate tasks in:
• Image and speech recognition • Natural language processing • Autonomous vehicle navigation (GPS)	• Predictive analytics • Fraud detection • Recommendation systems	• Finance • Healthcare • Marketing

Different Types of Neural Networks

- Standard artificial neural network (ANN)
- Convolutional neural network (CNN)
- Recurrent neural network (RNN)
- Long short-term memory (LSTM)
- Stacked autoencoders
- Variational autoencoders

Stock Market

- Housing price prediction:
  - Standard artificial neural network (ANN) can be used for the real estate market.
  - Deep learning approach can also be used to predict the housing prices in a given area, city or country with high accuracy and low risk involved.
  - The input data can be different home features and the output prediction will be pricing estimate. This is a supervised learning problem.

Covid Image Classification

- Image classification:
  - Real world applications of image classification includes classification of images with humans, objects and scenes.
  - Business applications for image classification include surveillance, medical diagnosis (healthcare), tagging of images, X-ray interpretation, CT-Scans/MRI interpretation and so on.
  - Deep neural networks have the capability to recognize images at a pixel level which is almost impossible for humans.
  - During Covid-19 times, CNN models were used to classify X-ray/CT-scan images in predicting likelihood of a person suffering from Covid.
  - This is a supervised learning problem.

(a) Normal (b) Pneumonia (c) COVD-19

Tesla

- Autonomous driving:
  - A custom or hybrid neural network architecture comprising of CNN, ANN etc. will be required to build a bunch of models which can be used for autonomous driving.
  - The input to these models will be images, radar information (device put on the top of the car) and the label / output will be position of other vehicles, objects etc. which will help in deciding the way of safe driving.
  - A key aspect of learning will be supervised learning.

Siri

- Speech recognition:
  - Deep neural networks can be used to recognize speech.
  - Deep learning models for Speech Recognition are Deep Neural Networks trained using Deep Learning techniques/algorithms, specifically:
    - Deep Feed-Forward NN (FFNN)
    - Deep Recurrent NN (RNN)
    - LSTM.
  - A key aspect of learning will comprise of supervised learning.

Deep Dive into Real World Example of a Neural Network

Voice Synthesis

- Voice synthesis with AI often involves the use of neural networks, particularly deep learning models
- These neural networks are designed to learn and generate speech patterns by processing and understanding large amounts of audio data, from a dataset.
- Our example of a neural network used for voice synthesis will be a custom So-Vits-SVC model that we trained ourselves. It learns to generate human-like speech from raw reference vocals.

So-Vits-SVC 4.0

- We will be working with the So-Vits-SVC client
- Work will be based off of https://github.com/34j/so-vits-svc-fork
- Model creation and training will be run on Google Colab's cloud based GPU, using this client's script.
- Inferences will be made using the client's local GUI.
- Code shown in next few slides

Dataset Collection

Travis Scott Vocal Takes

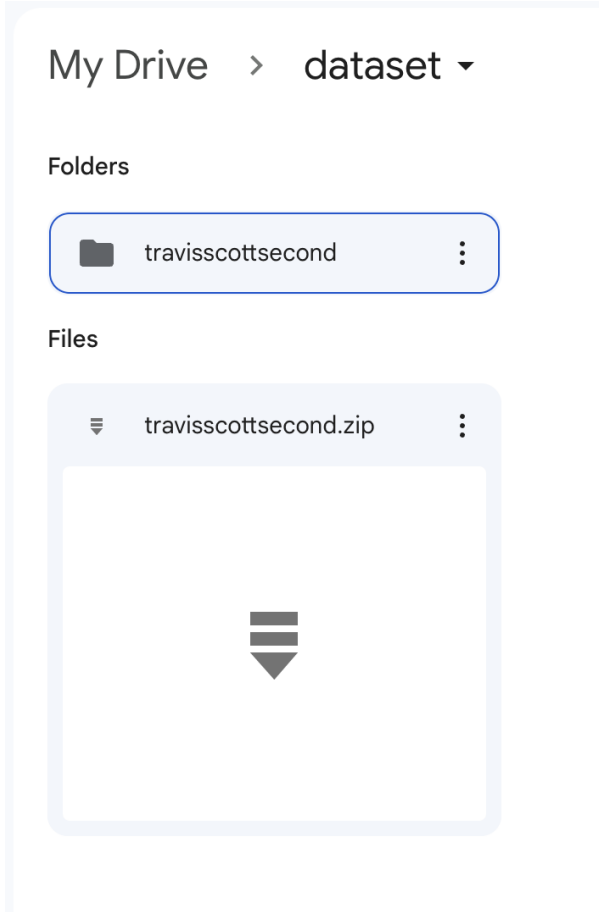
- Travis Scott is a well known rapper, singer, and songwriter, and this project example of voice synthesis through AI will focus on making an inference of his voice, using the specified client (So-Vits-SVC).
- In order to gather a proper dataset to create this model, we must collect raw 10-15 second vocal takes of his voice.
- The best way to do this is to browse public interviews on the internet, and cut up parts of it to collect these snippets of vocal takes.
- We need to make sure that the snippets have ONLY Travis Scott's vocals, with no background sound, interviewer, or sound effects included.

