City University London

Msc in Data Science

Project Report

2018

Multi-Resolution CNNs for Urban Sound Classification

Author: Joy Basford

Supervised By: Tillman Weyde

1st October 2018

Declaration

By submitting this work, I declare that this work is entirely my own except those parts duly identified and referenced in my submission. It complies with any specified word limits and the requirements and regulations detailed in the assessment instructions and any other relevant programme and module documentation. In submitting this work, I acknowledge that I have read and understood the regulations and code regarding academic misconduct, including that relating to plagiarism, as specified in the Programme Handbook. I also acknowledge that this work will be subject to a variety of checks for academic misconduct.
Signed:

Joy Basford

Abstract

This Msc Data Science Project investigated the use of Multi-Resolution Convolutional Neural Networks for urban sound classification. Informed by findings in adjacent literature, variations in kernel shape, size, size range and concatenation choice were examined for their effect on classification performance in a single convolutional layer neural network. Two neural network architectures were developed to this effect, the Multi-Resolution Single-Layer CNN and the Multi-Resolution Parallel Pipeline CNN. Variants of the former focused on evaluating the effect of kernel shape whereas variants of the latter focused on the effect of size, size range and how the pipelines were subsequently concatenated. Findings show that short and medium kernels were better at learning features from the UrbanSound8K dataset and that kernel shapes and concatenation choice made negligible impact on the architectures tested.

Keywords: Urban Sound Classification, Multi-Resolution Convolutional Neural Network, Audio Signal Processing, Convolutional Neural Network

Table of Contents

Page

Abstract 2

1. Introduction and Objectives 5
   1.1. Formatting 5
   1.2. Background, Purpose and Beneficiaries 5
   1.3. Objectives 6
   1.4. Methods 6
   1.5. Work Plan 6
   1.6. Changes 8
   1.7. Report Structure 8

2. Context 9
   2.1. Urban Sound Classification 9
   2.2. Data Augmentation and State-of-the-art Methods 10
   2.3. Multi-Resolution CNNs 10
   2.4. Parallel Pipelines 12
   2.5. Context Summary 13

3. Methods 14
   3.1. Establish Baseline Dataset 14
   3.2. Establish Baseline Model 14
   3.3. Multi-Resolution Single-Layer CNNs (MRSL) 16
   3.4. Multi-Resolution Parallel Pipeline CNNs (Single Category) 17
   3.5. Multi-Resolution Parallel Pipeline CNNs (Combined Categories) 18
   3.6. Model Depth 19
   3.7. Data Augmentation 20
   3.8. Training, Validation and Testing 20

4. Results 22
   4.1. Baseline Results 22
   4.2. Results of MRSL CNNs 23
   4.3. Results of MRPP CNNs (Single Category) 25
   4.4. Results of MRPP CNNs (Combined Categories) 27
   4.5. Model Depth 30
   4.6. Data Augmentation 30
   4.7. Analysis of Results 31

5. Discussion 33
   5.1. Evaluation of Results and Objectives 33
      5.1.1 Establish a Baseline, Modelled on (Piczak, 2015) 33
      5.1.2 Design and Develop Multi-Resolution CNNs 33
      5.1.3. Train Models and Evaluate Their Performance 34
   5.2. New Findings 34
   5.3. Implications of Findings 34
   5.4. Response to Research Question 35

6. Evaluation, Reflections, and Conclusions 36
   6.1. Choice of Objectives 36
   6.2. What Could Have Been Improved? 36
   6.3. What was Learned from the Project? 36
   6.4. Proposals for Further Work 36

7. References 37

Appendix A – Submitted Project Plan 40
Appendix B – Project Risk Register 48
Appendix C – Project Work Schedule 49
Appendix D – Updated Work Schedule Reflecting Delays 50
Appendix E – Parallel Pipelines in (Bae, Choi & Kim, 2016) 51

1 Introduction and Objectives

1.1 Formatting

Capitalization is used to identify the Author (Joy Basford), the Project (Multi-Resolution CNNs for Urban Sound Classification), and the Report (this report).
Software names are shown in *black italic*.
Quotes are shown in blue italic.

1.2 Background, Purpose and Beneficiaries

Next to sight, hearing is one of the primary ways that humans perceive characteristics of a location. While humans can identify individual sounds easily, computers require specialized programs to learn the salient features of audio signals before being able to carry this out. In recent years, the adoption of deep learning systems to this effect has increased.

While it was asserted that 'the power of shallow architectures is fundamentally limited' (Humphrey, Bello & LeCun, 2013), (Piczak, 2015) demonstrated that a relatively shallow Convolutional Neural Network (CNN) with two convolutional layers and two dense layers was able to obtain results comparable with state-of-the-art systems that relied on manually-extracted features. In presenting his model, Piczak's goal was not to introduce a state-of-the-art system but a baseline for CNNs applied to urban sound classification.

Current state-of-the-art systems (Salamon and Bello, 2016) and (Tokozume, Ushiku & Harada, 2018) emphasised the need for data augmentation, however, promising outcomes in adjacent fields of research also suggested that optimizing the kernel dimensions and modelling features in separate pipelines could have a positive effect on learning in urban sound classification. If the Project were to validate these findings, they could potentially be used to complement and improve state-of-the-art methods. To this purpose, the following research question was proposed:

What is the effect of Multi-Resolution CNNs on urban sound classification?

In response to the research question, two CNN architectures were created for investigation, heavily influenced by (Pons, Lidy and Serra, 2016) and (Pons and Serra, 2017). To the best of the Author's knowledge, Multi-Resolution CNN architectures have not yet been adapted for the task of urban sound classification.

The beneficiaries of this work would be anyone who would benefit from greater insight into urban sound classification, including:

- Researchers in audio signal processing either focused on urban sound classification or on related challenges including acoustic scene detection and domestic audio tagging would find new effective methods useful.
- Researchers requiring 'a data-driven understanding of urban sound and noise patterns' (Salamon and Bello, 2015). For example, in urban informatics research Aiello et al. (2016) use social media data to construct urban sound maps.
- Research groups funded by legislative bodies concerned with urban noise. For example, the European Commission funded the SILENCE research project to develop a system of methods and technologies to address this.

1.3 Objectives

The following objectives were outlined to satisfy the research question:

- Establish a suitable baseline, informed by findings in relevant literature.
- Measure the effect of various kernel shapes, sizes and combinations thereof on classification accuracy.
- Determine any compelling findings that could be used to compliment other methods in urban sound classification.

1.4 Fulfilment of Objectives

The following tasks were identified to satisfy these objectives:

- Establish a baseline CNN, modelled on (Piczak, 2015).
- Design and develop architectures to investigate the effect of kernel shapes, sizes and different combinations on performance.
- Train and evaluate the performance of these architectures against the baseline.

1.5 Methods

To carry out the tasks, the following methods were initially outlined:

- Establish a baseline CNN, modelled on (Piczak, 2015)
  Replicate fully the best performing variant of the model demonstrated in (Piczak, 2015). If this is unsuccessful, establish a baseline as closely as possible to (Piczak, 2015). Any departures should be justifiable, for example the priority of one python framework over the original another should be motivated by clarity and ease of replication for further work. Identify relevant implementations and test smaller sections of code separately and on sub-samples of data to understand how each function and class operates. In the different implementations, invest time to explore the use of different methods to achieve the same goal and consider the most efficient, robust and interpretable approach.
- Design and develop CNN architectures to investigate the effect of applying different kernel shapes, sizes and combinations on classification accuracy in urban sound classification. To this end, Multi-Resolution CNNs will be developed. In exploring different implementations and scripts, consider appropriate options for building these architectures in the chosen deep learning framework. For example, different combinations could be modelled and interpreted more easily with the use of separate pipelines.
- Train CNNs and evaluate their performance against the baseline.
  Use the same implementation script for training, validation and testing as the baseline, if possible from the original code used in (Piczak, 2015).

