ZillionPitches

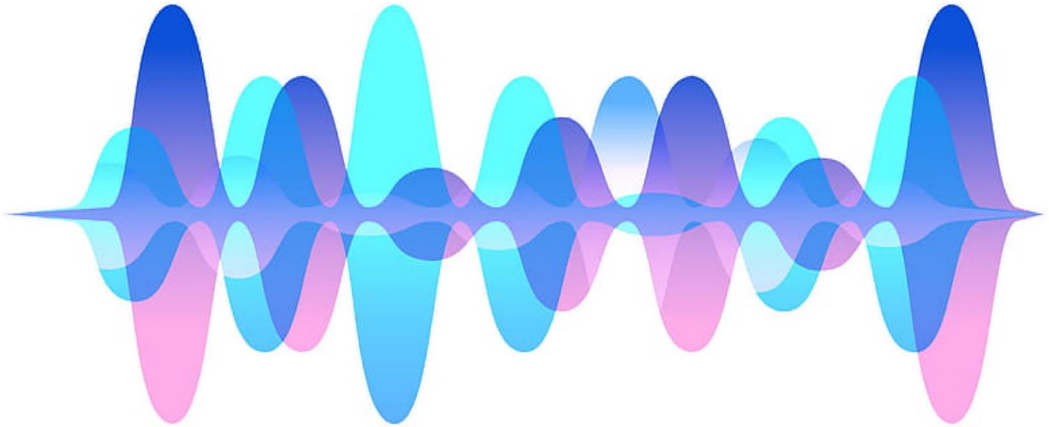
VOICE ANALYSIS

ZakaAi-MLC Program

6th of June 2022

Soulayman Al-Abdallah

Raed Diab



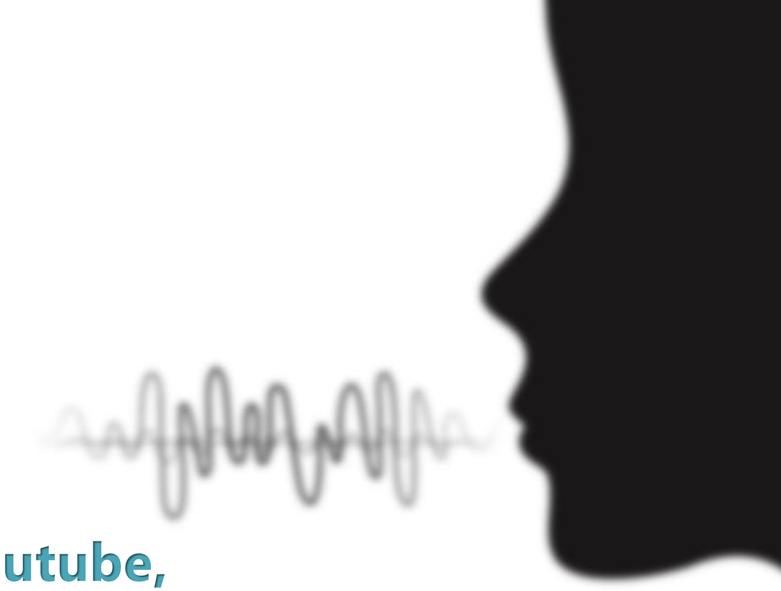
Project for: ZillionPitches Ai Solutions

A significant percentage of business start-ups fail due to some weaknesses in their elevator pitch.

Some business founders come up with unique, scalable and sustainable ideas, that satisfy the needs of a target niche in the market, but some pitching deficiencies disturb the accurate understanding by the potential investors.

We are using AI to help founders detect and correct mistakes in their elevator's pitching and content, by scoring some features through voice analysis.
In this way, they will be ready enough for pitching their ideas and succeed in getting the needed financial support.

Technical Approach



Our machine learning model focuses on two relevant features extracted from the pitch: Language Fluency & Sound Emotions.
The business founder needs to upload his elevator pitch on Youtube, and provide our web Flask app with the video's URL.

