Composite feature extraction approach for speech recognition

Emanuel Buttaci
Data Science and Engineering
Politecnico di Torino
Turin, Italy
s308589@studenti.polito.it

Giacomo Rosso
Data Science and Engineering
Politecnico di Torino
Turin, Italy
s309273@studenti.polito.it

Abstract—**In this report we introduce an approach to classification of audio recordings, specifically commands for voice assistants. Our approach relies on the extraction of statistical features, time domain features and frequency domain features from audio recordings, subject to proper preprocessing. The choice of machine learning models mainly fall onto ensemble classifiers and neural network, which indeed turn out to be the outperformers among all classifiers, when properly tuned. In the end, our strategy is capable of delivering a score ranging from 77% to 94% on the provided evaluation set, which highlights the successfulness of our approach.**

I. Problem overview

Audio classification has been a much investigated topic in the last decades. In this project we are dealing with a dataset of audio recordings of several people with different characteristics. These recordings represent command for a voice assistant like Alexa. The objective is to recognize which commands are expressed in each audio recording, that is a classification problem. The dataset is divided into two parts.

- **development set**, which contains about 9854 **labeled** records and will be used to train our model.
- **evaluation set**, which contains about 1455 **unlabeled** entries and will be subject to our classification pipeline.

II. Data exploration

Before diving into our problem approach, we need to explore how data is structured.

A. Data attributes

Each record is composed as follows.

- **path**, the path of the audio file associated to the recording
- **gender**, that is the gender of each speaker, either male of female.
- **ageRange**, the range of age of each speaker.
- **speakerId**, that is the identifier of each speaker (there could be multiple recordings associated to the same speaker).
- **First Language spoken**, which we assume to be the mother tongue of each speaker.
- **Current language used for work/school**, which we expect to be the language effectively used in each recording.
- **Self-reported fluency level**, namely the level of expertise of each speaker in mastering the language used in the recording.

B. Missing values

No missing value is present and we do not require any imputation strategy.

C. Evaluation problem considerations

Exploring the distributions of attributes emerges that there are different current language spoken, where English appears to be the most widely spoken, followed by French, and native speakers are the vast majority. Furthermore, we can count up to 87 different speakers, but the recordings are not evenly distributed among them. Concerning gender, there is an almost uniform distribution as expected, while age range distribution is more sparse as speakers are older, which means most of the speakers are young people. However, exploration of evaluation set is very relevant to us, it gives us an idea of how our problem is constrained with respect to the dataset attributes.

Surprisingly enough, as we can see both in Figure 1 and in Figure 2, the evaluation set is composed only of native English speakers, which is a very strict constraint on our problem. Anyway, this could be an advantage to us, since native speakers will not suffer any foreign accent in speaking English. From this consideration, we take our first decision, which is to train whatever model we are going to construct only on English speakers, thus removing all foreign language speakers from development set. Potentially, in case there would be multiple languages in the evaluation set, we may think of a strategy which trains multiple classifiers, each one corresponding to a different language. In our case, we discarded such approach since English is the unique language appearing in the evaluation set.

D. Prior features removal

By removing foreign speaker, we lose any discriminant information about the language spoken, which is uniquely English. Therefore we also decide that any attribute about language will be discarded since it is irrelevant to our approach. Using the same consideration, we also ignore the speaker level information, which is unique in the evaluation set and

