Machine Learning in Multimedia Data: Vehicle Tracker

Evangelia Baou
University of Piraeus

Michael Zouros
University of Piraeus

July 6, 2021

Abstract

In this paper, we are studying the application of various Machine and Deep Learning Techniques, in our effort to classify audio signals emanated from various vehicles to a specific label, which denotes the approximate number of vehicles that passed from a static place during a specific recording time frame. Specifically, we consider three different approaches of classification. The first approach concerns the extraction of important frequency and time domain features (eg. Spectral Centroid, Zero Crossing Rate) and the use of an SVC classifier, the second one the production of Mel Spectograms and the use of a CNN Architecture and the third one the extraction of MFCC features and the use of an RNN-LSTM Architecture. Based on the study, we experiment on a number of different kernels, Regularization Parameters (C) and Kernel's Coefficients (gamma) for the SVC algorithm and on a number of different architectures for the CNN and RNN-LSTM models. We assume that, due to our small and noisy dataset, the CNN and RNN-LSTM architecture will prove less robust than the SVC algorithm. We expect the results of the prediction to be satisfactory on both approaches. The results suggest that the SVC algorithm outperforms its DL counterparts, on all of our algorithm's lifecycle stages (train, test, predict), with RNN-LSTM coming second, while bearing very good results in training and testing, but having a hard time during prediction.

I. Introduction

There is a total of 1.4 billion vehicles on the planet, with studies suggesting this number will surpass 2 billion by 2035. This ever-increasing number of vehicles creates an urgent need for monitoring their everyday movement. Applications like vehicle detection and/or classification can be proven very useful on various vehicle-related fields, like traffic control, traffic moderation, autonomous vehicle behaviour programming, vehicle tracking, and many more. Those applications can be implemented through the process of a rich array of technologies such as video, radar, magnetic and acoustics. This paper concentrates on the extraction of useful and meaningful information through the use of acoustic information. The reason is that vehicles detection and classification seems to be more robust through sound, from the moment sound is not affected by any alteration of the lighting conditions (night time, fog, bad weather), in contrast to an image/video.

II. Methods

i. Problem Presentation

The purpose of this study is to collect audio signals from different static places near a road, extract useful information from these signals and implement different types of Machine and Deep Learning algorithms, in order to train them to recognize, given a new audio signal, how many vehicles have passed during the signal's duration. Roads can be of any type (one way, 2 lanes, boulevards, highways, etc.) and vehicles can be of any type (motorcycles, cars, vans, trucks, buses, etc.) and size, as long as they have an engine (eg. bicycles are not calculated). One of the most challenging problems during the implementation was the collection of the dataset, in which, in contrast with most studies, we decided to follow a more realistic path (eg. we also recorded on heavy traffic roads, during heavy traffic hours). The script used in this work is implemented in Python 3.6 and the visualization of the graphs has been done with the help of matplotlib library.

ii. Data Collection and Preparation

Data collection took place near various types of roads. The recordings were made via our smartphones' microphones and had a duration that lasted just over 30 seconds. Approximately over 200 different samples were collected, but only 172 were compliant and used for this study. Each sample was named in accordance to its label (numeric), indicating the vehicles that passed by, plus an incremental index, indicating the number of the recording (eg. `9_recording17`). Data preparation took place on the Audacity application and included the signal trimming to exactly 30 seconds, noise reduction (eg. reduce the volume of bird singing or people talking), stereo to mono transformation (so to have a balanced dataset, because one of our smartphones was producing mono signals) and last but not least, conversion to WAV for better quality.

iii. Feature Extraction and Data Augmentation

Feature extraction was different for each of the three approaches.

Concerning the CNN algorithm, we created the Mel Spectograms of our audio signals with the help of the librosa library. We then proceeded and achieved data augmentation via filtering / masking our Mel Spectograms on both of their axes (mel scale, time). With this technique we managed to double the size of our initial dataset to 344 audio signals.

Concerning the RNN-LSTM algorithm, we extracted the Mel-frequency Cepstral Coefficients (MFCCs) from our audio signals. We then generated new audio files from the original, using various sound augmentation techniques. In particular, we used White Noise Addition, Time and Pitch Shifting.