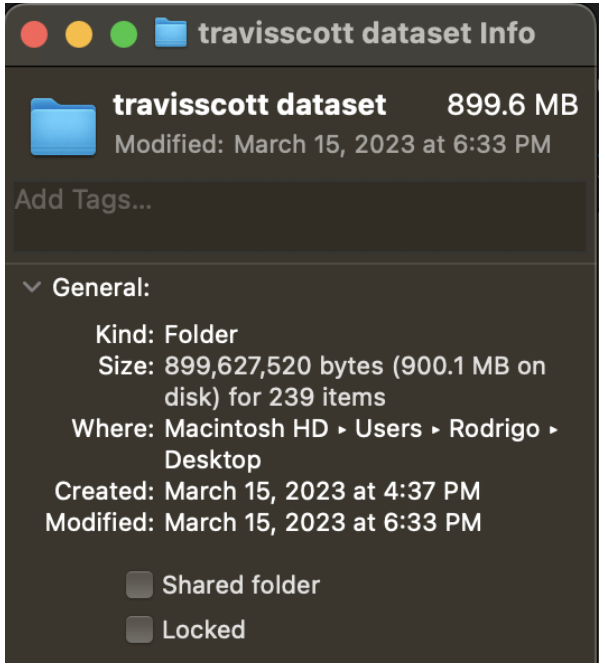
Travis Scott Studio Acapellas

- I have also included a couple publicly available studio acapellas.
- These are recordings of him singing/rapping, with no mixing effects added to alter the reference.
- Effects such as autotune, reverb, echo/delay and pitch shifting may negatively effect how the AI model will sound.

Dataset Size

- After collecting all the data needed to create the dataset, it came out to be about 240 .wav audio files of 10-15 second snippets
- Size of almost 1 gigabyte.
- Upload compressed zip folder of the dataset and upload to mounted Google Drive.
- At this point, the model is ready to be trained.



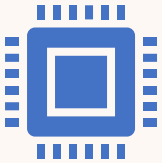


Training Process

Training Process

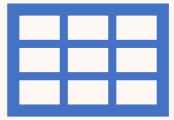
- What the So-Vits-SVC client does:
  - Downloads ContentVec model file
  - Connects to Google drive to store new copies of trained model
  - Loads and unzips compressed folder of uploaded dataset. This is what the model trains off of.
  - Resamples dataset

Training Process (cont.)



Generates new config file for current training session. This tells training progress, data and model values, and saves speaker name (name of person you are synthesizing).

Generates hubert and f0. Hubert is basically the pre-trained model that your new model uses as a reference.



Saves and loads the preprocessed dataset files for training convenience.

Saves new trained model files to mounted Google Drive for backup purposes. Automatically downloads pre-trained model to start training (G and D files).

Training Process (cont.)

- Starts training and turns on tensor board.
- Training basically means the AI model goes through the dataset a certain number of times, known as steps.
- Once the model reaches a milestone of 800 steps, it saves the new model (G and D files) into the mounted Google Drive.
- The tensor board is used to keep track of training progress in real-time, being able to hear inferences of how the model's voice is sounding during the training process.
- Users must manually delete old versions of model files from their Google Drives to maintain enough storage space.
- Once model reaches a certain amount of step intervals (typically 20,000 minimum) the AI voice model will start sounding more realistic, and ready to make inferences.