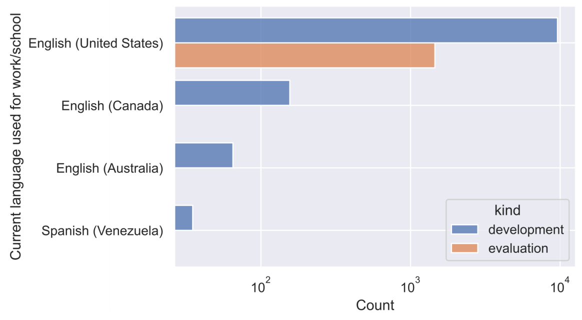

Fig. 1. Distribution of spoken languages in development and evaluation set

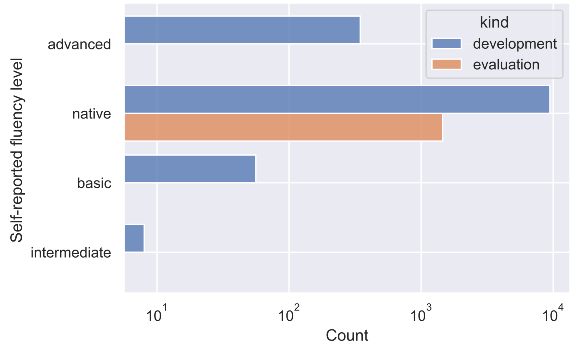

Fig. 2. Distribution of fluency levels in development and evaluation set

therefore constant. Anyhow, we also decide to keep all kinds of speakers from training set since they may capture different accents within the English language.

Evidently, none of the speakers from evaluation set is also part of the development set, which leads us to indeed discard the ‘speakerId‘ information, which cannot relate any information between training set and evaluation set. Only ‘gender‘ and ‘ageRange‘ remain, which could be easily encoded by means of one hot encoding. Yet, we think that spectral information extracted from each voice recording may encode the difference between voices belonging to both genders and different age ranges. Thus, we also eliminate ‘gender‘ and ‘ageRange‘ attributes. Indeed, by training any model using such attributes we had proved how scores get worse when they are included.

E. Audio recordings exploration

This approach leaves us only the audio recording, which, in fact, will be the effective object of feature extraction. We exhaustively motivated why we genuinely believe that audio recording may encode all meaningful information regarding our problem. Audio recordings are encoded using wav format, that is they are stored as sequences of 16 bits integers holding amplitudes in time domain.

Evidently, some minority of recordings belonging to the development set share a higher sample rate of 22050 Hz, which thereby has to be adjusted in order to have uniform recordings. The strategy which is going to be employed is a down resampling of such recordings to the mostly spread sample rate of 16000 Hz.

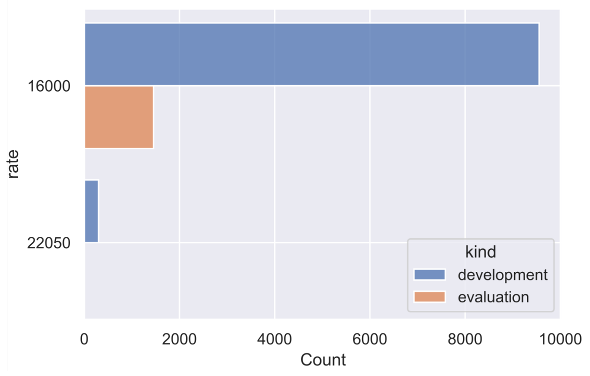

Fig. 3. Sample rates across development and evaluation set

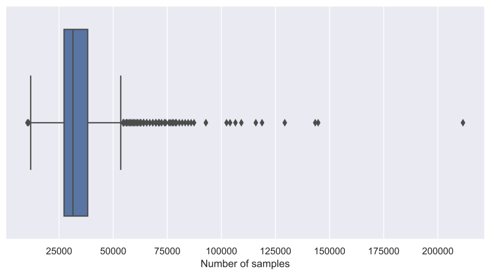

Fig. 4. Boxplot of audio recordings lengths

As we can see in Figure 4, the lengths of audios are quite different and they range from roughly 10k samples to 441k samples, though the distribution of lengths is quite centered towards shorter lengths which suggests that excessive lengths may be outliers. This could be a clue suggesting the presence of long leading and trailing silences in some longer recordings. However, the 95% of audios’ lengths fall within about 57k samples.

TABLE I
PERCENTILES OF AUDIO RECORDINGS LENGTHS FOR 1600 HZ RATE

Percentile	Length as number of samples
0	10403
5	21845
25	27307
50	31403
75	38229
95	50517
100	211627

F. Target Inspection

In the classification problem the target label (speakers' commands) is made by labels 'action' and 'object'. Since we think that some combinations of 'action' and 'object' may be semantically meaningless, for example 'change language' + 'volume', we decide to combine from the very beginning both targets into a single target which directly encodes the intent. The encoding follows the same rule that is expected in the evaluation. Therefore, we are going to construct the overall target by means of a string concatenation, namely the intent. We end up with 7 different classes, which constitutes our final target for the classification problem.

