Spoken Digit Recognition with Deep Neural Network

Rolamjaya Hotmartua

MSCA 31009 - Machine Learning and Predictive Analytics

Spring 2023

Github link for this project: https://github.com/Rolamjaya/ML_Final_Project_MSCA.git

Outline

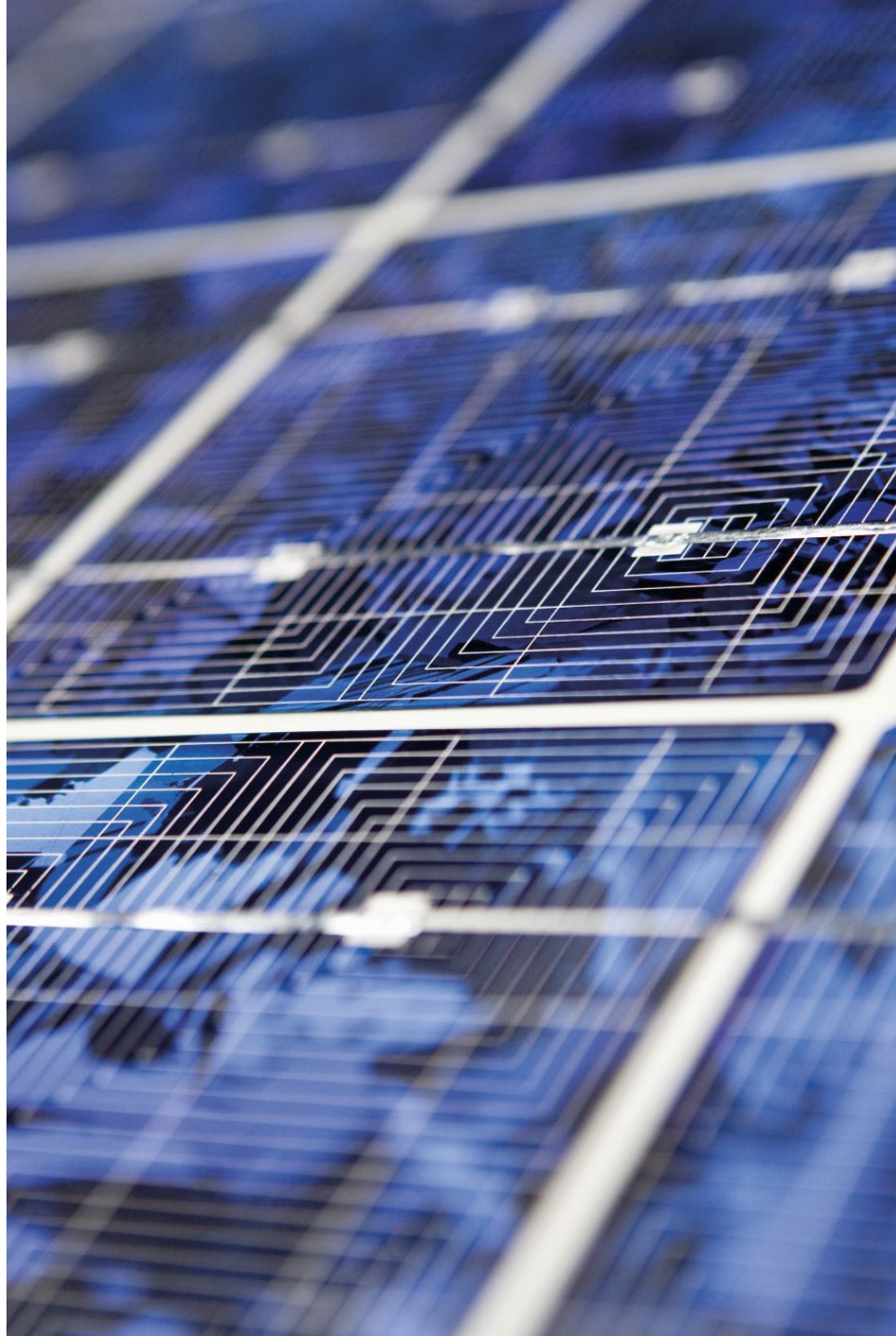
Introduction

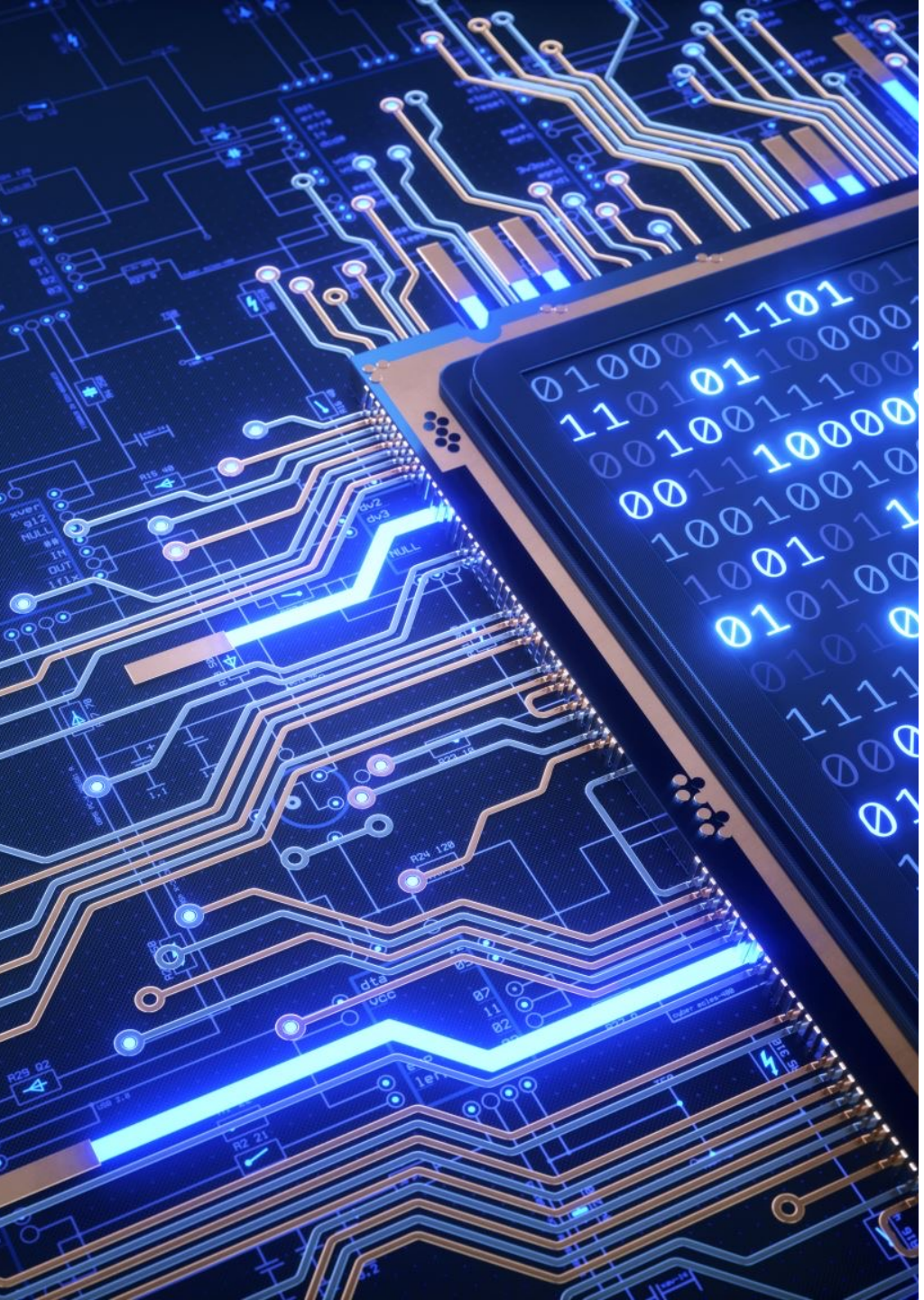
Data Sources, Preparation and Exploration

Model Exploration

Model Evaluation

Future Work





Introduction

- Human speech recognition is an important technology to enable machine understand human command through human voice.
- Spoken digit recognition is one application of human speech recognition used in various business context, e.g. to recognize phone number, account number, etc. through phone.
- Deep Neural Network enable machine to translate this spoken digit to text and then process the information for further purpose.
- This project develop a Deep Neural Network architecture to accurately classify human speech containing numerical information into text-based numbers.

Target Project and Assumption

- Build a model that could be able to recognize cardinal numbers (e.g., "one," "two," "three") from human voice recording, regardless of the medium of recording, gender and pronunciation.
- Model are trained with two methods based on the target:
    1. Scenario 1: Training and testing data come from all speakers.
    2. Scenario 2: Training and testing data come from different speaker.
- Model built is not processed real time.
- To see the performance of the model for recording from different medium of recording and gender, we test model using new independently generated dataset.
- Hypothesis: Model will work well regardless of the medium of recording, gender and pronunciation.
- Measure of success: Generalization and Accuracy.