1.6 Work Plan

After the Project Plan was submitted, it was changed on the view that access to the same training data used for the original baseline would be unlikely. To reflect the change of Project and provide guidance throughout its execution a new Work Plan (Figure 1), Risk Register and Work Schedule set up accordingly (see Appendices B and C).

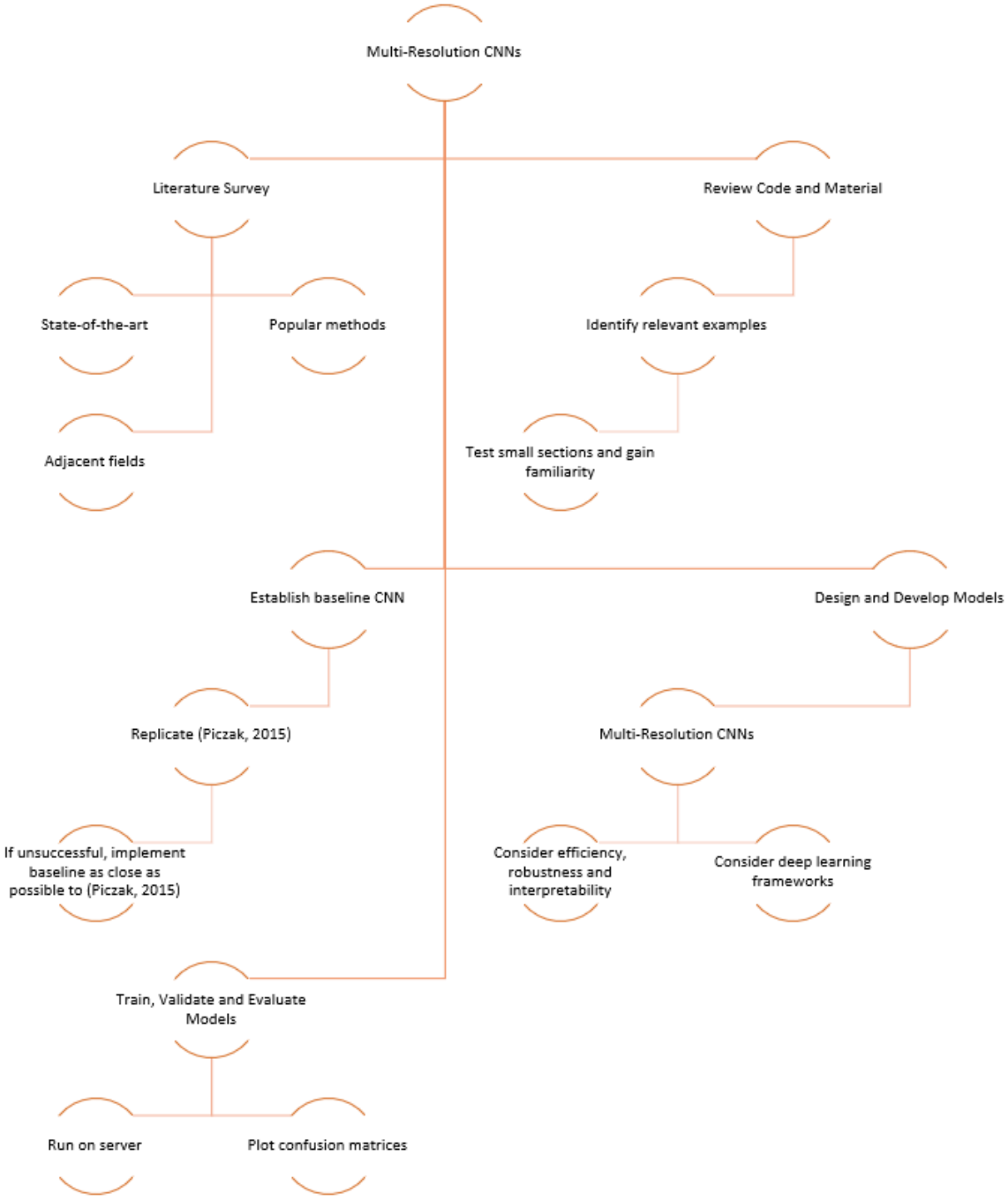


Figure 1. Work Plan

1.7 Changes

Due to complications with training data access, the Project underwent changes after the original Project Plan was submitted, resulting in a new task and corresponding objectives, using a new baseline and dataset. These are outlined in Table 1.

	Original Project Plan	Current Project
Task	Singing Voice Source Separation	Urban Sound Classification
Objectives	Investigate the effect of Capsule Networks on Singing Source Separation	Investigate the effect of Multi-Resolution CNNs on Urban Sound Classification
Training Dataset	Music owned by Spotify	UrbanSound8K
Baseline Model	U-Net (Jansson et al., 2017)	CNN (Piczak, 2015)

Table 1. Changes to Project since submission of Project Plan

1.8 Report Structure

The remaining sections of the Report are structured as below:

Chapter 2: Context
This chapter offers the contextual frame of reference, including commentary on state-of-the-art methods and research that influenced the design of the models and experiments.

Chapter 3: Methods
In this chapter, the methods used to answer the research question are presented, including the approaches taken to establish a baseline model, the design and development of the models and the setup of the experiment.

Chapter 4: Results
Here, the results of the experiments are presented and analysed in relation to the baseline, related architectural variants and with regards to class-specific performance.

Chapter 5: Discussion
In this chapter, the results are reviewed in comparison with the objectives and in relation to relevant literature.

Chapter 6: Evaluation, Reflection and Conclusions
This chapter evaluates what was achieved in carrying out the Project and draws conclusions.

2 Context

2.1 Urban Sound Classification

Audio classification methods have made significant use of deep learning methods in recent years. A useful baseline was established in (Piczak, 2015) demonstrating the potential for CNNs to achieve performances comparable to those by systems using manually engineered features. Piczak extracted log-scaled mel-spectrograms of the data as inputs to his model. His best performing variant used input dimensions of (60 bands, 101 frames) combined with first-order differences as 2-channel inputs into a convolutional neural network (Figure 2), trained for 150 epochs. Piczak's model outperformed a state-of-the-art system using manually-extracted features, achieving 73.1% accuracy on the long-segment inputs of the UrbanSound8K dataset. In terms of class-wise performance, the baseline model performed better at recognizing the following classes: air conditioner, car horn, children playing and dog bark. Classes that presented difficulties for the model were those with short-scale temporal structure: drilling, engine idling and jackhammer. Confusion matrices were used to identify class-wise performance (Figure 3) with the following abbreviations for class labels (Table 2).