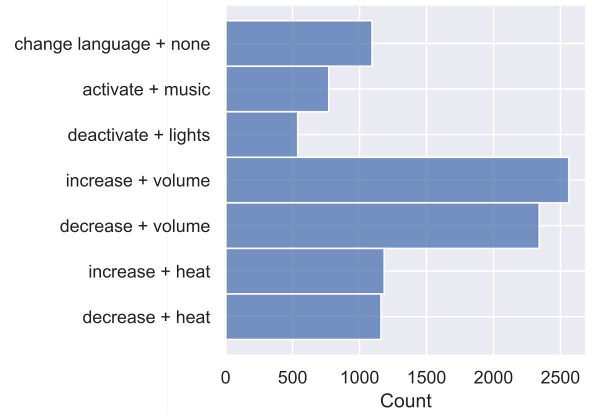

Fig. 5. Distribution of intent across development set

As it is seen in Figure 5 target categories are not uniformly distributed. Besides, each attempt to balance the problem, such as sampling the same amount of audio recordings with respect to each target category, returns worse results.

III. Preprocessing

As we already stated, the provided audio files are encoded in wav format, that is a sequence of 16 bits integers. Specifically, all the recordings have been sampled at frequency 16000 Hz. Before proceeding, each audio recording is normalized in range [-1, 1] using 32768, which is the maximum feasible (absolute) value for any amplitude (largest 16 bits integer).

The first step consists in cleaning each recording from irrelevant silence by cutting off those amplitudes which are below a certain threshold. Such threshold has been set to 20 dB. The choice has been made in order to preserve the overall resemblance of the trimmed audio with respect to the original one as can be seen in Figure 6 and Figure 7.

Then, we have to uniform the amount of samples across all the recordings, that is the length. To us, the simplest way to achieve this consisted in computing the index of the last non zero amplitude across all recordings, in order to exclude the trailing silence. Hence we compute the 95% percentile of these lengths. This value will be the final number of samples chosen for each recording. Those recordings which are longer in time are cut (because of our choice they are 5% of all), while shorter recordings are filled with trailing zeros.

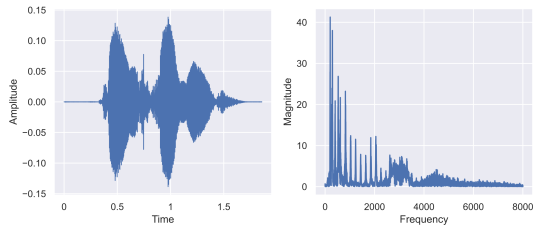

Fig. 6. Amplitude and Fourier transformed of raw audio[1]

Finally this results in a value which is roughly 30000 samples, corresponding to a duration of 1.9 seconds since the sampling frequency is 16000 Hz.

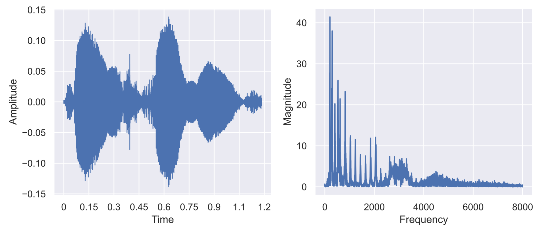

Fig. 7. Amplitude and Fourier transformed of processed audio

IV. Feature Extraction

Feature extraction is the most delicate and important step in the construction of our pipeline. We known from the beginning that the choice of features is what really matters and defines the accuracy of any model we may build. Thereby, we are going to extract three macro groups of features from audio recordings. Each group is evidently capable of encoding relevant information when modelling each recording's wave signal.

A. Statistical features

Statistical features are information computed in time domain. Particularly, we divide each recording $x(t)$ in time windows of the same length. Each time window will have a duration of 50 milliseconds, whose value was chosen after some attempts. So, each window will contain exactly $W = 800$ samples and each recording, due to a length of about 30000 samples, will be partitioned into less that 40 windows. Now, for each n-th window, we compute some statistical information.