Commands and Python Code:

```
#@title Clone github repo
!git clone https://github.com/effusiveperiscope/so-vits-svc -b eff-4.0

#@title Install dependencies
%cd /content/so-vits-svc
!pip install pyworld praat-parselmouth
!python -m pip install --upgrade pip
!pip install fairseq==0.12.2 librosa==0.8.1

#@title Download ContentVec model file
!wget -P hubert/ https://huggingface.co/therealvul/so-vits-svc-4.0-
init/resolve/main/checkpoint_best_legacy_500.pt

#@title Mount google drive
#@markdown Mount google drive, which will be used to hold your dataset
files.
from google.colab import drive
drive.mount('/content/drive')

#@title Load the dataset from .zip in Google Drive for preprocessing
#@markdown Name of the zip folder
DATASETNAME = "Chrysalis"  #@param {type:"string"}
#@markdown Zip path (usually do not need to change this unless you
uploaded to a different directory)
ZIP_PATH = "/content/drive/MyDrive/dataset/"  #@param {type:"string"}
ZIP_NAME = ZIP_PATH + DATASETNAME

!unzip -d /content/so-vits-svc/dataset_raw {ZIP_NAME}.zip
#@title Resample to 44.1k
!python resample.py
```

```
#@title Segment training set and generate configuration files
!python preprocess_flist_config.py

#@title Generate hubert and f0
!python preprocess_hubert_f0.py

#@title Save preprocessed dataset files to Google Drive (for training
convenience)
#Compress dataset folder
!zip -r dataset.zip /content/so-vits-svc/dataset
#@markdown Customize name of preprocessed dataset folder to avoid
conflicts.
dataset_name_drive = "44k_dataset"  #@param {type:"string"}
DATASET_PATH_DRIVE = "/content/drive/MyDrive/dataset/" +
dataset_name_drive
!mkdir -p {DATASET_PATH_DRIVE}

!cp /content/so-vits-svc/dataset.zip "{DATASET_PATH_DRIVE}"
!cp configs/config.json "{DATASET_PATH_DRIVE}"
!cp filelists/train.txt "{DATASET_PATH_DRIVE}"
!cp filelists/val.txt "{DATASET_PATH_DRIVE}"

#@title Load preprocessed dataset files from Google Drive (no need to run
first round)
#@markdown If you already have preprocessed dataset files on Google Drive,
you can load them here instead of re-running the preprocessing steps.
back_up_name = "three_dataset"  #@param {type:"string"}
BACK_UP_DATASET_PATH = "/content/drive/MyDrive/dataset/" + back_up_name
!unzip {BACK_UP_DATASET_PATH}/dataset.zip -d /
!cp {BACK_UP_DATASET_PATH}/config.json /content/so-vits-
svc/configs/config.json
!cp {BACK_UP_DATASET_PATH}/val.txt filelists/val.txt
!cp {BACK_UP_DATASET_PATH}/train.txt filelists/train.txt

#@title  Model saving/pretrained model preferences
Clone = "44k"
%cd /content/so-vits-svc

#@markdown **Save the trained model files to Google Drive instead of the
colab runtime filesystem. You also need to check and execute this when
resuming training. If you are short on Drive space you may want to uncheck
this (in which case you must use the file menu to manually download the
model checkpoints in so-vits-svc/logs/ yourself and copy the latest
checkpoint back into so-vits-svc/logs/ to resume training).**
Save_to_drive = False #@param {type:"boolean"}
```