Finally, regarding the SVC algorithm, we extracted and experimented with various different features, starting with MFCCs and STFTs. Later, we experimented with more important to our task features, like the Spectral Centroid and the Bandwidth for the Frequency domain and the Root Mean Square Energy and Zero Crossing Rate for the Time domain. We also achieved data augmentation through Pitch Shifting, thus tripling our initial data.

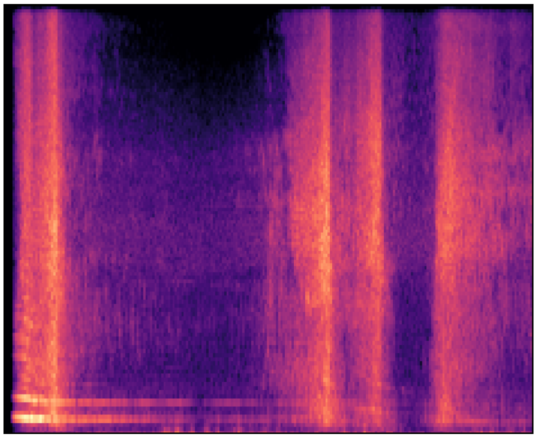


Figure 1: *Mel Spectogram*

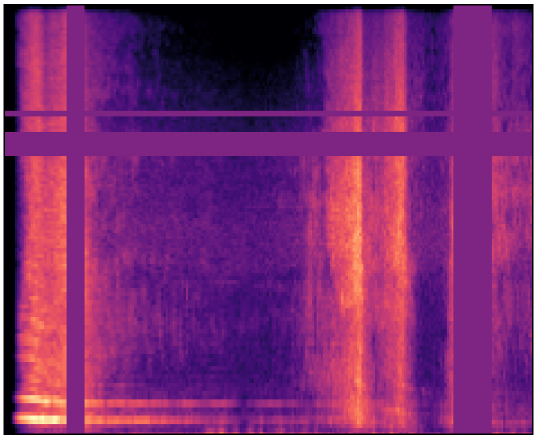


Figure 2: *Masked Mel Spectogram*

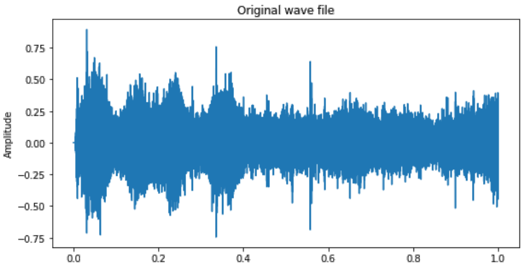


Figure 3: *Original Soundwave*

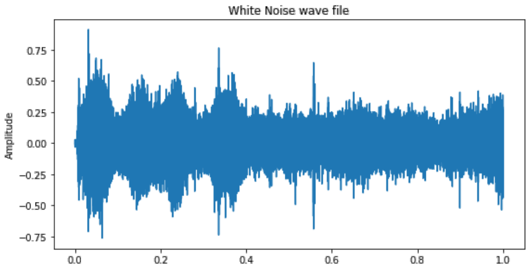


Figure 4: *White Noise Added Soundwave*

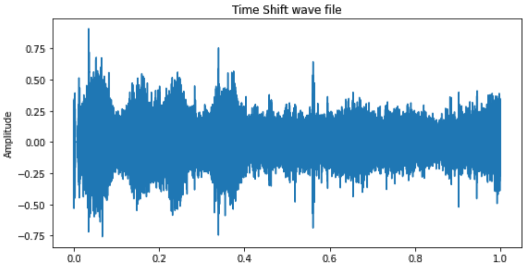


Figure 5: *Time Shifted Soundwave*

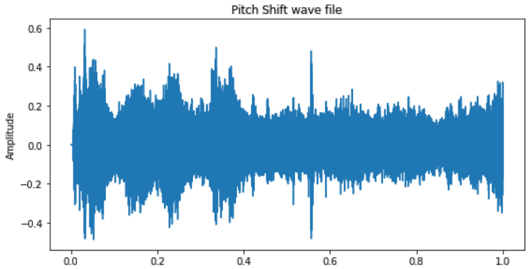


Figure 6: *Pitch Shifted Soundwave*

iv. Algorithms

For the CNN model, we produced each signal's Mel Spectogram and (later) Masked Mel Spectogram. We saved the plots as images (PNG), we loaded them as features and we standardized them. We then trained with different CNN model architectures, with different values for batch sizes, epochs, activation functions, layers, etc.