AI	CA	CH	DO	DR	EN	GU	JA	SI	ST
Air Conditioner	Car Horn	Children Playing	Dog Bark	Drilling	Engine Idling	Gun Shot	Jackhammer	Siren	Street Music

Table 2. Abbreviations for UrbanSound8K class labels

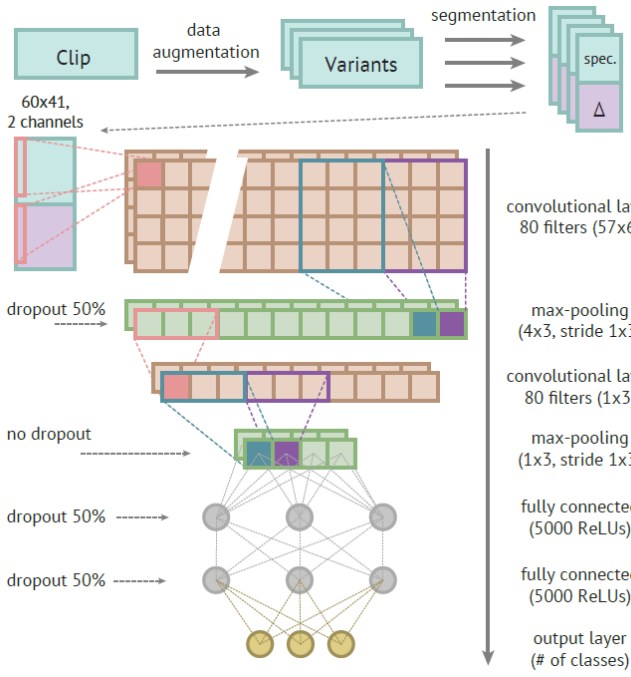


Figure 2. Feature Extraction and CNN architecture in (Piczak, 2015).

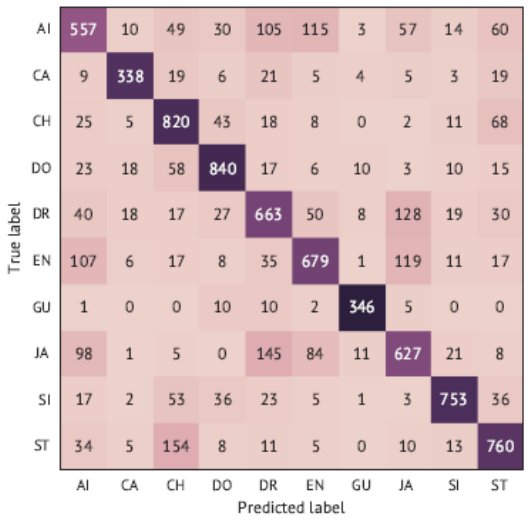


Figure 3. Confusion matrix showing abbreviations for classes 0-9.

2.2 Data Augmentation and State-Of-The-Art Methods

Although Piczak (2015) found negligible benefit in data augmentation through the use of pitch-shifting and time-stretching with the *librosa* library, other augmentation techniques used on the UrbanSound8K dataset, including (Salamon and Bello, 2016) and (Tokozume, Ushiku & Harada, 2018), have been shown to contribute significantly to state-of-the-art performance. In (Salamon and Bello, 2016), a range of data augmentation techniques were assessed, and state-of-the-art performance was obtained using augmented data combined with deep architectures. Additionally, they observe the impact of each augmentation method on class-wise performance and note that this is different for each class, suggesting that class-specific data augmentation could further improve results. Data augmentation raises accuracy by increasing the number of training samples available. This also enables smaller datasets to be applicable for the use of deep learning methods. By comparison, Tokozume, Ushiku and Harada (2018) adopted a novel approach resembling a form of data pre-processing but also considered as augmentation. The objective was to learn the mixing ratio of two separate classes mixed together as inputs. This proved highly effective when applied to class prediction and achieves comparable results to (Salamon and Bello, 2016).

These state-of-the-art approaches make a strong case for the UrbanSound8K dataset to be augmented and the use of deep architectures is widely observed in surrounding literature (Choi, Fazekas & Sandler, 2016), (Gencoglu, Virtanen & Huttunen, 2014), (Kons and Toledo-Ronen, 2014). However, if there remains a possibility that other methods might re-frame this task or complement existing approaches, there is an opportunity to advance current state-of-the-art performance. Motivated to identify such methods, the following sections review recent developments in adjacent fields, with potential application to urban sound classification.

2.3 Multi-Resolution CNNs

In the adjacent field of music classification, Pons, Lidy and Serra (2016) demonstrate the possibility of producing superior results after reshaping convolutional kernels to adapt to the desired input features. They present new architectures, used to assess the extent to which kernel shape could affect the representational power of the convolutional layer for classifying music genre using the Ballroom dataset (Gouyon, 2004). Their experiments compare the performance of a ‘Frequency’ CNN with a convolutional kernel of 1-by-n dimensions mapping representations along the frequency axis, a ‘Time’ CNN with a kernel of m-by-1 dimensions to capture information along the time axis and a ‘Black-box’ CNN with an m-by-n shaped kernel based on (Lidy, 2015). Figures 4 and 5 show schemas of these architectures.

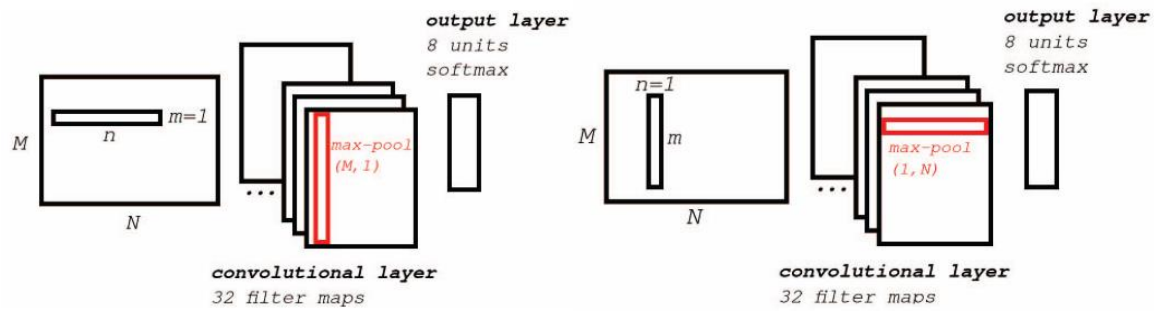


Figure 4. Schemas of the Frequency and Time architectures (Pons, Lidy & Serra., 2016).

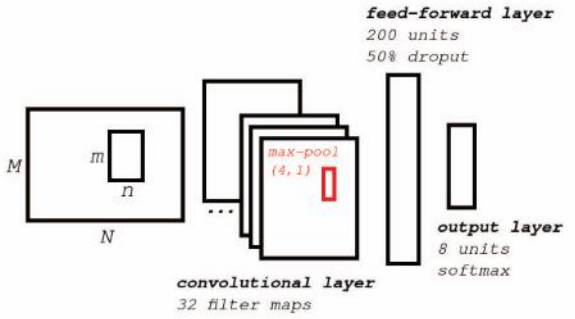


Figure 5. Schema of the Black-box architecture (Pons, Lidy & Serra., 2016)

The models are then trained and evaluated on mel-spectrograms. As the Ballroom dataset is characterised by its tempo, it is understandable that the Time architecture produced results comparable to the baseline (Gouyon, 2004). However, results also indicated that frequency features in this dataset were relevant (Pons, Lidy & Serra, 2016) (Figure 6). Furthermore, the authors found that combining separate temporal and frequency pipelines achieved similar results to the Black-box architecture but had the advantage of greater interpretability.