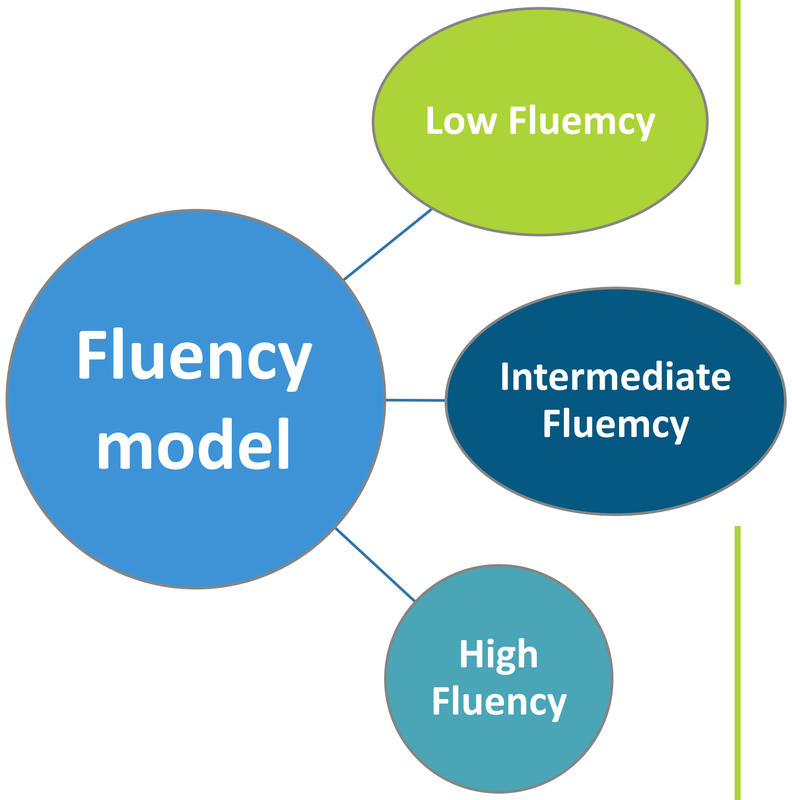
The video will be downloaded, the audio .wav copy will be generated, divided into windows of 5 seconds, each window will get the suitable classification by the model and then an overall classification of the whole audio file will be shown.

Now for each 5s audio window, we pass it in both fluency model and emotions model we used a neural network with conv1D, pooling, and dense layers in addition to some dropouts and batch normalizations.

Fluency model

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d (Conv1D)              (None, 23, 64)            256

conv1d_1 (Conv1D)            (None, 21, 64)            12352

max_pooling1d (MaxPooling1D  (None, 7, 64)             0
)

conv1d_2 (Conv1D)            (None, 7, 32)             6176

conv1d_3 (Conv1D)            (None, 7, 32)             3104

global_average_pooling1d (G  (None, 32)                0
lobalAveragePooling1D)