For the RNN-LSTM model prior to data augmentation, the initial dataset was split into training and test parts and the augmentation process applied only on the training data. Then the Mel-frequency cepstral coefficients (MFCCs) and all the corresponding labels for all the files were extracted and stored in a NumPy array. The model was trained with different values for batch sizes, epochs, activation functions, layers, etc. Cross validation was used in both cases of neural networks.

Finally, for the SVC algorithm, from the data collected, we extracted important features from time and frequency domains from each recording. We standardized the features by removing the mean and scaling to unit variance. We then trained a model on a number of different values for kernels, regularization parameters (C) and kernel's coefficients (gamma).

In all three approaches, we extracted the labels from the audio signals names and we created 8 labels corresponding to approximately +5 vehicles each time. So our labels were ranging from 1-5 (label 0) to 35+ (label 7) vehicles detected.

v. Evaluation

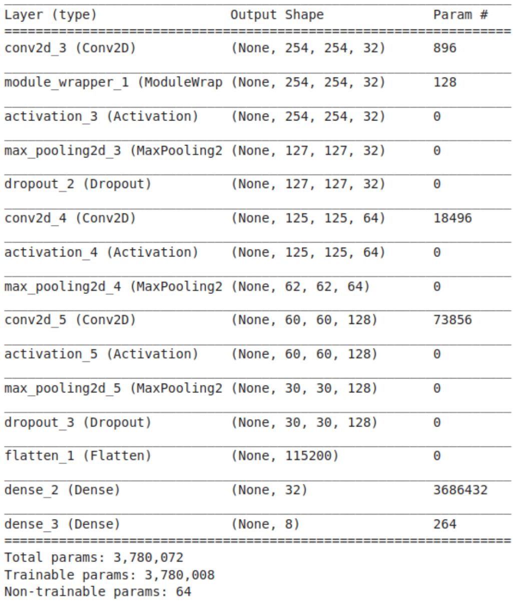
The evaluation of our Neural Network models' has been done with respect to performance in terms of training accuracy/loss considering the validation accuracy/loss.

The evaluation of our SVC algorithm has been done with respect to performance in terms of training/validation accuracy. The performance metrics that are used are accuracy and F1 score.

Finally, a live recording and prediction code structure was implemented, where we evaluate how well our algorithms could predict the label of a new, unknown audio signal.

III. Results

The following graphs visualize important metrics relevant to our three approaches. We present the model's structure for our two NN architectures and important metrics during the training and testing of our algorithms.



```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_3 (Conv2D)            (None, 254, 254, 32)      896

module_wrapper_1 (ModuleWrap (None, 254, 254, 32)      128

activation_3 (Activation)    (None, 254, 254, 32)      0

max_pooling2d_3 (MaxPooling2 (None, 127, 127, 32)      0

dropout_2 (Dropout)          (None, 127, 127, 32)      0

conv2d_4 (Conv2D)            (None, 125, 125, 64)      18496

activation_4 (Activation)    (None, 125, 125, 64)      0

max_pooling2d_4 (MaxPooling2 (None, 62, 62, 64)        0

conv2d_5 (Conv2D)            (None, 60, 60, 128)       73856

activation_5 (Activation)    (None, 60, 60, 128)       0

max_pooling2d_5 (MaxPooling2 (None, 30, 30, 128)       0

dropout_3 (Dropout)          (None, 30, 30, 128)       0

flatten_1 (Flatten)          (None, 115200)            0

dense_2 (Dense)              (None, 32)                3686432

dense_3 (Dense)              (None, 8)                 264
=================================================================
Total params: 3,780,072
Trainable params: 3,780,008
Non-trainable params: 64
```