Architecture	Input (M,N)	Filter shape (m,n)	# param.	Max-pool	Accuracy: mean ± std 10 cross-fold validation	Baseline
Black-box	(40,80)	(12,8)	3.275.312	(4,1)	**87.25** ± 3.39 %	93.12 % → Marchand *et al.* [11]
Black-box	(40,250)	(12,200)	2.363.440	(4,1)	82.80 ± 5.12 %	93.12 % → Marchand *et al.* [11]
Time	(40,80)	(1,60)	7.336	(40,1)	**81.79** ± 4.72 %	82.3% → Guyon *et al.* [6]
Time	(40,250)	(1,200)	19.496	(40,1)	81.52 ± 3.87 %	82.3% → Guyon *et al.* [6]
Frequency	(40,80)	(30,1)	3.816	(1,80)	59.45 ± 5.02%	15.9 % → Most probable class
Frequency	(40,80)	(32,1)	3.368	(1,80)	**59.59** ± 5.82 %	15.9 % → Most probable class
Frequency	(40,80)	(34,1)	2.920	(1,80)	58.17 ± 3.58 %	15.9 % → Most probable class
Frequency	(40,80)	(36,1)	2.472	(1,80)	57.88 ± 5.38 %	15.9 % → Most probable class
Frequency	(40,80)	(38,1)	2.024	(1,80)	57.45 ± 5.93 %	15.9 % → Most probable class
Frequency	(40,80)	(40,1)	1.576	(1,80)	52.43 ± 5.63 %	15.9 % → Most probable class
Time-Frequency	(40,80)	(1,60)-(32,1)	196.816	(40,1)-(1,80)	86.54 ± 4.29 %	93.12 % → Marchand *et al.* [11]
Time-FrequencyInit	(40,80)	(1,60)-(32,1)	196.816	(40,1)-(1,80)	**87.68** ± 4.44 %	93.12 % → Marchand *et al.* [11]

Figure 6. Experiment results (Pons, Lidy & Serra, 2016)

In a similar investigation, Pons and Serra (2017) assert that the small, default, m-by-n convolutional filters limit the representational capacity of the first convolutional layer by being restricted to map only short time and frequency contexts. The consequence is that larger contexts require deeper architectures. The authors argue that larger contexts could be represented in the first convolutional layer, freeing up subsequent layers to model other features and increasing overall network efficiency. An example of where this could be applied would be a hybrid architecture comprised of a CNN for modelling short temporal features and an RNN for modelling long temporal features, where a more efficient CNN would enable the RNN to model larger contexts (Pons and Serra, 2017).

Two models (O-net and P-net) were designed to model short time-scale features (onsets) and longer time-scale features (patterns), selecting kernel sizes according to dataset-specific heuristics. As with (Pons, Lidy & Serra, 2016) these experiments were carried out on the Ballroom dataset (Gouyon, 2004). Their results indicate that while P-nets generally outperform O-nets, the highest accuracy was obtained by combining O-net and P-net architectures together, only 3.75% away from the state-of-the-art performance (96%) that used manually-engineered features (Figure 7).

Model:	hop	# params	accuracy	Model:	hop	# params	accuracy
O-net	250/80	4,188	76.66/85.24 %	*4x O-net + 4x P-net*	250/80	46,408	88.82/91.55 %
P-net	250/80	7,428	83.95/89.26 %	*8x O-net + 8x P-net*	250/80	92,808	88.68/**92.27** %
2x O-net	250/80	8,368	81.53/86.54 %	*Marchand et al.* [10]	-	-	**96** %
O-net + P-net	250/80	11,608	87.25/89.68 %	*Time* [4]	80	7,336	81.79 %
2x P-net	250/80	14,848	85.67/89.11 %	*Time-freq* [4]	80	196,816	87.68 %
2x O-net + 2x P-net	250/80	23,208	87.25/91.27 %	*Black-box* [4]	80	3,275,312	87.25 %

Figure 7. Results from (Pons and Serra, 2017). 92.27% was achieved using the 8 x O-net + 8 x P-net model, 3.75% away from the baseline.

Both papers had a strong influence on the design of the Multi-Resolution CNNs in the Project. The central design of this architecture employs multiple kernels concatenated in a single convolutional layer and provides the ability to test for variations in kernel shape, size and different combinations thereof.

2.4 Parallel Pipelines

Another musically motivated architecture was presented in (Lidy and Schindler, 2017) where parallel convolutional neural networks learn temporal and timbral information separately before being subsequently merged.

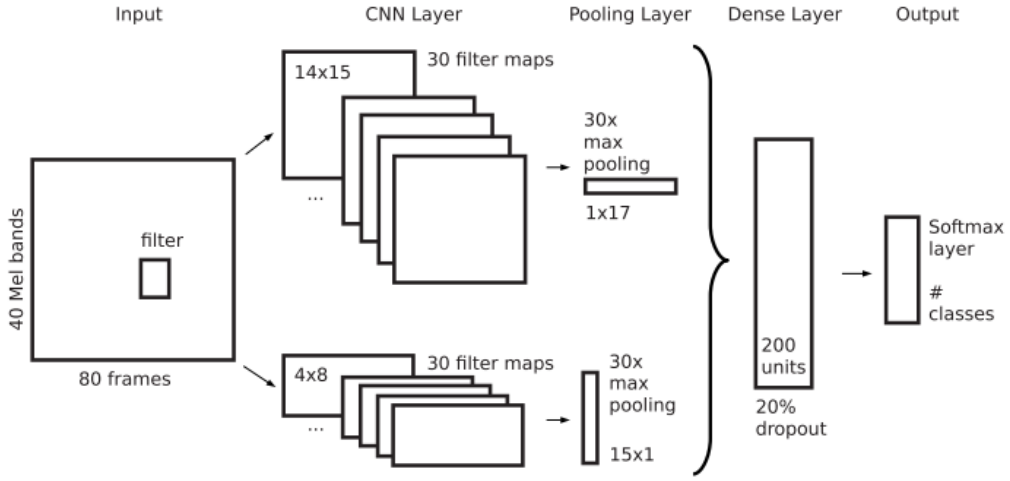


Figure 8. One of the parallel CNNs presented in (Lidy and Schindler, 2017)

Lidy and Schindler designate the upper pipeline to model frequency features and the lower pipeline to temporal features. This architecture is based on an earlier publication for their winning submission in the music/speech classification challenge for the eleventh running of the Music Information Retrieval Evaluation eXchange (MIREX 2015). As with the use of re-shaped kernels, parallel pipelines offer the advantage for potentially greater interpretability of the network learning process than using a single pipeline, since the modelling of each desired feature can be controlled separately.

A different example of parallel pipeline architecture was observed in (Bae, Choi & Kim, 2016), comprised of a combination of a LSTM and CNN, in their submission for the Acoustic Scene Classification (ASC) task in the Detection and Classification of Acoustic Scenes and Events workshop (DCASE16). Since this is a similar task to urban sound

classification, there is potential for methods applied to one to be adapted for the other. Bae, Choi and Kim model different input features in their system, using the LSTM to model sequences and the CNN to model spectrogram images (see Appendix E). Their system outperformed the accuracies of conventional DNN, CNN and LSTM architectures since the combined structure learned 'complimentary information'. These findings draw attention to the benefit of modelling features separately when there is a relationship between the desired features and demonstrate that different neural network architectures can be combined to this end to make full use of their respective learning processes.

2.5 Context Summary

- Effective feature representation can be obtained by re-shaping kernels (Pons, Lidy & Serra, 2016), (Pons and Serra, 2017) and corresponding pool shapes (Lidy and Schindler, 2017) in accordance with known characteristics of the input data.
- The representational capacity of the first convolutional layer can be maximized by combining varying shapes and sizes of convolutional kernel (observed in O-net and P-net combinations), freeing up additional layers or neural network architectures to be used for modelling higher-level features (Pons and Serra, 2017).
- Combining complementary information can enable neural network architectures to obtain higher classification accuracies, (Lidy and Schindler, 2017), (Bae, Choi & Kim., 2016).