B. Time domain features

Time domain features are another set of meaningful features related to the shape of each recording in the time domain. They directly followed statistical features in this sense. Again, they are computed with respect to each time window.

[1]Audio file is *0a3129c0-4474-11e9-a9a5-5dbec3b8816a.wav*

TABLE II
STATISTICAL FEATURES

Feature	**Value**	**Granularity**
Mean	$\frac{1}{W}\sum_i x(i)$	window
Standard deviation	$\sqrt{\frac{1}{W}\sum_i (x(i)-\bar{x})}$	window
Skewness	$W\dfrac{\sum_i (x(i)-\bar{x})^4}{[\sum_i (x(i)-\bar{x})^2]^2}$	overall
Kurtosis	$W\dfrac{\sum_i (x(i)-\bar{x})^3}{[\sum_i (x(i)-\bar{x})^2]^{\frac{3}{2}}}$	overall

TABLE III
TIME DOMAIN FEATURES

Feature	Value	Granularity
Zero crossing rate	$\dfrac{\sum_i \lVert \mathrm{sgn}(i+1)-\mathrm{sgn}(i)\rVert}{W}$	window
Root mean square	$\sqrt{\dfrac{1}{W}\sum_i x(i)^2}$	window
Peak-peak distance	$\max(x(\cdot)) - \min(x(\cdot))$	window
Peak	$\max(\lVert x(\cdot)\rVert)$	window
Crest factor	$\dfrac{\max(\lVert x(\cdot)\rVert)}{\sqrt{\dfrac{1}{W}\sum_i x(i)^2}}$	window
Shape factor	$\dfrac{\sqrt{\dfrac{1}{W}\sum_i x(i)^2}}{\dfrac{1}{W}\sum_i \lVert x(i)\rVert}$	window
Impulse factor	$\dfrac{\max(\lVert x(\cdot)\rVert)}{\dfrac{1}{W}\sum_i \lVert x(i)\rVert}$	window
Margin factor	$\dfrac{\max(\lVert x(\cdot)\rVert)}{\dfrac{1}{W}\sum_i \sqrt{\lVert x(i)\rVert}}$	window

The zero crossing rate is related to how many times the signal changes its sign within each time window. The root mean square is also well known as the energy of a signal in signal analysis.

C. Frequency domain features

Now we concern about the spectral domain. Each wave signal can be transformed from the time domain into the frequency domain. The transformation is performed through the discrete Fourier transform (DFT), which delivers a spectrum of magnitudes associated to each frequency. In particular, we extract only the part with real frequencies.

$$X(k) = \mathcal{F}\{x(t)\} = \sum_{t=0}^{T} x(t)\exp\Big(-i\frac{2\pi}{T}kt\Big)$$

On the spectrum of frequencies the main features are considered and extracted. Like for other features, we consider several time windows.

TABLE IV
FREQUENCY DOMAIN FEATURES

Feature	**Value**	**Granularity**
MFCC	Mel-frequency cepstral coefficients	window
Mel spectrogram	Spectrogram with frequencies in Mel scale	window
Spectral centroid	Weighted mean of frequencies using magnitudes	window
Spectral rolloff	Frequency below which 85% of spectrum magnitudes are located	window
Spectral bandwidth	Variance with respect to the spectral centroid	window
Spectral flatness	Quantifies how much noisy the wave signal is (1.0 is close to white noise)	window
Spectral kurtosis	Tailedness of the distribution of frequency spectrum	window
Spectral skewness	Asymmetry of the distribution of frequency spectrum	window
Spectral flux	Measures how quickly power spectrum changes across a signal	window
Spectral contrast	Difference between spectral energy peak and minimum	window
Spectral entropy	Uncertainty of power spectrum according to Shannon entropy	window

Mel-frequency cepstral coefficients are very popular when doing audio analysis and classification. Such coefficients are extracted from a complex computation which involves Fourier transform and discrete cosine transform on logarithmic power spectrum in Mel frequency. Mel scale is a logarithmic scale which is very close to the human perception of frequencies.

$$[f]_{\text{mel}} = \begin{cases} f & \text{if} \quad f \leq 1000 \text{ Hz} \\ 1127 \cdot \ln(1 + f/700) & \text{if} \quad f > 1000 \text{ Hz} \end{cases}$$

Since Mel scale is human-like, the usage of Mel-frequency cepstral coefficients is a powerful tools when doing audio recognition. All other features are extracted from the spectrum of magnitude of frequencies or from the power spectrum. They encode information about the shape of frequency distributions within each time window. Hence, they may be very good predictors for our task.