Figure 7: *CNN Model Architecture*

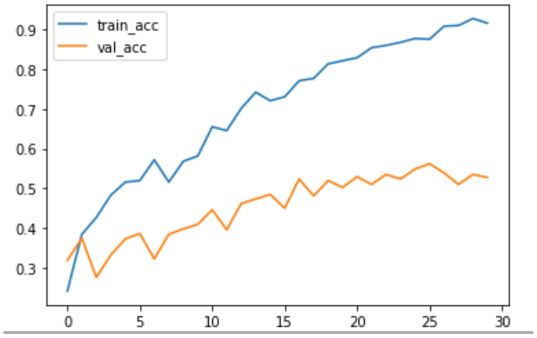


Figure 8: *CNN Training/Test Accuracy (30 epochs)*

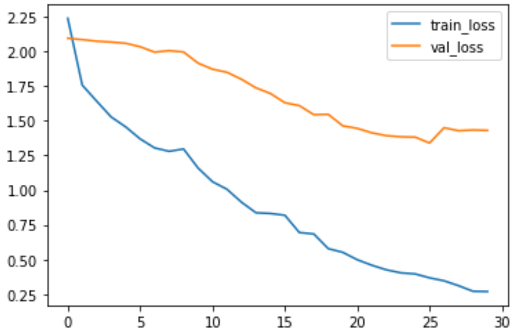
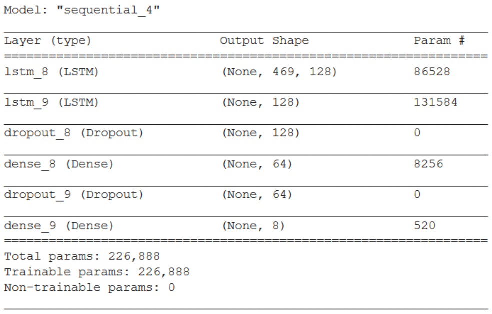


Figure 9: *CNN Training/Test Loss (30 epochs)*



```
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_8 (LSTM)                (None, 469, 128)          86528

lstm_9 (LSTM)                (None, 128)               131584

dropout_8 (Dropout)          (None, 128)               0

dense_8 (Dense)              (None, 64)                8256

dropout_9 (Dropout)          (None, 64)                0

dense_9 (Dense)              (None, 8)                 520
=================================================================
Total params: 226,888
Trainable params: 226,888
Non-trainable params: 0
_________________________________________________________________
```