dropout (Dropout)            (None, 32)                0

dense (Dense)                (None, 3)                 99

=================================================================
Total params: 21,987
Trainable params: 21,987
Non-trainable params: 0
_________________________________________________________________
```

1D array of signal processing features by Librosa



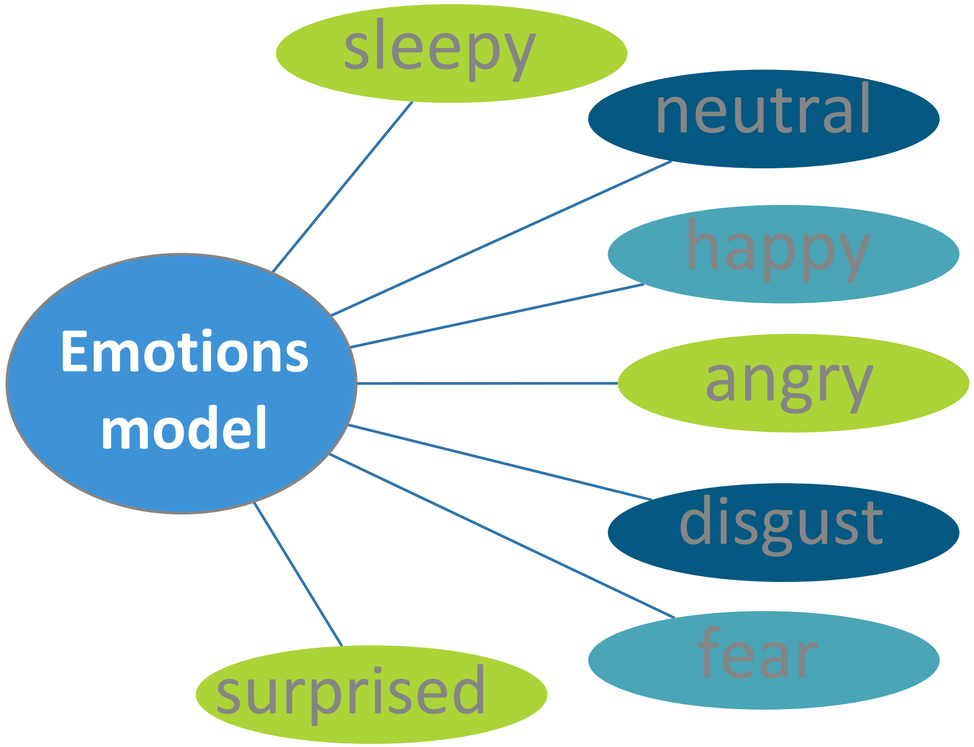
At the end, regarding the whole audio, a percentage of each class, for both, the fluency and emotion classification model, will be featured on the interface.

Now for each 5s audio window, we pass it in both fluency model and emotions model we used a neural network with conv1D, pooling, and dense layers in addition to some dropouts and batch normalizations.

Emotions Model

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 conv1d (Conv1D)              (None, 23, 128)           1152

 activation (Activation)      (None, 23, 128)           0

 conv1d_1 (Conv1D)            (None, 23, 128)           131200

 batch_normalization (BatchN  (None, 23, 128)          512
 ormalization)

 activation_1 (Activation)    (None, 23, 128)           0

 dropout (Dropout)            (None, 23, 128)           0

 max_pooling1d (MaxPooling1D  (None, 2, 128)            0
 )

 conv1d_2 (Conv1D)            (None, 2, 64)             65600

 batch_normalization_1 (Batc  (None, 2, 64)             256
 hNormalization)

 activation_2 (Activation)    (None, 2, 64)             0

 dropout_1 (Dropout)          (None, 2, 64)             0

 conv1d_3 (Conv1D)            (None, 2, 32)             16416

 activation_3 (Activation)    (None, 2, 32)             0

 flatten (Flatten)            (None, 64)                0

 dense (Dense)                (None, 7)                 455

 activation_4 (Activation)    (None, 7)                 0

=================================================================
Total params: 215,591
Trainable params: 215,207
Non-trainable params: 384
_________________________________________________________________
```

1D array of signal processing features by Librosa



At the end, regarding the whole audio, a percentage of each class, for both, the fluency and emotion classification model, will be featured on the interface.

* IBM Watson uses **speech-to-text** conversion, then classifies the emotion feature from the generated text directly.
* Our approach differs by the fact that the designed model, identifies the speech emotions and language fluency through **direct voice analysis**.
* After a wide research about machine learning use to classify audios, we found unsatisfying accuracies, and biased model cases.
* The thing we took into consideration, while gathering and training on more data, designing an immune architecture, and tuning parameters to seek a **significant accuracy** score.

Have others solved it?

Fluency model preparation:

- **Cloning** dataset from: Avalinguo-Audio-Set
- Using Librosa library to extract **23 audio features**: MFCCs, RMSE, Spectral flux, Zero Crossing Rate
- **Rewriting** the audio data samples in a suitable .wav format
- **Labeling** the data samples based on their father directory
- Classes are:Low Fluency: 438 Intermediate: 527 High Fluency: 459

Method description with technical details of your model and data preprocessing pipeline.