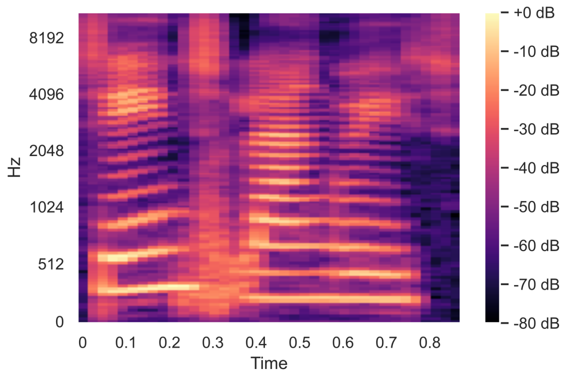

Fig. 8. Mel spectrogram of preprocessed audio

V. Classification

A. *Implementation of FFNN*

Leaded by the thought of not missing any opportunity in making more accurate predictions, we decided to implement a feed forward neural network by stacking different layers. The choice related to the number of layers, neurons and epochs has been made very practically by making different attempts in assessing the performance over the development set. Anyway, such investigation has been avoided here. Also, we explicitly enabled the multi processing computation in order to speed up the classification.

B. *Model performance evaluation*

Classification is approached by trying different models, including the neural network implemented before. The classification is carried out on a representative sample of 1000 elements, in order to avoid using the whole dataset, which would result in huge computational times. The training set is 80% of the original sample, therefore consisting in 800 records. Furthermore, audio signals are preprocessed according to the pipeline we constructed. Finally, a standardization step is prefixed to those models which otherwise would suffer the absence of a common scale within the data, namely the logistic regression, support vector machine, nearest neighbours and neural network.

Not very surprisingly, ensemble models performed better than remaining ones. However, all scores are similarly aligned.

[2]The implementation using Keras library suffers from the lack of setting a random state for creating reproducible results. So, as it can be seen in practice, scores obtained using the neural network will often be similar tough not the same.

TABLE V
BASE SCORES ON MULTIPLE CLASSIFIERS

Model	**Sample score**
LogisticRegression	0.465
RandomForestClassifier (RF)	0.460
HistGradientBoostingClassifier (HGB)	0.520
SVC	0.455
KNeighborsClassifier	0.425
FeedForwardNeuralNetworkClassifier (FFNN)	0.500[2]

C. *Feature selection*

The amount of features extracted from the audio recordings is relatively large. On a dataset of roughly 9500 instances, almost 2000 features are computed for each of them. Computational costs are to be taken into account with such a heavy training. Therefore different strategies have been tested in order to decrease the dimensionality of our data. In the most simple case a variance threshold was used, but it did not improve the results significantly, since the selection criterion was based on a statistical property, namely the variance, rather than the true meaningfulness of the feature with respect to the target. Another employed strategy consisted in using f-scores from builtin ANOVA statistical tests. Relative performance was marginaly improved both on development set and evaluation set. However, the technique which delivered the most promising performance was based on features importances from a fitted ensemble model. In practice, feature selection using a trained random forest or extra trees classifier, according to an importance criterion, did outperform other techniques and brought the most significant improvement partly on development set and mainly on the evaluation set. Indeed, in the end we employed an extra trees classifier with 750 trees and entropy criterion for the computation of features importances. This tree based ensemble is peculiar since the decisions on which attributes to split are made randomly instead of optimistically, as it happens for any random forest.

VI. Optimization

We decide to investigate and improve the performance of three models which seem to be promising, namely ‘RandomForestClassifier‘, ‘HistGradientBoostingClassifier‘ and ‘FeedForwardNeuralNetworkClassifier‘. Since the dataset is relatively large and feature extraction is heavily expensive in terms of computational time, we decided to select a subset of 1000 elements from the original development set. We also perform prior feature extraction and not a part of the classifier pipeline, in order to speed up the process. Note that optimization is performed by selecting the best scoring model on average. This means that the score of a certain configured model is averaged over a 5-fold partition.