Figure 10: *RNN-LSTM Model Architecture*

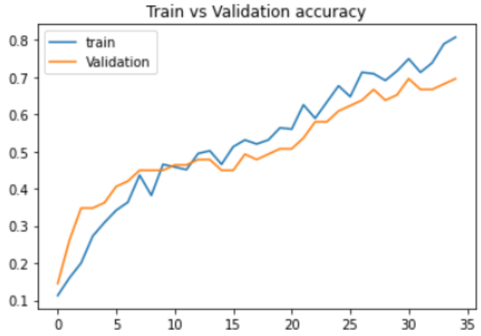


Figure 11: *RNN-LSTM Training/Test Accuracy (35 epochs)*

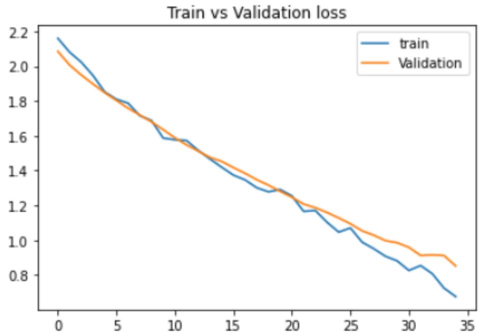


Figure 12: *RNN-LSTM Training/Test Loss (35 epochs)*

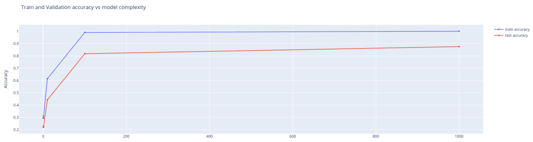
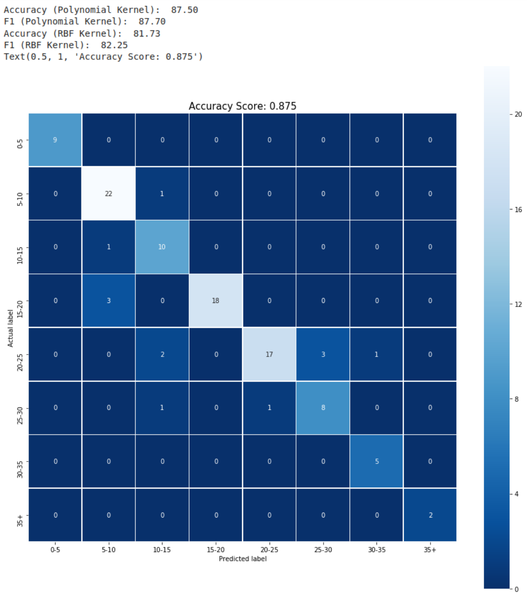


Figure 13: *SVC Model Complexity*



```
Accuracy (Polynomial Kernel):  87.50
F1 (Polynomial Kernel):  87.70
Accuracy (RBF Kernel):  81.73
F1 (RBF Kernel):  82.25
Text(0.5, 1, 'Accuracy Score: 0.875')
```

Actual label \ Predicted label	0-5	5-10	10-15	15-20	20-25	25-30	30-35	35+
0-5	9	0	0	0	0	0	0	0
5-10	0	22	1	0	0	0	0	0
10-15	0	1	10	0	0	0	0	0
15-20	0	3	0	18	0	0	0	0
20-25	0	0	2	0	17	3	1	0
25-30	0	0	1	0	1	8	0	0
30-35	0	0	0	0	0	0	5	0
35+	0	0	0	0	0	0	0	2

Figure 14: *SVC Confusion Matrix*

Below we present some of the predictions made by our SVC algorithm during the recording of new, unseen data:

Table 1: *Vehicle Number Predictions*

Passed	Label Predicted	Classified
11	2 (11-15)	Correct
42	7 (35+)	Correct
29	3 (15-20)	Wrong
9	1 (5-10)	Correct
10	1 (5-10)	Correct
4	0 (1-5)	Correct
33	4 (20-25)	Wrong
17	4 (20-25)	Wrong

IV. Discussion

The results suggest that our two NN models don't perform as well as our SVC algorithm. The RNN-LSTM model seems to perform much better than the CNN during training, but still faces problems during the prediction stage. Our SVC algorithm has the best prediction results, with a score of 5/8 (≈0.65) correct classified examples (Table 1). This lead us to the conclusion that the correct prediction is way harder in samples with lots of vehicles passing by. It also makes clear that small and noisy datasets are better to be approached via the traditional machine learning techniques and algorithms.

In this work we implemented three different algorithms in order to study how well these algorithms can predict, given an audio signal, the number of vehicles that are heard in this signal. The study can be extended in a number of different ways. First of all, it can be implemented using a Regression approach. It can also be extended from vehicle detection to vehicle detection and classification too. Last but not least, a combination of both acoustic and image/video data could yield far more better prediction results.

References

[1] Dalir, Ali, Ali Asghar Beheshti, and Morteza Hoseini Masoom. "Classification of vehicles based on audio signals using quadratic discriminant analysis and high energy feature vectors." arXiv preprint arXiv:1804.01212 (2018).

[2] George, Jobin, et al. "Exploring sound signature for vehicle detection and classification using ANN." International Journal on Soft Computing 4.2 (2013): 29.

[3] Wieczorkowska, Alicja, et al. "Spectral features for audio based vehicle and engine classification." Journal of Intelligent Information Systems 50.2 (2018): 265-290.

[4] Johnstone, Michael N., and Andrew Woodward. "Automated detection of vehicles with machine learning." (2013).

[5] Chellappa, Rama, Gang Qian, and Qinfen Zheng. "Vehicle detection and tracking using acoustic and video sensors." 2004 IEEE International Conference on Acoustics, Speech, and Signal Processing. Vol. 3. IEEE, 2004.