Emotions model preparation:

- **Gathering** 4 datasets from kaggle.com and GitHub.com : SAVEE - RAVDES - TESS - CREMA-D
- Renaming and **unifying** the classes for all datasets samples
- Using **Librosa** library to extract 23 audio features: MFCCs, RMSE, Spectral flux, Zero Crossing Rate, Spectral Bandwidth, Spectral Centroid..
- Classes are: Angry: 1923 Fear: 1923 Disgust: 1923 Happy: 1923 Sad: 1923 Neutral: 1935 Surprize:652

Method description with technical details of your model and data preprocessing pipeline.

Fluency model preparation:

- One hot **encoding** the labels of the samples
- Using train_test_split to **divide** data between testing(30%) and training(70%)
- Creating a **traditional neural network** sequential model with 7 dense layers
- After **fitting** the model on our training data, with a batch size of 16, sgd optimizer, and 100 epochs, we reached a training accuracy of 60% and validation accuracy of 52%.

Different Conducted Experiments

Fluency model preparation:

- Since the accuracy score was not satisfying, we redesigned the model by creating a **sequential model** with 7 layers (conv1D, pooling, dropouts, dense)

- After fitting the model on our training data, with a batch size of 32, rmsprop optimizer, and 100 epochs, we reached a training **accuracy** of 98% and validation **accuracy** of 91%.

Different Conducted Experiments

Emotions model preparation:

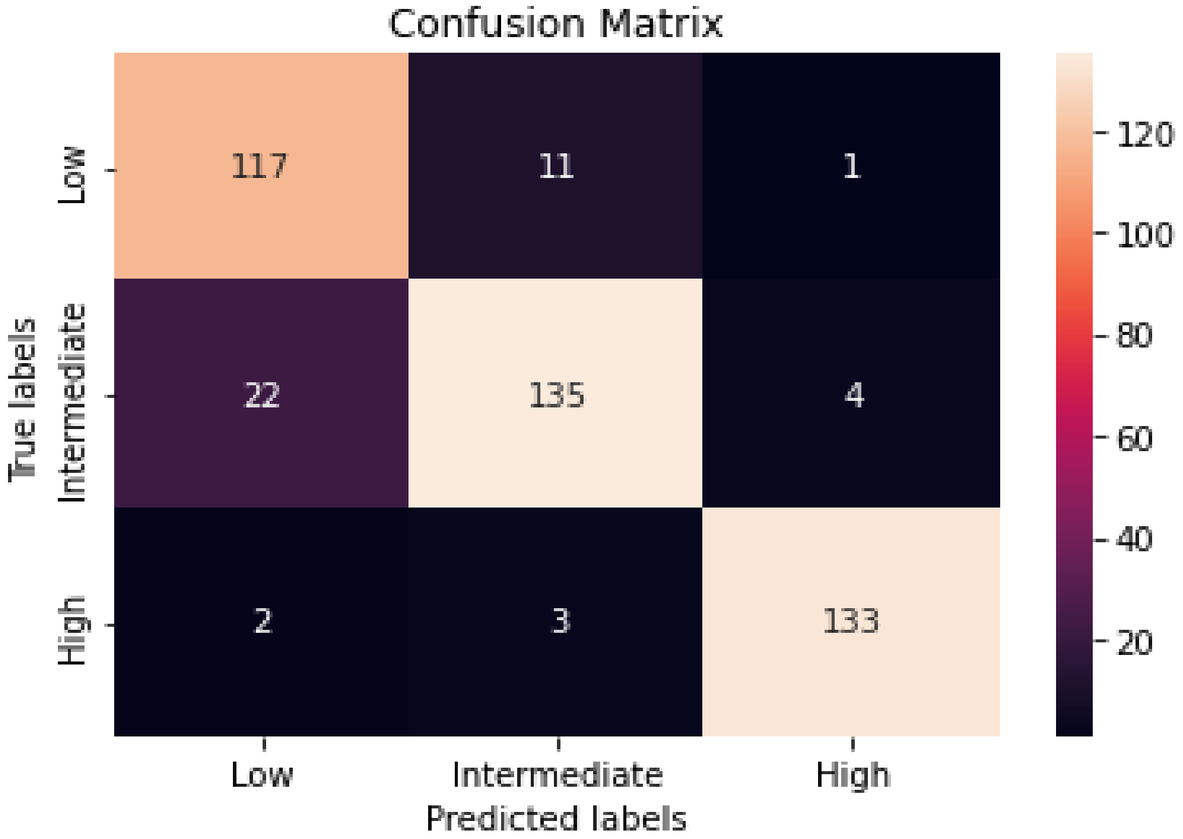
- One hot **encoding** the labels of the samples
- Using train_test_split to **divide** data between testing(25%) and training(75%)
- Trying the same model used for the language fluency feature here, and editing its output layer for sure.
- After **fitting** the model on our training data, with a batch size of 16, Adam optimizer, and 100 epochs, we reached a training accuracy of 55% and validation accuracy of 45%.