3 Methods

3.1 Establish Baseline Dataset

As the largest dataset used in (Piczak, 2015), the dataset selected for the Project was the UrbanSound8K dataset (Salamon, Jacoby and Bello, 2015). Datasets providing suitable back-up options were also identified in (Hettiola, n.d). UrbanSound8K contains 8,732 audio samples of 10 classes. Some samples have a duration of 1-2 seconds, but the majority are 4 second clips of longer original samples from an online sound repository. To avoid overfitting caused by segments from the same original sample being used for validation and testing, Salamon, Jacoby and Bello arranged the data carefully into 10 folders (for 10-Fold cross validation) to the effect that each folder is roughly balanced (2015). A metadata file contains the audio file names in the format: [freesoundID]-[classID]-[occurrenceID]-[sliceID].wav where the classID is a zero-indexed, numeric identifier for the class. Classes are shown in Table 3. Using *librosa*, the Author generated and displayed log-scaled mel-spectrograms from each class for initial visual inspection (Figure 9).

Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7	Class 8	Class 9
Air Conditioner	Car Horn	Children Playing	Dog Bark	Drilling	Engine Idling	Gun Shot	Jackhammer	Siren	Street Music

Table 3. UrbanSound8K Class Labels.

3.2 Establish Baseline Model

As it was used both in (Salamon and Bello, 2016) and (Tokozume, Ushiku & Harada, 2018), Piczak's CNN was the preferred model for the Project baseline. Several alternative implementations had also been identified and tested in an earlier review period of materials. These were kept on hand to facilitate the understanding of Piczak's scripts and provide back-up options should they be necessary. Given that the source code was available, a fortnight was considered sufficient for the Author to pre-process the dataset, replicate the same inputs and model that Piczak used to achieve 73.1% accuracy. However, the technical challenges were greater than expected. One issue with installing and running *ctypes*, a *Python* library for using dynamic link libraries (DLLs), caused significant delay to the Project, as time was invested into understanding the dependencies and alternatives were sought. After an additional 4 days over the original time allocated, it became clear that alternative implementations needed to be leveraged.

From the relevant implementations identified, Zhu, Kaznady and Hendry (2017) provided the most appropriate option, since they preserved the organization of the data in their original folders and utilized 10-Fold cross-validation as recommended by Salamon, Jacoby & Bello (2014). Moreover, their use of the *Keras* framework for constructing and training the neural network model is preferable to *Pylearn2* due to widespread use of *Keras* in the deep learning community and the availability of clear, detailed documentation and examples. Their implementation diverged from Piczak's methods, however, so it became imperative to try to restore these. Three main differences were identified in the implementation by Zhu, Kaznady and Hendry (2017):

- New input dimensions (150 frames and 150 bands).
- The use of second-order features, resulting in 3-channel input.
- The use of a different CNN architecture.

Zhu, Kaznady and Hendry reference the fact that longer input variants in (Piczak, 2015) achieved a higher accuracy and therefore used log-scaled mel-spectrograms of 150

frames and 150 bands, combined with their first and second-order features to create a 3-channel input to the model. The Author adapted their implementation to restore the inputs to their original dimensions (101 frames and 60 bands), combined the features to their deltas as 2-channel inputs and replaced their deep CNN with Piczak's CNN model successfully translated from *Pylearn2* into *Keras*. Changing the implementation was expected to result in only minor discrepancies. However, a test at this stage revealed a difference of over **16%** between the mean accuracy of the new script and the accuracy achieved by Piczak. Over the following days, the script was inspected with each section were tested separately in efforts to find the cause. Cognisant that time spent on this task was becoming disproportionate to the remaining requirements of the Project, it was decided to keep the input shape as 101 frames and 60 bands as per (Piczak, 2015) and use 3 channels as per (Zhu, Kaznady & Hendry, 2017), which narrowed the discrepancy slightly to **12.71%**.

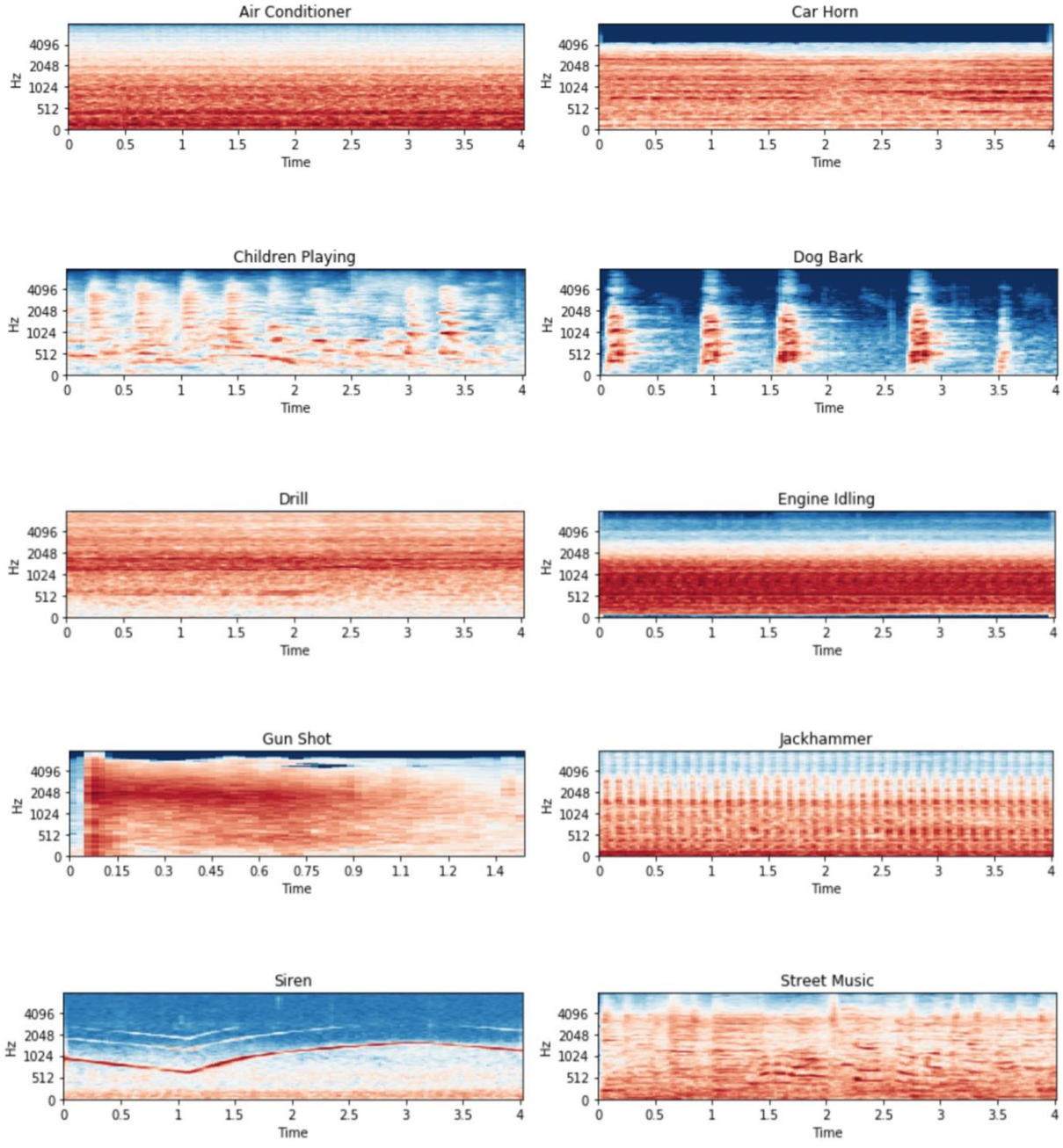


Figure 9. The Author used *librosa* to generate and display log-scaled mel spectrograms of each class.