Random forest

TABLE VI
SEARCH GRID OF HYPER PARAMETERS

Model	Parameter	Values
RF	random_state	0
	n_estimators	100, 250, 500, 750
	criterion	gini, entropy, log_loss
HGB	random_state	0
	learning_rate	0.075, 0.1, 0.25
	loss	log_loss
	max_iter	100, 125, 150
FFNN	activation	relu, tanh, sigmoid
	optimizer	adam, sgd, adadelta
	epochs	100, 150, 200
	batch_size	128

The random forest model is a powerful ensemble model which trains independently several decision trees, each one on a random sample of the dataset. Moreover, each tree selects only a subset of features to evaluate each split. Therefore, result trees are decorrelated. This technique has the main advantage of being robust to overfitting. In our tuning we focus on the quantity of trees and the split criterion chosen at each node.

Gradient Boosting

In order to capture some relationships within the data in more depth, we decide to optimize the gradient boosting model. Like the random forest, it makes use of several trees. However the training process is completely different. Whilst the random forest independently fits each decision tree, possibly in parallel, instead the gradient boosting model trains each decision tree sequentially, thus it is not capable of parallelizing. Anyhow, its training benefits from a corrective process in which each new tree is trained on the residual error. Therefore this tree is constructed in order to correct the prediction error of previous trees with respect to the target. The residual error and correction step are applied by means of gradient computation and descent. The learning rate and number of epochs is investigated during optimization in this case.

Feed forward neural network

The last model which we decide to optimize is the feed forward neural network, according to the architecture chosen for the implementation. A neural network has a very high capability of learning the non linear and peculiar relationships from the data and the target. Such training is performed again through a variant of stochastic gradient descent. In practice the network is trained in order to minimize the prediction errors and the neurons' coefficients are updated accordingly by means of gradient computation and descent. In this case we decide to try different activation functions involved in computing each neuron's output. Besides, differently from tree based models, here we need to scale the data since linear combination of features are effectively calculated before feeding activation functions.

VII. Results

Once we found out which are the optimized configurations for our models, let us compute the average score on the entire development set by means of a cross validation strategy. Such strategy partitions the dataset in 5 groups and uses 4 of them to train each model and the remaining one for the validation. Thus, given each tuned model, five scores are computed for each group. The average score delivers us an idea of the overall performance of the optimized model. Besides, given that results get worse applying feature selection based on variance, we opted for the one based on feature importances measured by extra trees classifier. Differences between table VII and table VIII show the improvements achieved by implementing such technique.

TABLE VII
SCORES OF TUNED MODELS ON DEVELOPMENT SET

Model	Parameter	Value	Score
RF	random_state	0	0.654 ± 0.054
	n_estimators	750	
	criterion	entropy	
HGB	random_state	0	0.696 ± 0.049
	learning_rate	0.1	
	loss	log_loss	
	max_iter	150	
FFNN	activation	relu	0.760 ± 0.057
	optimizer	adam	
	epochs	200	
	batch_size	128	

Evidently, the neural network outperforms the other tuned models, while the gradient boosting seems better than the random forest. Since the neural network uses gradient descent in order to minimize the error function when fitting the data, this could lead to a deeper interpretation of the relationships between features, particularly the non linear ones. The same applies to the gradient boosting classifier, apparently like a random forest, which is trained on the minimization of residuals, namely prediction errors. In fact, the random forest constructs many independent trees and such independence may be the cause of missed relationships in the data, which is solved by the gradient based mechanism behind the gradient boosting model.

A. *Submission*

The highest submission scores on the leaderboard are possibly obtained by tweaking some hyper parameters after the optimization step, that is the only way we figured out to improve the scores. In particular the highest score for 'FeedForwardNeuralNetworkClassifier' is 0.930 which is due to the randomness present in the training of the model. The underlying implementation, using Keras library, does not enable setting a random state for reproducible results. Therefore we made use of a voting classifier constructed

[3]Extra trees classifier with 750 trees and entropy split was trained onto development set and its features importances were employed for selection.