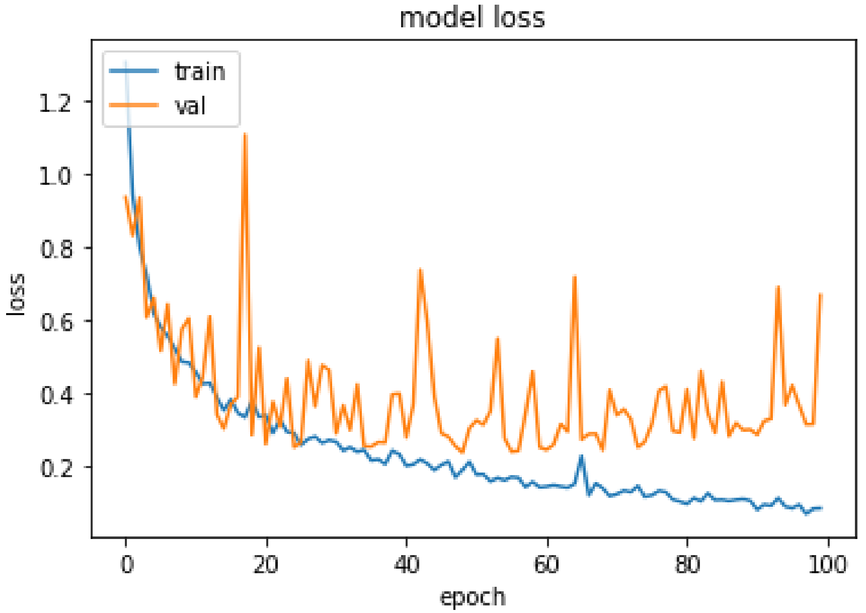
Different Conducted Experiments

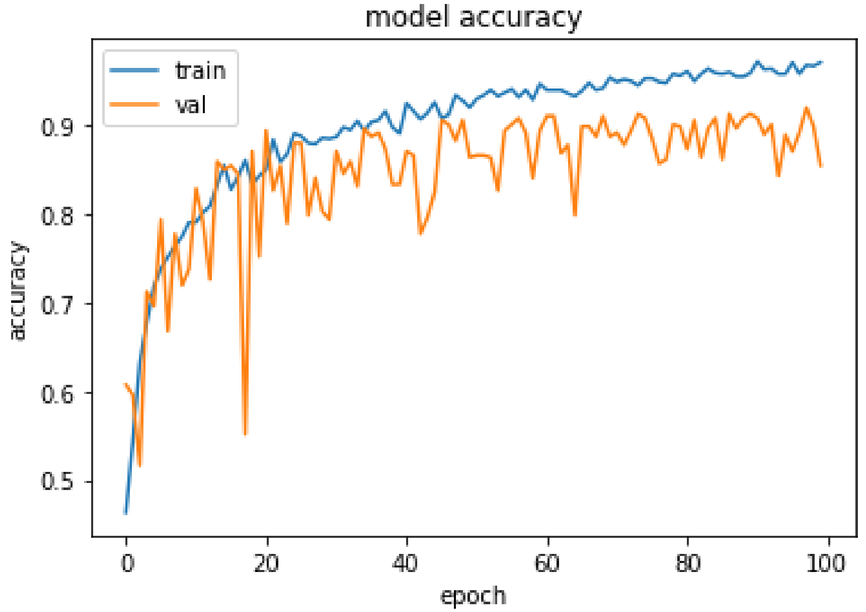
Emotions model preparation:

- Since the use of the model, over different data distributions did not lead to accepted results, we tried to improve our application on **3 levels**: Data, model architecture and hyper parameters tuning.
- In addition to the 4 datasets, we increased the training set with the **OpenSLR** dataset, and to treat the occurring unbalance of classes (relatively lower amount of audios labeled “Surprised”, we used Librosa library to generate new “surprised” samples having some noise, pitch and shift **effects**.
- On the model design level, we used a more **complicated** sequential model with about 14 layers (conv1D, pooling, dropouts, batch normalization, dense)
- After **fitting** the model on our updated training data, with a batch size of 16, Adam optimizer, and 100 epochs, we reached a training accuracy of 74% and validation accuracy of 73%.

Different Conducted Experiments

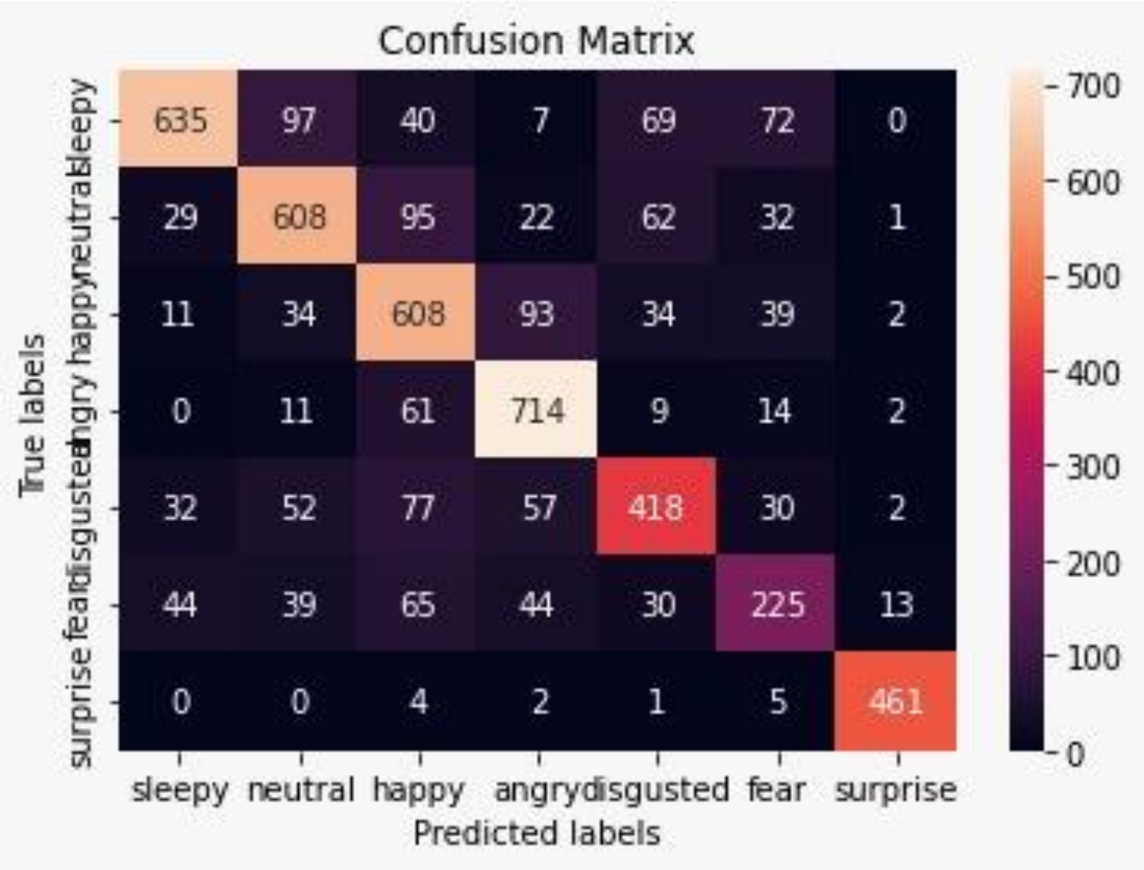


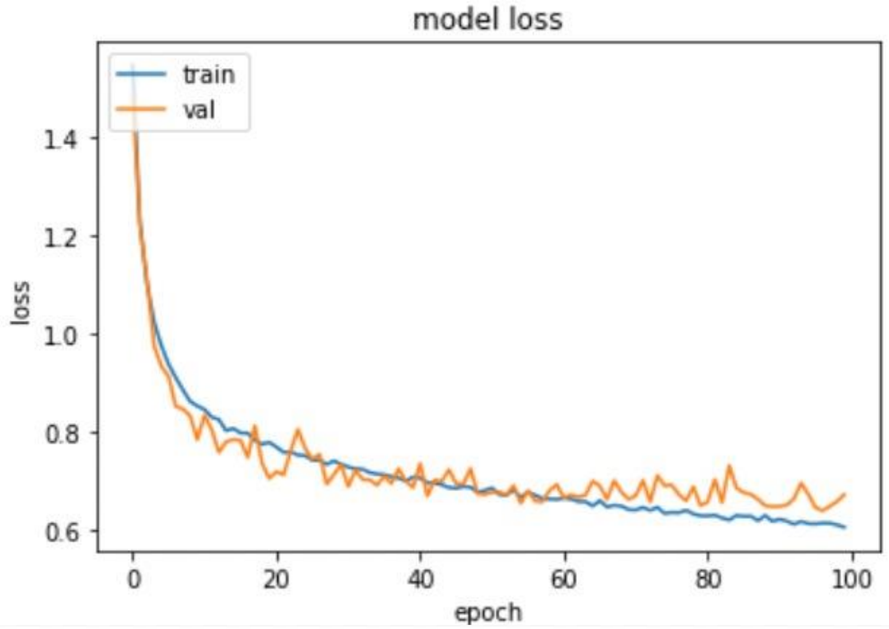


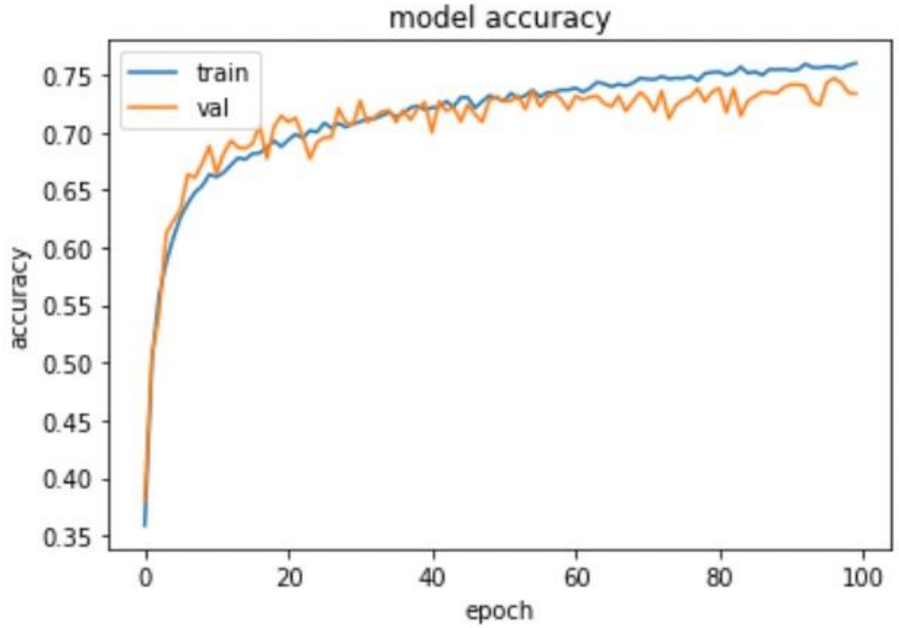