Components of the final baseline are outlined below.

Feature Extraction:

- Audio samples were loaded and either padded or truncated to equally sized segments of 4 second duration before being converted into log-scaled mel-spectrograms, using 60 mel-bands.
- First and second-order features were extracted and combined with each 2D sample to create a 3-channel input to the neural network model with dimensions (101, 60, 3).
- Labels were extracted from the classID in the file names of each sample and one-hot encoded to be used with categorical cross-entropy loss.

Model construction, training, validation and testing:

- A replication of Piczak's CNN was translated to *Keras*, compiled and fitted to the input data.
- The models were trained on mini-batches (batch-size 1000) with stochastic gradient descent using 0.9 Nesterov momentum. This was performed up to a maximum of 150 epochs, with early-stopping if no improvement was made after any successive 20 epochs.
- Piczak's use of regularisation techniques (l2 and dropout) were preserved.
- Following a 10-Fold cross-validation regime, the model was validated and tested.
- The mean accuracy and standard deviation were recorded after each fold.

3.3 Multi-Resolution Single-Layer CNNs (MRSL)

Inspired by (Pons, Lidy & Serra, 2016) and (Pons and Serra, 2017), this architecture consisted of a single convolutional layer in which kernels of a range of sizes (designed to accommodate desired features) are concatenated and followed by a max-pooling layer that reinforces this preference. The desired features were measured in frequency and temporal patterns. In the first experiment, the architecture was configured to prefer frequency features, prioritizing information captured in the y-axis, and in the second, it was configured to prefer temporal features, prioritizing information captured in the x-axis.

Construction was carried out in *Keras*, by specifying separate filter and pool dimensions before flattening and concatenating them. Working off the heuristic that the filter sizes should vary in equally sized increments, the size range spanned from 4 to 60, with an increment of 4. The 2D filter shapes are expressed as a tuple, with one side remaining fixed at 4 while the other increases incrementally to 60. Pools were also configured to prefer information along the chosen axis, with the side corresponding to the desired axis set at 1 and the other set to 2. Kernel and pool sizes are shown in Tables 4 and 5.

After the single convolutional and pooling layer, the information learned was flattened and passed to a single dense hidden layer with 5000 ReLU units before being passed to an output layer using Softmax activation (Figure 10). Experiments are outlined below:

- Experiment 1: Multi-Resolution Single-Layer (MRSL) CNN prioritizing frequency
- Experiment 2: Multi-Resolution Single-Layer (MRSL) CNN prioritizing temporal

Experiment 1 - MRSL CNN (Frequency)															
Kernel	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15
Size	**4,4**	**8,4**	**12,4**	**16,4**	**20,4**	**24,4**	**28,4**	**32,4**	**36,4**	**40,4**	**44,4**	**48,4**	**52,4**	**56,4**	**60,4**
Pool	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
Size	**1,2**	**1,2**	**1,2**	**1,2**	**1,2**	**1,2**	**1,2**	**1,2**	**1,2**	**1,2**	**1,2**	**1,2**	**1,2**	**1,2**	**1,2**

Table 4 Multi-Resolution Single-Layer (MRSL) CNN prioritizing Frequency.
C1-C15 denote the convolutional kernels and P1-P15 denote the max-pools.

Experiment 2 - MRSL CNN (Time)															
Kernel	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15
Size	**4,4**	**4,8**	**4,12**	**4,16**	**4,20**	**4,24**	**4,28**	**4,32**	**4,36**	**4,40**	**4,44**	**4,48**	**4,52**	**4,56**	**4,60**
Pool	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
Size	**2,1**	**2,1**	**2,1**	**2,1**	**2,1**	**2,1**	**2,1**	**2,1**	**2,1**	**2,1**	**2,1**	**2,1**	**2,1**	**2,1**	**2,1**

Table 5 Multi-Resolution Single-Layer (MRSL) CNN prioritizing Time
C1-C15 denote the convolutional kernels and P1-P15 denote the max-pools.

Expectations for the performance of this architecture were limited because in its preference for one feature, it must neglect the other. The objective was to measure this effect and to establish the extent to which the model could still distinguish between class when prioritizing learning on only one of the 2D input axes. This was useful to compare against the baseline, which strongly favoured the frequency axis in the first convolutional layer kernels. Figure 10 offers a schema of this model.

3.4 Multi-Resolution Parallel Pipeline CNNs (Single Category)

In this architecture, kernels were grouped by size into the following categories: short, medium or long. In a separate experiment for each of these groups, both frequency and temporal features were learned separately in parallel pipelines before being flattened and concatenated according to feature type. For each axis, the relevant side of the kernel shape ranged between 4 and 20 for the short group, 24-40 for the medium group and 44-60 for the long group. After a single convolutional and pooling layer, the information learned was flattened and passed to a single dense hidden layer with 5000 ReLU activation units and subsequently to a dense output layer using Softmax activation.

The purpose of this design was to assess the impact of restricting kernels by size on overall accuracy, to validate Pons and Serra's assertion that this potentially limits the network learning process (2017). Moreover, in class-specific performance, it was anticipated that the short variant of this architecture would achieve better results than the baseline in the Drilling, Engine and Jackhammer classes, which possess shorter time-scale features. The experiments are shown below with their abbreviations and Tables 6 to 8 display the sizes of kernels and pools used in each.

- Experiment 3: Short kernels for frequency and time (SF + ST)
- Experiment 4: Medium kernels for frequency or time (MF + MT)
- Experiment 5: Long kernels for frequency or time (LF + LT)

Experiment 3	5 Short Kernels					5 Short Kernels				
Kernel	C1	C2	C3	C4	C5	C1	C2	C3	C4	C5
Size	**4,4**	**8,4**	**12,4**	**16,4**	**20,4**	**4,4**	**4,8**	**4,12**	**4,16**	**4,20**
Pool	P1	P2	P3	P4	P5	P1	P2	P3	P4	P5
Size	**1,2**	**1,2**	**1,2**	**1,2**	**1,2**	**2,1**	**2,1**	**2,1**	**2,1**	**2,1**
Merge by	Frequency					Time				

Table 6 Multi-Resolution Parallel Pipeline (MRPP) CNN with short kernels.

Experiment 4	5 Medium Frequency Kernels					5 Medium Temporal Kernels				
Kernel	C1	C2	C3	C4	C5	C1	C2	C3	C4	C5
Size	**24,4**	**28,4**	**32,4**	**36,4**	**40,4**	**4,24**	**4,28**	**4,32**	**4,36**	**4,40**
Pool	P1	P2	P3	P4	P5	P1	P2	P3	P4	P5
Size	**1,2**	**1,2**	**1,2**	**1,2**	**1,2**	**2,1**	**2,1**	**2,1**	**2,1**	**2,1**
Merge by	Frequency					Time				

Table 7 Multi-Resolution Parallel Pipeline (MRPP) CNN with medium kernels.

Experiment 5	5 Long Frequency Kernels					5 Long Temporal Kernels				
Kernel	C1	C2	C3	C4	C5	C1	C2	C3	C4	C5
Size	44,4	48,4	52,4	56,4	60,4	4,44	4,48	4,52	4,56	4,60
Pool	P1	P2	P3	P4	P5	P1	P2	P3	P4	P5
Size	1,2	1,2	1,2	1,2	1,2	2,1	2,1	2,1	2,1	2,1
Merge by	Frequency					Time				

Table 8 Multi-Resolution Parallel Pipeline (MRPP) CNN with short kernels.