TABLE VIII
SCORES OF TUNED MODELS ON DEVELOPMENT SET AFTER FEATURE SELECTION [3]

Model	Parameter	Value	Score
RF	random_state n_estimators criterion	0 750 entropy	0.661 ± 0.057
HGB	random_state learning_rate loss max_iter	0 0.1 log_loss 150	0.688 ± 0.052
FFNN	activation optimizer epochs batch_size	relu adam 200 128	0.766 ± 0.053

with an odd number of neural network, 17 voters to be precise, for simulating a majority voting mechanism. In this way we could achieve the average performance of the neural network without much randomness. Indeed we got the highest score of 0.942. Regarding ‘HistGradientBoostingClassifier‘, the peak obtained on the evaluation set was 0.854. Finally, the ‘RandomForestClassifier‘ touches its best score of 0.797 without further tweaking after optimization.

VIII. DISCUSSION

We strongly believe that our approach and experimentation to this classification problem has been quite successful. Our belief mainly lies in the preprocessing and feature extraction steps, which are the true cores of the entire pipeline. The consideration of statistical features, time domain features and spectral features in frequency domain turned out to be an exhaustive and meaningful way of performing feature engineering from original audio recordings. Furthermore, the partitioning of each wave signal into time windows also yielded to a more granular characterization of our audio recording. Overall, we ended up with about 2000 extracted features, which is a large amount, thus the implementation of features selection helped in distinguish the most significant ones (roughly 950). Since the process of feature engineering is computationally heavy, we think that a prior extraction and storage of features from training set would significantly improve the development speed and focus. Finally, using ensemble models and the much popular neural network, we were able to reach considerable accuracy scores varying from 61% to 76% on the development set and from 79% to 94% on the evaluation set, which again confirms that our approach was right in combination with such powerful classifiers. Possibly, other strategies may take place for further investigation, and we mainly were interested in the concept of wavelet transform, its coefficients and a more profound strategy of building the neural network inner structure. These could be some enhancements to our current approach presented in this project.

REFERENCES

[1] Sim, J., Min, J., Kim, D., Cho, S. H., Kim, S., Choi, J.-H. (2022). “A python based tutorial on prognostics and health man- agement using vibration signal: signal processing, feature extraction and feature selection. Journal of Mechanical Science and Technology”, pp. 4083-4097, April 2022. http://doi.org/10.1007/s12206-022-0728-z

[2] P. Mahana, G. Singh (2015). “Comparative Analysis of Machine Learning Algorithms for Audio Signals Classification”, International Journal of Computer Science and Network Security, VOL.15 No.6, June 2015.

[3] R. Lenain, J. Weston, A. Shivkumar, E. Fristed (2020). “Surfboard: Audio Feature Extraction for Modern Machine Learning”, May 2020. https://arxiv.org/pdf/2005.08848.pdf

[4] D. Gerhard (2003). “Audio Signal Classification: History and Current Techniques”, Technical Report TR-CS 2003-07, November 2003. https://www.uregina.ca/science/cs/assets/docs/techreports/2003-07.pdf

[5] Wikipedia contributors (2022). “Mel-frequency cepstrum”. In Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Mel-frequency_cepstrum&oldid=1106625289

[6] Wikipedia contributors (2022). “Mel scale”. In Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Mel_scale&oldid=1117532023

[7] Wikipedia contributors (2022). “Spectrogram”. In Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Spectrogram&oldid=1127110588

[8] Wikipedia contributors (2022). “Discrete Fourier transform”. In Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Discrete_Fourier_transform&oldid=1130001170

[9] Pedregosa et al. (2011). “Scikit-learn: Machine Learning in Python”, JMLR 12, pp. 2825-2830.

[10] Chollet, F., others (2015). “Keras”. GitHub. https://github.com/fchollet/keras

[11] Alexey Natekin1 and Alois Knoll (2013). “Gradient boosting machines, a tutorial”, DepartmentofInformatics,TechnicalUniversityMunich, December 2013. https://www.frontiersin.org/articles/10.3389/fnbot.2013.00021/full