Fluency Model Metrics

Emotions Model Metrics







ZAKA AI

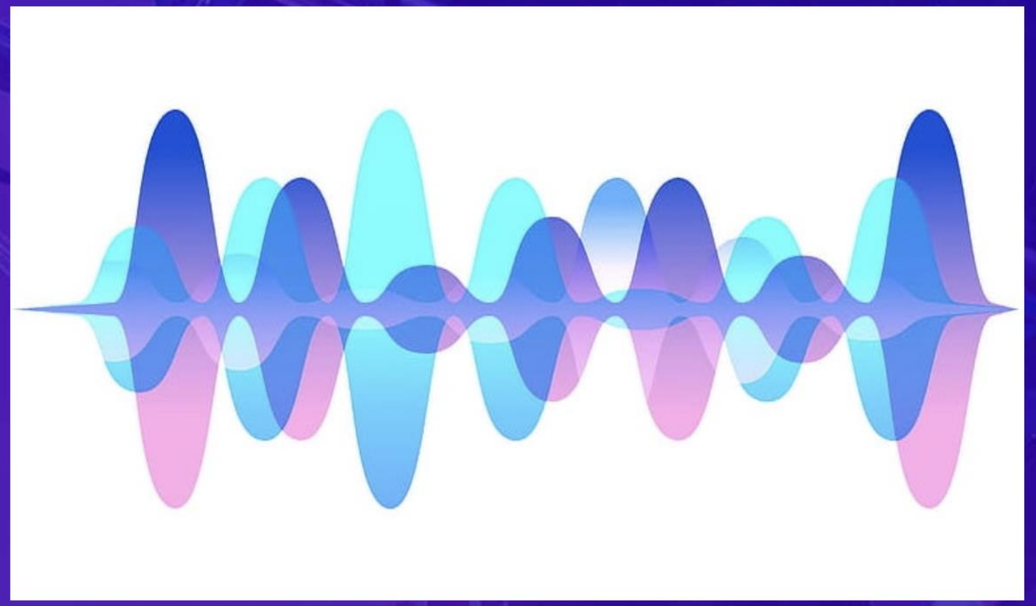
Github Repo of this App

Improve your elevator pitch by describing the speaking fluency and emotions

Our model recieves a URL of your uploaded pitch on Youtube,
and provides you with unique insights about your voice's emotions and language fluency

Put your Youtube URL here

Extract Features and wait for results downthere



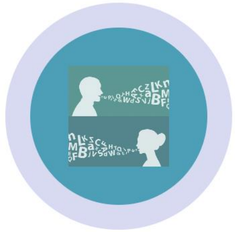
ZAKA AI

Github Repo of this App

Improve your elevator pitch by describing the speaking fluency and emotions

Our model recieves a URL of your uploaded pitch on Youtube,
and provides you with unique insights about your voice's emotions and language fluency

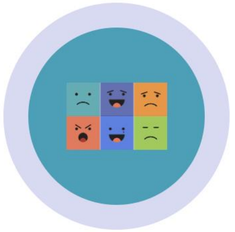
RESULTS



FLUENCY

Results will be shown here

Your Flwency in the uploaded video was as follows:
0% of your pitching was Very Low
29% of your pitching was Intermediate
71% of your pitching was High



EMOTIONS

Results will be shown here

Your Emotions in the uploaded video was as follows:
94% of your pitching was sleepy
6% of your pitching was neutral
0% of your pitching was happy
0% of your pitching was Very angry
0% of your pitching was disgusted
0% of your pitching was fear
0% of your pitching was surprise

The created models showed satisfying scores, and could be reliable as a speech scoring tool.

Based on the using purpose, we can map the output scores and percentages, to case-based classifications, ex: considering "happy" and "surprised" as "Excited about his start-Up", or "neutral" and "sleepy" as "unconfident about his solution's potential", and so on..

Also, the fluency score, can describe how smoothly the founder's ideas are delivered to the ears of the potential investors.

Futures steps that can upscale our application, are adding more features either computed or model predicted, like the word-level pronunciation, filler words detecting, same word repetition, tonality levels..etc

Conclusion