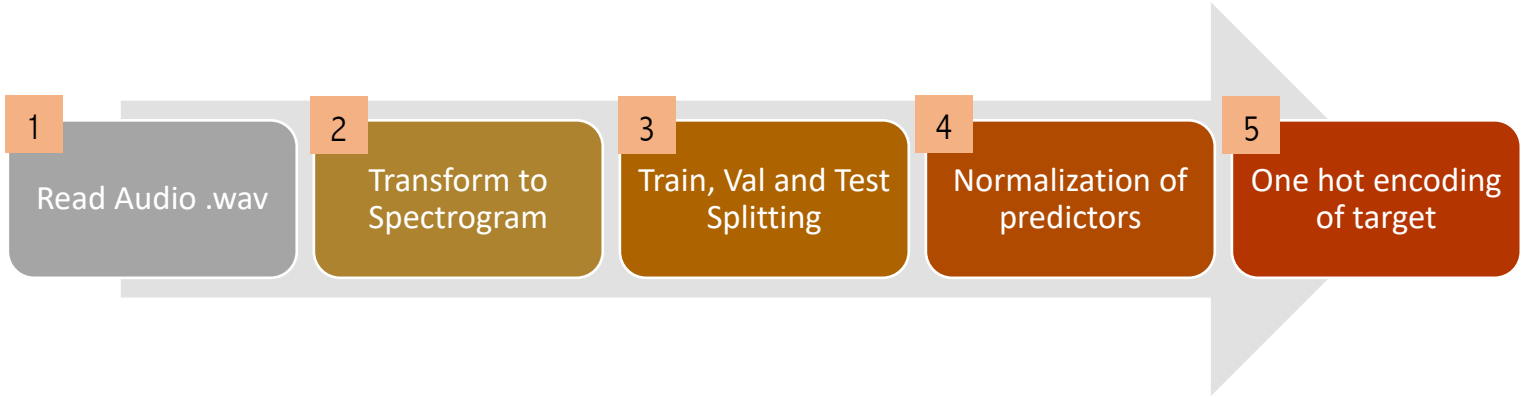
Data Description and Sources

- Dataset comprises audio recordings of spoken numbers along with their corresponding text-based representations.
- Data source : https://github.com/Jakobovski/free-spoken-digit-dataset
- Recording Detail:

No	Number of Speakers	Gender	Digits	Recording @Digit	Total Recording	Purpose	Format
1	6	All Male	10	50	3000	Train, Validation, Test	.Wav
2	3	1 Male 2 Female	10	1	30	Out of sample test	.Wav

- Data No 1 are used for building the model and testing the model
- Data No 2 are supplementary to see the generalization of the previous model built for out of sample dataset

Data Exploration and Transformation [1]



1. Reading the individual .wav file and stored as data: numpy.array and sample_rate: int.
2. Data is transformed to spectrogram using matplotlib, saved as image and stored.
3. Image loaded and split into Train, Validation and Test Set, according to this scenario:
   - Scenario 1, we use data from all speakers and use 40 recording as train set, 5 recordings as validation set and 5 recordings as test set for each digit and each speaker.
   - Scenario 2: we use data from five speakers as the train (45 recordings) and validation set (5 recordings) for each digit, and each speaker, and use the recording of the remaining speaker as test set.
4. Predictors are normalized by dividing it with 255 (8 bit).
5. Target class is one-hot encoded.

Data Exploration and Transformation [2]

Predictor

Format_Name: 0_George_0.wav

Split(Format_Name)[0] – Digit

Split(Format_Name)[1] – Participant Name

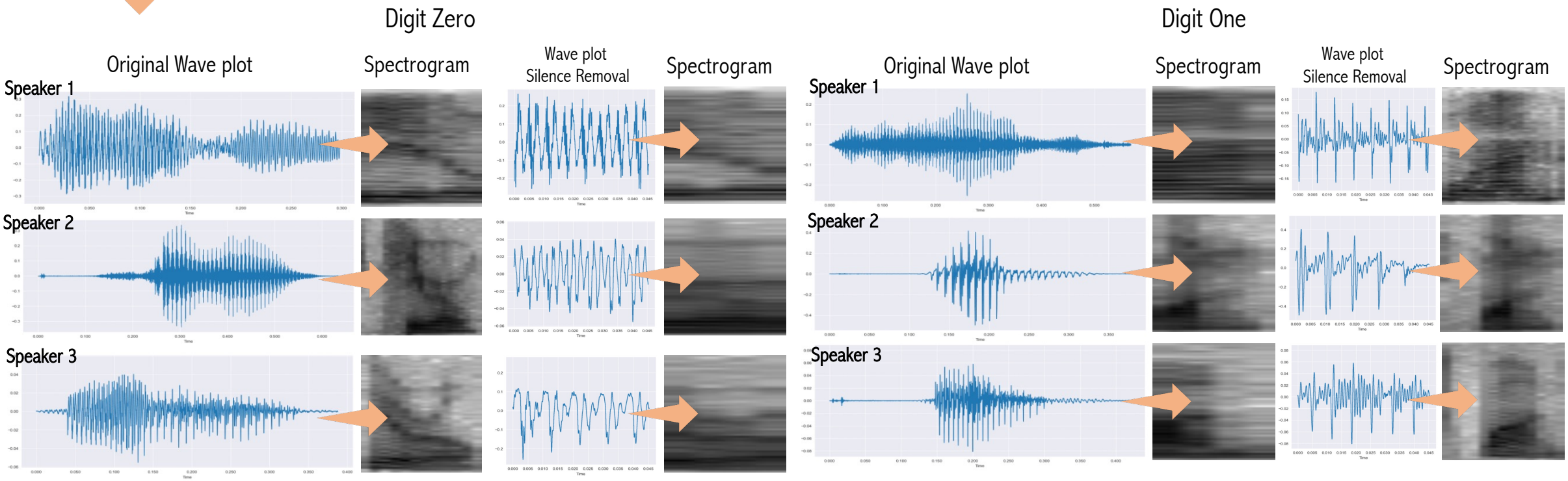
Split(Format_Name)[2] – Order of recording