Although both feature types were equally prioritized, the expectations for the performance of Multi-Resolution Parallel Pipeline (Single Category) CNNs were still limited because they all excluded two of the three size groups, restricting the amount of information that could be potentially mapped by the kernels. The objective was to see whether the classification accuracy was higher when the CNN prioritized both axes equally while restricted to a single kernel size group compared to when prioritizing only one axis but with access to a wider group of kernel sizes.

3.5 Multi-Resolution Parallel Pipeline CNNs (Combined Categories)

As a variant of the previous architecture, more than one kernel size group was concatenated, presenting the option to concatenate the pipelines by feature type or by kernel size group. The purpose of this design was to measure the effect of extending the range of kernel sizes and merging the information captured according to feature type or kernel size group, to determine if there was an impact on the network's learning process.

After the convolutional and pooling layer, the information learned was flattened and concatenated into pipelines according to feature type (frequency or time) or to the size group of the kernels (short, medium, long). The pipelines were then concatenated again and passed to a dense hidden layer comprising of 5000 ReLU units before going to a dense output layer with a Softmax activation. Figure 11 provides a schema of this model.

The experiments are listed below with their abbreviations:

- Experiment 6: Short and medium kernels for frequency and time, concatenated according to feature type (SMF + SMT, FT). 10 kernels are used.
- Experiment 7: Short and medium kernels for frequency and time, concatenated according to their kernel size group (SMF + SMT, SM). 10 kernels are used.
- Experiment 8: Medium and long kernels for frequency and time, concatenated according to feature type (MLF + MLT, FT). 10 kernels are used.
- Experiment 9: Medium and long kernels for frequency and time, concatenated according to their kernel size group (MLF + MLT, ML). 10 kernels are used.
- Experiment 10: Short, medium & long kernels for frequency and time, concatenated according to feature type (SMLF + SMLT, FT). 15 kernels are used.
- Experiment 11: Short, medium & long kernels for frequency and time, concatenated according to their kernel size group (SMLF + SMLT, SML). 15 kernels are used.

The first four experiments concatenated adjacent kernel size groups, for example the short group was combined with the medium group and the medium group was combined with the long group, and these were then concatenated again either by feature type or kernel size. The last two experiments combined all three kernel size groups and concatenated them by either feature type or kernel size. Tables 9 to 14 display the kernel and pool dimensions.

Experiment 6	10 Frequency Kernels					10 Temporal Kernels				
Kernel Size	C1	C2	C3	C4	C5	C1	C2	C3	C4	C5
	4,4	8,4	12,4	16,4	20,4	4,4	4,8	4,12	4,16	4,20
	C6	C7	C8	C9	C10	C6	C7	C8	C9	C10
	24,4	28,4	32,4	36,4	40,4	4,24	4,28	4,32	4,36	4,40
Pool Size	P1	P2	P3	P4	P5	P1	P2	P3	P4	P5
	1,2	1,2	1,2	1,2	1,2	2,1	2,1	2,1	2,1	2,1
	P6	P7	P8	P9	P10	P6	P7	P8	P9	P10
	1,2	1,2	1,2	1,2	1,2	2,1	2,1	2,1	2,1	2,1
Concatenate	Frequency					Time				

Table 9 MRPP SM concatenated by feature

Experiment 7	10 Frequency Kernels					10 Temporal Kernels				
Kernel Size	C1	C2	C3	C4	C5	C1	C2	C3	C4	C5
	4,4	8,4	12,4	16,4	20,4	4,4	4,8	4,12	4,16	4,20
	C6	C7	C8	C9	C10	C6	C7	C8	C9	C10
	24,4	28,4	32,4	36,4	40,4	4,24	4,28	4,32	4,36	4,40
Pool Size	P1	P2	P3	P4	P5	P1	P2	P3	P4	P5
	1,2	1,2	1,2	1,2	1,2	2,1	2,1	2,1	2,1	2,1
	P6	P7	P8	P9	P10	P6	P7	P8	P9	P10
	1,2	1,2	1,2	1,2	1,2	2,1	2,1	2,1	2,1	2,1
Concatenate	Size					Size				

Table 10 MRPP SM concatenated by size

Experiment 8	10 Frequency Kernels					10 Temporal Kernels				
Kernel Size	C1	C2	C3	C4	C5	C1	C2	C3	C4	C5
	24,4	28,4	32,4	36,4	40,4	4,24	4,28	4,32	4,36	4,40
	C6	C7	C8	C9	C10	C6	C7	C8	C9	C10
	44,4	48,4	52,4	56,4	60,4	4,44	4,48	4,52	4,56	4,60
Pool Size	P1	P2	P3	P4	P5	P1	P2	P3	P4	P5
	1,2	1,2	1,2	1,2	1,2	2,1	2,1	2,1	2,1	2,1
	P6	P7	P8	P9	P10	P6	P7	P8	P9	P10
	1,2	1,2	1,2	1,2	1,2	2,1	2,1	2,1	2,1	2,1
Concatenate	Frequency					Time				

Table 11 MRPP ML concatenated by feature

Experiment 9	10 Frequency Kernels					10 Temporal Kernels				
Kernel Size	C1	C2	C3	C4	C5	C1	C2	C3	C4	C5
	24,4	28,4	32,4	36,4	40,4	4,24	4,28	4,32	4,36	4,40
	C6	C7	C8	C9	C10	C6	C7	C8	C9	C10
	44,4	48,4	52,4	56,4	60,4	4,44	4,48	4,52	4,56	4,60
Pool Size	P1	P2	P3	P4	P5	P1	P2	P3	P4	P5
	1,2	1,2	1,2	1,2	1,2	2,1	2,1	2,1	2,1	2,1
	P6	P7	P8	P9	P10	P6	P7	P8	P9	P10
	1,2	1,2	1,2	1,2	1,2	2,1	2,1	2,1	2,1	2,1
Concatenate	Size					Size				

Table 12 MRPP ML concatenated by size

Experiment 10	15 Frequency Kernels					15 Temporal Kernels				
Kernel Size	C1	C2	C3	C4	C5	C1	C2	C3	C4	C5
	4,4	8,4	12,4	16,4	20,4	4,4	4,8	4,12	4,16	4,20
	C6	C7	C8	C9	C10	C6	C7	C8	C9	C10
	24,4	28,4	32,4	36,4	40,4	4,24	4,28	4,32	4,36	4,40
	C11	C12	C13	C14	C15	C11	C12	C13	C14	C15
	44,4	48,4	52,4	56,4	60,4	4,44	4,48	4,52	4,56	4,60
Pool Size	P1	P2	P3	P4	P5	P1	P2	P3	P4	P5
	1,2	1,2	1,2	1,2	1,2	2,1	2,1	2,1	2,1	2,1
	P6	P7	P8	P9	P10	P6	P7	P8	P9	P10
	1,2	1,2	1,2	1,2	1,2	2,1	2,1	2,1	2,1	2,1
	P11	P12	P13	P14	P15	P11	P12	P13	P14	P15
	1,2	1,2	1,2	1,2	1,2	2,1	2,1	2,1	2,1	2,1
Concatenate	Frequency					Time				