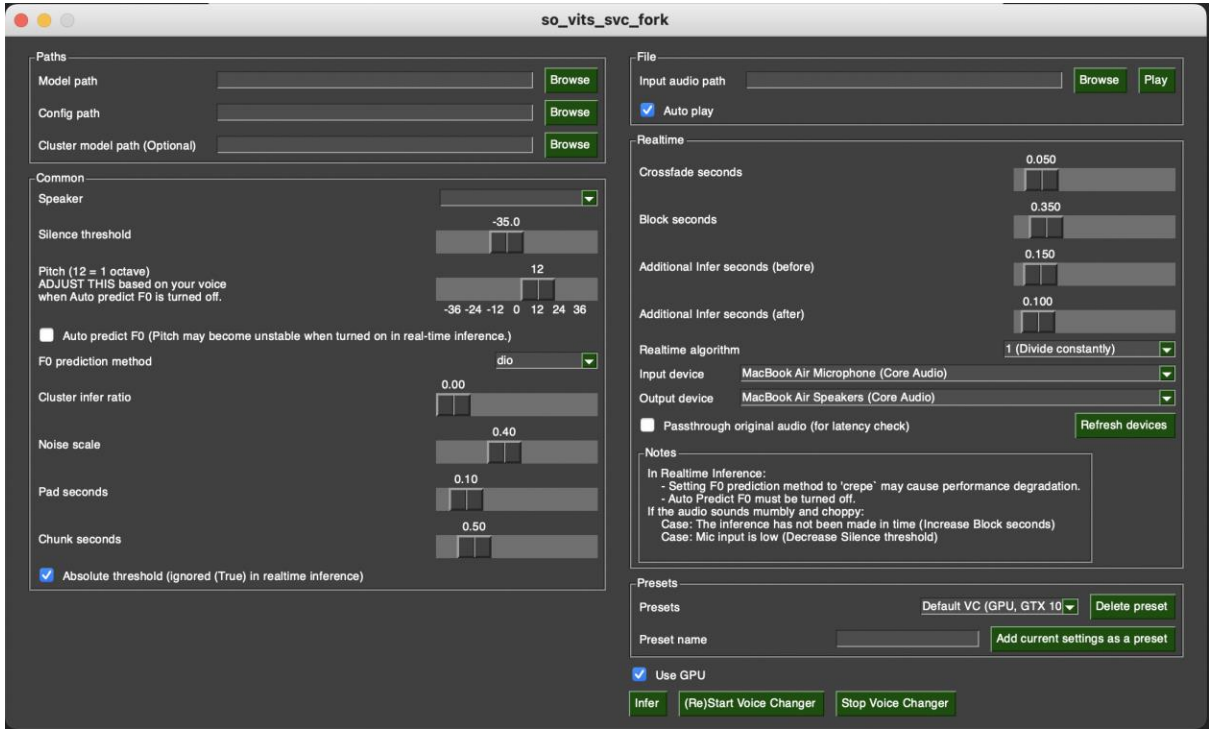
Commands and Python Code (cont.):

```
if Save_to_drive:
  !rm -rf /content/so-vits-svc/logs/"{Clone}"
  !mkdir -p /content/drive/MyDrive/"{Clone}"
  !ln -s /content/drive/MyDrive/"{Clone}" /content/so-vits-
svc/logs/"{Clone}"

#@markdown **Download the pre-trained model for the first training. For
continuing training, use the checkpoints saved by yourself after
continuing to train.**
pre_pth = True #@param {type:"boolean"}
if pre_pth:
  !wget -O logs/"{Clone}"/G_0.pth -P logs/"{Clone}"/
https://huggingface.co/therealvul/so-vits-svc-4.0-
init/resolve/main/G_0.pth
  !wget -O logs/"{Clone}"/D_0.pth -P logs/"{Clone}"/
https://huggingface.co/therealvul/so-vits-svc-4.0-
init/resolve/main/D_0.pth

#@title  Start training
#@markdown **Start training**
Clone = "44k"

#@markdown **Enable tensorboard for data visualization**
tensorboard_on = True #@param {type:"boolean"}
if tensorboard_on:
  %load_ext tensorboard
  %tensorboard --logdir logs/"{Clone}"

!python train.py -c configs/config.json -m "{Clone}"
```

Inference Process

Inference Process

- What the So-Vits-SVC GUI does:
  - Downloads the Hubert model file. Used for reference for own model.
  - GUI locates path for model directory (G and config files).
  - Upload raw audio file (reference audio, or audio file that you want the AI model to imitate).



Inference Process (cont.)

- Client then locates:
  - G file path (G file is required for making inferences
  - D file is only needed for training), config file (so that the script knows what speaker to use)
  - Raw audio
  - Transpose (pitch of inference voice, typically stays at 0).
- An inference of the AI model is created based on the raw audio, and it outputs into the same directory as the raw audio is located.