Target Value

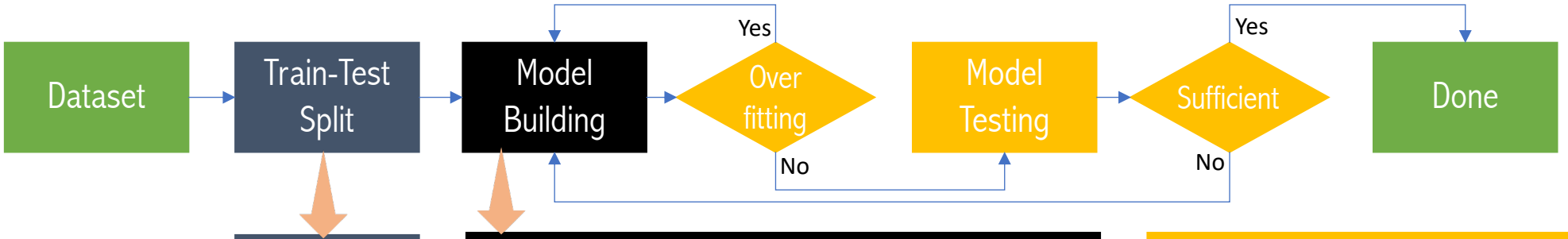
[Will be encoded]

Notes:

- Feeding the soundwave signal to the deep learning network does not give good result according to study literature.
- However, spectrogram is a useful representation of the signal in which CNN can extract similar pattern from the picture across different speakers.
- Two spectrogram is generated, from the original wave and from wave that has been filtered.



Model Building Process



Scenario 1
Train: 0.8
Val: 0.1
Test: 0.1

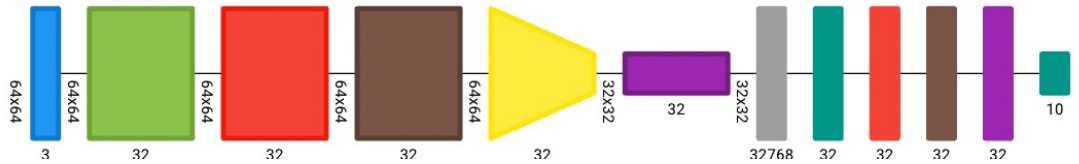
Scenario 2
Train: 0.73
Val: 0.1
Test: 0.17

1. Build Model for image classification
2. Using Convolutional Neural Network (CNN)
3. Handling overfitting with:
    1. Stacking Batch Normalization layer
    2. Stacking Dropout layer
    3. Hyperparameter Tuning using Optuna
    4. Early stopping

Metrics used
- Accuracy – Dataset is balanced

Overfitting observed when:
- Accuracy Train set >> Accuracy Test Set

Basic Architecture of the Model

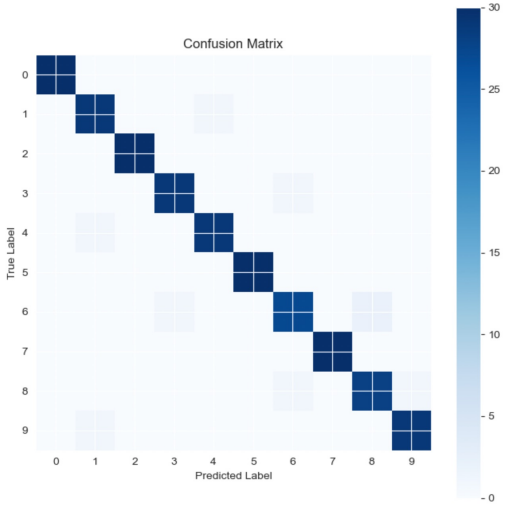


InputLayer Conv2D BatchNormalization Activation MaxPooling2D Dropout Flatten Dense

Variation of the model built:
1. Number of CNN layer and the neuron
2. The utilization of BatchNormalization and Dropout layer
3. Number of Dense layer

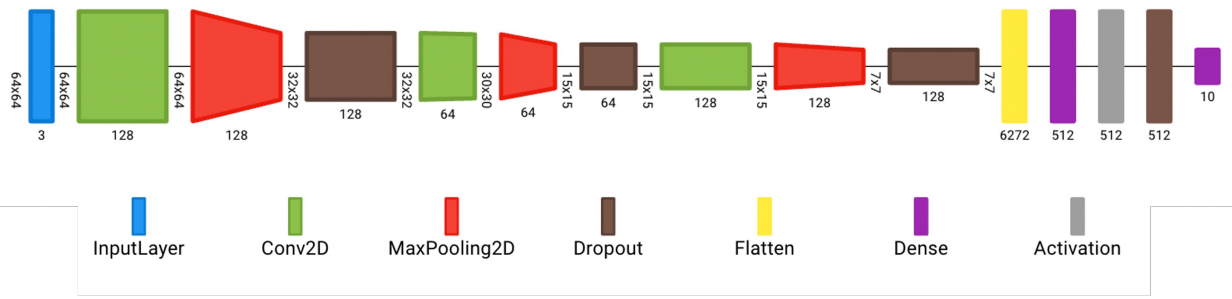
Model exploration: Scenario 1

No.	CNN	Batch Normalization	Max pooling	Dropout	Dense	Optuna	Overfitting	Test Accuracy (# Epochs)
#1	1 Layer (64)	Yes	(2,2)	0.25	2 layers (64, 10)	Yes	No	0.90 (#27)
#2	3 Layers (128,64,32)	Yes	(3,3)	0.25	2 layers (512,10)	No	No	0.9826 (#36)
#3	6 Layer (32,64,64, 64,128,128)	Yes	(3,3)	0.5	2 layers (512,10)	No	No	0.978 (#19)

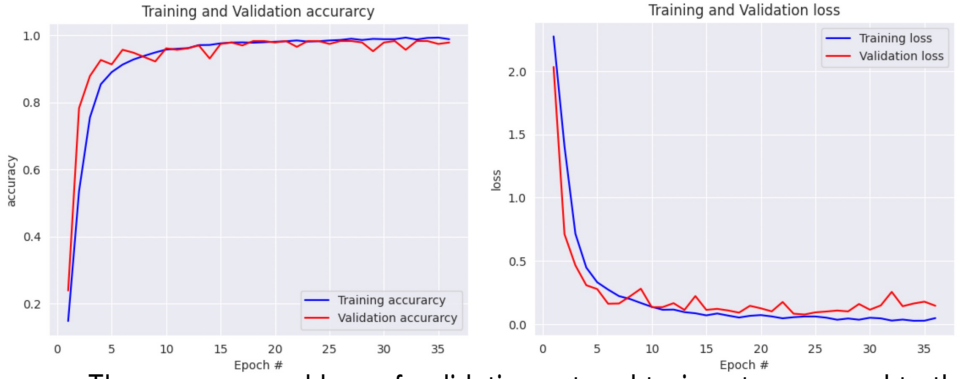


All: Optimizer: Adam, Learning Rate = 0.001 (#2 and #3) and 2.6e-4 (#1), Early stopping: Yes

Best Architecture: Model No #2



Graphic (Model #2)



Notes:

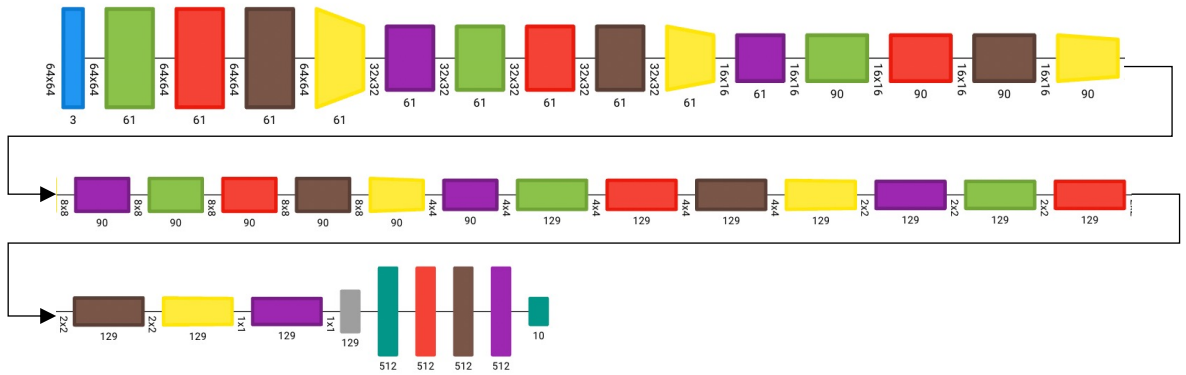
- The model we get from Scenario 1 with original spectrogram shows satisfying result with increasing accuracy when we add more layer to the model.
  – Thus, we do not explore Scenario 1 for wave silence removal spectrogram.

- The accuracy and loss of validation set and train set converged to the same number. However, if we continue the epoch, it will overfit the model. Thus, we use early stopping with bestparameter = True.
- The sign of instability in validation set remains despite of utilizing batch normalization layer

Model exploration: Scenario 2A

No.	CNN	Batch Normalization	Max pooling	Dropout	Dense	Optuna	Optimizers	Overfitting	Test Accuracy (# Epochs)
#1	3 Layers (128, 64, 32)	Yes	(2,2)	0.25	2 layers (64, 10)	No	AdamW	Yes	0.69 (#18)
#2	6 Layers (61,61,90, 90,129,129)	Yes	(3,3)	0.2	2 layers (512,10)	Yes	AdamW	Yes	0.86 (#24)