Commands and Python Code:

```
#@title Clone Github Repository
!git clone https://github.com/effusiveperiscope/so-vits-svc -b eff-4.0

#@title Install Dependencies
%cd /content/so-vits-svc
!pip install pip==23.0.1
!pip install pyworld==0.3.1 praat-parselmouth
!pip install fairseq==0.12.2 librosa==0.8.1

#@title Download the necessary model files
# Source Repository Address :
[contentvec](https://github.com/auspicious3000/contentvec)
# Model original download link :
[checkpoint_best_legacy_500.pt](https://ibm.box.com/s/z1wgl1stco8ffooyatzd
wsqn2psd9lrr)
# Since the source network disk can not provide http direct link,
according to the mit protocol, the model is distributed twice to provide
download direct link
!wget -P hubert/ https://huggingface.co/datasets/makiligon/required-stuff-
for-things/resolve/main/checkpoint_best_legacy_500.pt

#@markdown ## Mount your Google Drive
#@markdown (This is an essential step if you want to load your own trained
model)
from google.colab import drive
drive.mount('/content/drive')

#@title Import model and config file from Google Drive
#@markdown example of whats inside the zip file
https://cdn.discordapp.com/attachments/749847915660443678/1080738982654132
234/image.png remember to add a slash at the end of ZIP_PATH
ZIP_FILE_NAME = "arianagrande"  #@param {type:"string"}
ZIP_PATH = "/content/drive/MyDrive/models/"  #@param {type:"string"}
CONFIG_NAME = "config"  #@param {type:"string"}
ZIP_NAME = ZIP_PATH + ZIP_FILE_NAME

!unzip -d /content/so-vits-svc/logs/44k {ZIP_NAME}.zip
!mv "/content/so-vits-svc/logs/44k/{CONFIG_NAME}.json" "/content/so-vits-
svc/configs"
```

```
#@markdown ## Upload your reference audio or zip file of reference audios
#@markdown alternatively you can just drag and drop from google drive to
so-vits-svc/raw if you bothered mounting to google drive. Google Drive
method is faster

%cd "/content/so-vits-svc/raw/"

from google.colab import files
uploaded = files.upload()

%cd "/content/so-vits-svc/"
print("\n\033[32m\033[1mdone")

#@title Synthesize audio (inference)
#@markdown SPK_NAME is the name of the folder you gave to the training
dataset. If you dont remember the SPK_NAME check config.json
G_MODEL_PATH = "logs/44k/G_69420.pth"  #@param {type:"string"}
CONFIG_PATH = "configs/config.json"  #@param {type:"string"}
SPK_NAME = "beberexha"  #@param {type:"string"}
RAW_AUDIO = "Name_of_audio_file.wav"  #@param {type:"string"}
TRANSPOSE = "0"  #@param {type:"string"}

!python inference_main.py -m "{G_MODEL_PATH}" -c "{CONFIG_PATH}" -n
"{RAW_AUDIO}" -t {TRANSPOSE} -s {SPK_NAME}
```

Finished Song Remaster

Low Quality Snippet from Movie

Remastered Using My Demonstrated AI Model

References

- Neural Networks
  - Baker, B. (2022, February 10). Computer Scientists Prove Why Bigger Neural Networks Do Better. Quanta Magazine. https://www.quantamagazine.org/computer-scientists-prove-why-bigger-neural-networks-do-better-20220210/.
  - Burns, Ed, and John Burke. “What Is a Neural Network? Explanation and Examples.” Enterprise AI, TechTarget, 26 Mar. 2021, https://www.techtarget.com/searchenterpriseai/definition/neural-network.
  - Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT press.
  - Larry Hardesty | MIT News Office. “Explained: Neural Networks.” MIT News | Massachusetts Institute of Technology, https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414.
  - LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444.
  - Russell, S. J., & Norvig, P. (2010). Artificial Intelligence: A Modern Approach. Prentice Hall.
  - Vitalflux. (2021, July 22). Deep Neural Network Examples from Real Life. https://vitalflux.com/deep-neural-network-examples-from-real-life/.

References (cont.)

- Voice Synthesis
  - Bansal, S. (2021). Deep Learning for Speech Synthesis: A Review. IEEE Access, 9, 56224-56243. https://doi.org/10.1109/ACCESS.2021.3072378.
  - Google. (2021, September 29). G Suite Learning Center: Google Docs. https://docs.google.com/document/d/1y1pfS0LCrwbbvxdn3ZksH25BKaf0LaO13uYppxIQnac/edit#heading=h.dc64xv18v6pq.
  - svc-develop-team. (2022, January 10). so-vits-svc. GitHub. https://github.com/svc-develop-team/so-vits-svc.
  - Van Den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., & Kavukcuoglu, K. (2016). WaveNet: A generative model for raw audio. arXiv preprint arXiv:1609.03499. Retrieved from https://arxiv.org/abs/1609.03499.
  - 34j. (2022, February 15). so-vits-svc-fork [Forked from svc-develop-team/so-vits-svc]. GitHub. https://github.com/34j/so-vits-svc-fork.