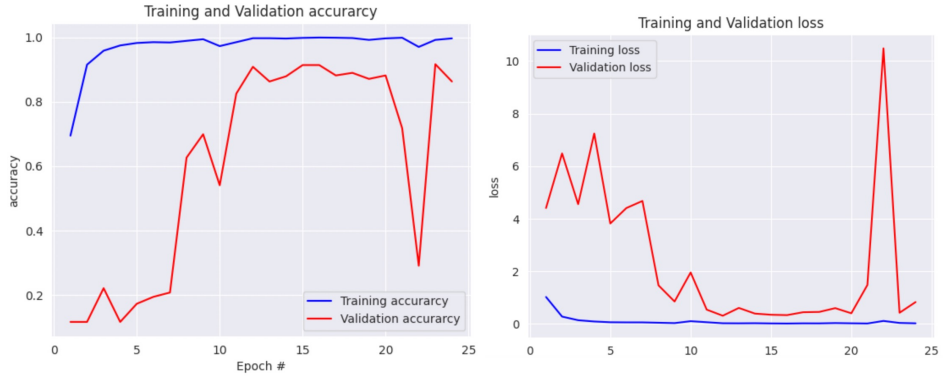
Best Model: Model #2 – CNN – 6 Layers



Notes:

- This scenario is performed in order to see the generalization of our model for speaker that has not been trained on the model. Clearly, new speaker is not recognized very well, compared to those that has been trained previously (as shown in training strategy in scenario 1).

Graphic (Model #2)

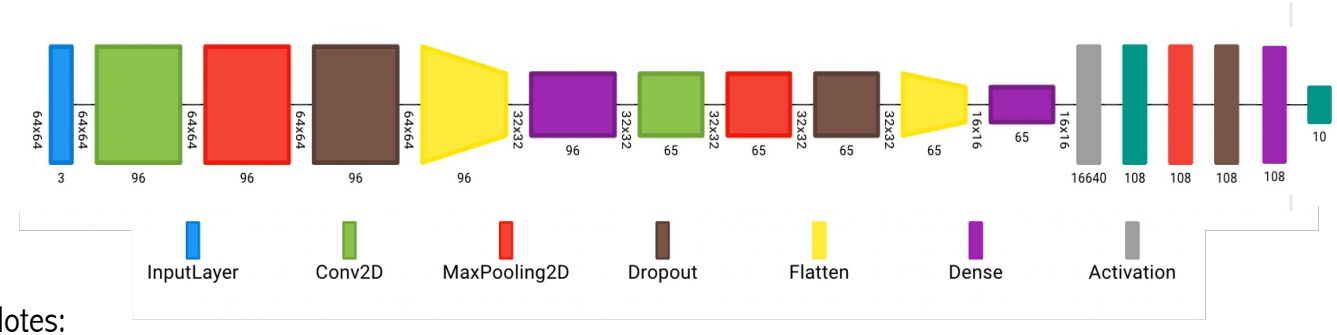


- The best model we found in this scenario is still severely overfitting despite of using Optuna for 20 trials, AdamW optimizers and Dropout layer.
- The surface of the graphic for validation are also not smooth despite of using BatchNormalization.

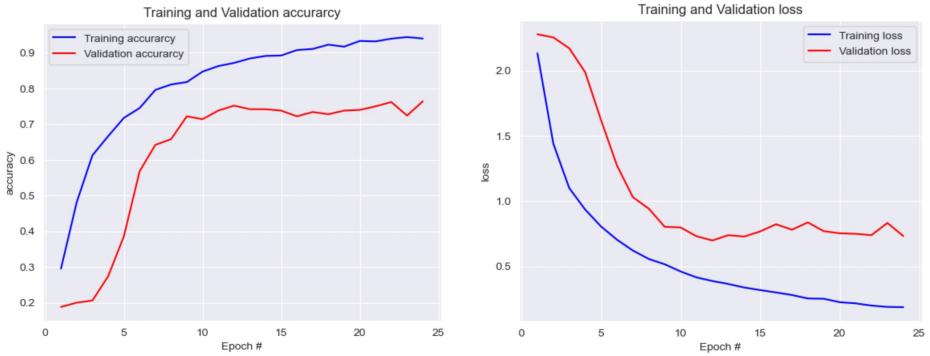
Model exploration: Scenario 2B

No.	CNN	Batch Normalization	Max pooling	Dropout	Dense	Optuna	Optimizers	Overfitting	Test Accuracy (# Epochs)
#1	3 Layers (128, 64, 32)	Yes	(2,2)	0.25	2 layers (64, 10)	No	Adam (LR: 1e-3)	Yes	0.75 (#21)
#2	2 Layers (96,65)	Yes	(3,3)	0.16	2 layers (108,10)	Yes	Adam (LR: 1.05e-5)	Yes	0.75 (#12)
#3	6 Layers (57,57,101, 101, 119, 119)	No	(3,3)	0.39	2 layers (512,10)	Yes	AdamW (LR: 9e-4)	Yes	0.69 (#28)

Best Model: Model #2



Graphic (Model #2)



Notes:

- This scenario is conducted to see whether transforming the model by removing the silent interval will improve the accuracy or overfitting issue.
– However this approach does not help resolve our previous issues as perhaps removing silent interval also removing particular pattern in the recording.

- The surface of loss and accuracy is quite smooth, however, our best model is severely overfitting despite of using hyperparameter tuning, decreasing the layer of the CNN, using AdamW and dropout layer.

Out of Sample Model Evaluation

Goal: Test the generalization of our model created in Scenario 1, 2A and 2B with out of sample dataset.

Metrics: Accuracy

Out of Sample Data Profile:

- Sound profile: 1 Male, 2 Female. Each subject record once for each digit.
- Dataset Transformation:

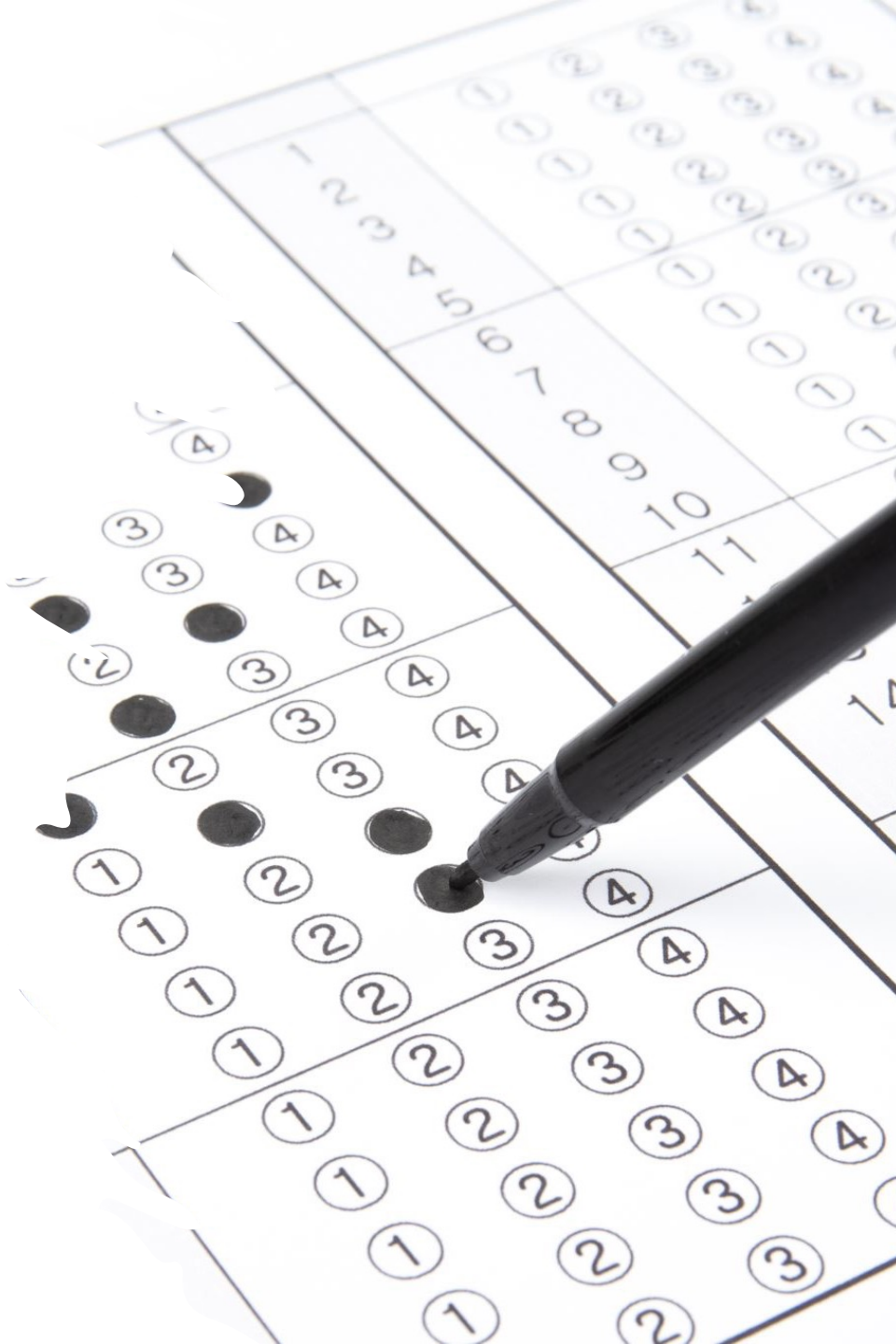
Original file (.ogg) → Converted (.wav) → Split per digit (.wav) → Spectrogram (@scenarios)

Results:

Model No	Accuracy
1	0.1
2A	0.067
2B	0.1

Conclusion:

- The model created in every scenario is not generalized enough that it can not recognize out of sampe data that comes from different modality of the recording, different gender and accent of the new speakers.



Lesson Learned

Data manipulation performed, i.e. Filtering silent interval, does not help model performance.

Plotting accuracy and loss across epoch help to conclude the fitness of the model and determine the right epoch.

Batch Normalization sometimes help smoothen the training surface so it can learn more stably.

AdamW makes the model perform better but it solely can not make the model escape the overfitting zone.

Optuna with Bayesian algorithm help hyperparameter tuning but to maximize the search high computational capacity is required.

Method of training and testing, variational.

Recommendation for Future Work

MAKING THE MODEL MORE GENERALIZED.

Further hyperparameter tuning may reduce overfitting of the model.

DESIGNING END TO END WORK.

End to end work from recording the data to model building can control variability in the environment, especially when generating the dataset.

FINE TUNING MODEL FOR NEW SPEAKER.

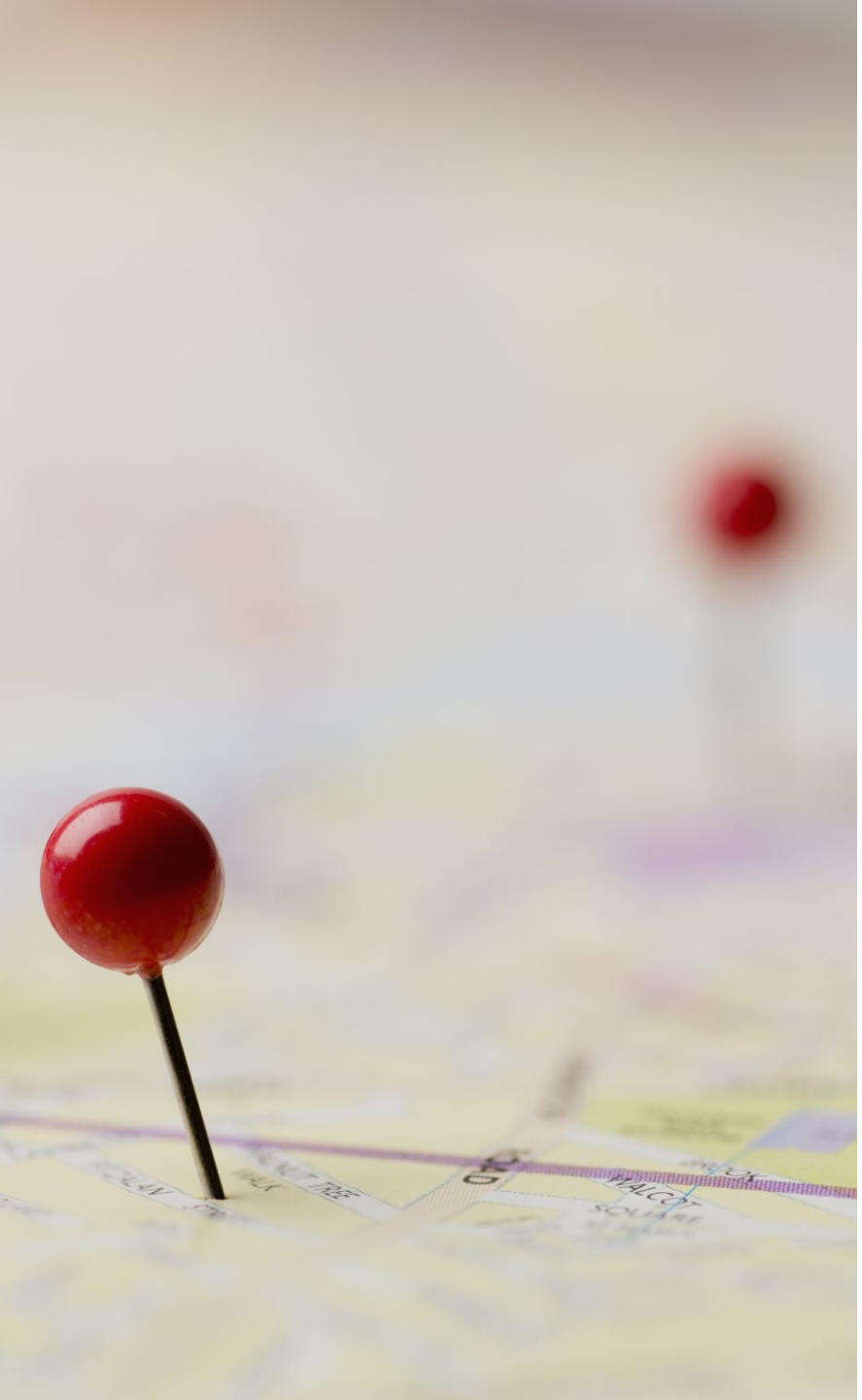
Consider transfer learning and fine-tune the pretrained model for every new speaker.

TRY DATA AUGMENTATION.

Consider creating variability in the dataset by shifting the time, adding white noise and etc. to the spectrogram

TRY MEL SPECTROGRAM.

Consider creating variability in the dataset by shifting the time, adding white noise and etc. to the spectrogram



Reference

- https://towardsdatascience.com/audio-deep-learning-made-simple-part-1-state-of-the-art-techniques-da1d3dff2504
- https://towardsdatascience.com/audio-deep-learning-made-simple-part-2-why-mel-spectrograms-perform-better-aad889a93505#:~:text=As%20we%20learned%20in%20Part,architectures%20developed%20for%20handling%20images.
- https://www.kaggle.com/code/sainathrk/spoken-digit-recognition-using-cnn-and-lstm#On-a-whole,-the-CNN-Model-with-6-CNN-Layers-and-Early-Stopping-gives-the-best-accuracy-of-98.66%25

Thank You

Rolamjaya Hotmartua

MSCA 31009 - Machine Learning and Predictive Analytics

Spring 2023

Github link for this project: https://github.com/Rolamjaya/ML_Final_Project_MSCA.git