Table 13 MRPP SML concatenated by feature

Experiment 11	15 Frequency Kernels					15 Temporal Kernels				
Kernel Size	C1	C2	C3	C4	C5	C1	C2	C3	C4	C5
	4,4	8,4	12,4	16,4	20,4	4,4	4,8	4,12	4,16	4,20
	C6	C7	C8	C9	C10	C6	C7	C8	C9	C10
	24,4	28,4	32,4	36,4	40,4	4,24	4,28	4,32	4,36	4,40
	C11	C12	C13	C14	C15	C11	C12	C13	C14	C15
	44,4	48,4	52,4	56,4	60,4	4,44	4,48	4,52	4,56	4,60
Pool Size	P1	P2	P3	P4	P5	P1	P2	P3	P4	P5
	1,2	1,2	1,2	1,2	1,2	2,1	2,1	2,1	2,1	2,1
	P6	P7	P8	P9	P10	P6	P7	P8	P9	P10
	1,2	1,2	1,2	1,2	1,2	2,1	2,1	2,1	2,1	2,1
	P11	P12	P13	P14	P15	P11	P12	P13	P14	P15
	1,2	1,2	1,2	1,2	1,2	2,1	2,1	2,1	2,1	2,1
Concatenate	Size					Size				

Table 14 MRPP SML concatenated by size

To compare with experiments 1 and 2, where the architectures were limited to prioritizing only one feature (either frequency or time) and therefore anticipated to have incomplete learning, the architectures in experiments 3 to 9 could prioritize both features. However, with deliberate exclusions of size category these were also expected to have incomplete learning of the dataset. By comparison, in the last two experiments (10 and 11) these variants prioritized features along both axes and had access to a wider range of kernel sizes. Therefore, they were expected to outperform all the other variants.

The objective of these experiments was to determine whether this assumption was correct and if not, which combination of size groups was able to obtain higher accuracy. The second objective was to find out whether concatenating the pipelines by feature type or by size group category influenced the accuracy.

3.6 Model Depth

Although, not one of the primary objectives, it was decided that it would be useful to compare the results of the MRPP CNNs before and after adding an identical hidden dense layer after the existing one. The expectation was that this would potentially produce slightly better results by facilitating the modelling of slightly higher-level features.

3.7 Data Augmentation

Again, while not one of the primary objectives, it was decided that it would be useful to explore the effect of data augmentation on one or more of the presented architectures. In an additional experiment, the UrbanSound8K dataset was augmented in real-time mini-batches and applied to the MRPP CNN variant with short, medium and long kernels, concatenated by kernel size. This variant was selected arbitrarily, as one of the two variants with a wider range of kernel sizes.

The augmentation technique was executed in the *Keras ImageDataGenerator* class, adapted from scripts provided by (Daisuke, 2017). Although not clearly specified in the source, this script also leverages a class called *mixup-generator* designed for use with random erasing in image inputs. (Uchida, 2017). Random erasing was omitted from the final augmentation technique, which was restricted to shifting the inputs horizontally by up to 40% of the total width and randomly flipping inputs horizontally. The expectation was that augmenting the dataset would improve the results as the number of training samples would increase.

3.8 Training, Validation and Testing

Experiments followed the training, validation and testing processes and parameters as the established baseline with one exception. After initial tests encountered difficulties with the memory requirements of some of the experiments, the batch size was lowered to 40, with a view to increase this if opportunity permitted while the batch-size baseline was kept to 1000 to preserve its resemblance to (Piczak, 2015). While the design and construction of the neural networks was carried out on a local machine, given the potentially long run times, the experiments were run on a server equipped with an Nvidia GTX10 GPU (1080 Ti). The experiments varied between 50-90 minutes in running time over the course of 3 weeks. This took place over a shared server, where the day-time hours proved to be the optimal times to run the models.

After each experiment, an additional run of the test was carried out for the purpose of plotting a single confusion matrix, instead of aggregating from the results of 10 folds. The script for this was included in the implementation by Zhu, Kaznady and Hendry (2017).

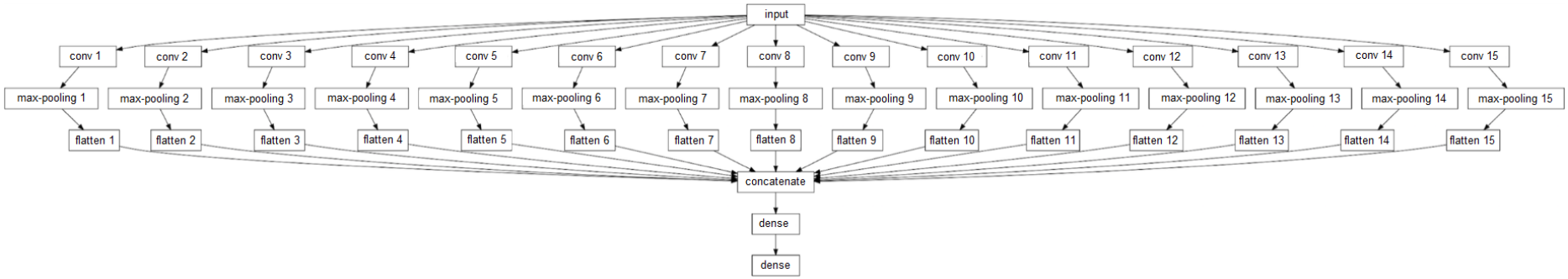


Figure 10. MRSL CNN architecture

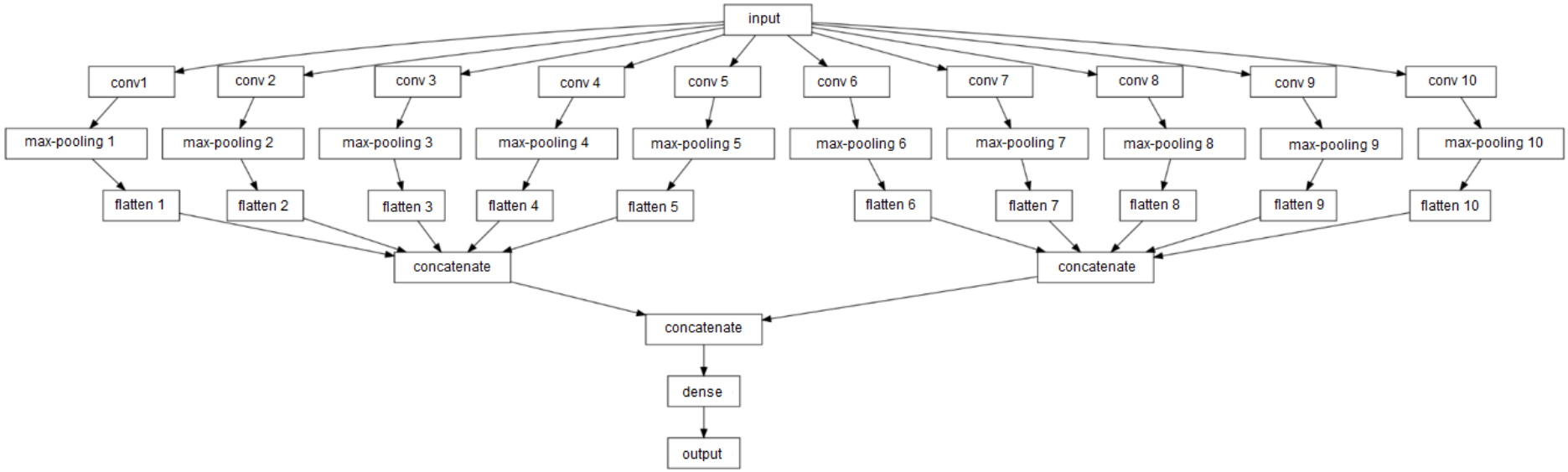


Figure 11. A schema of the MRPP CNN architecture consisting of a single convolutional with 10 kernels and corresponding pooling layer, followed by a dense layer with 5000 ReLU units and a dense Softmax output layer.

4 Results

4.1 Baseline Results

Initial tests revealed that both baselines underperformed (Piczak, 2015). However, since the performance using 3-channel inputs produced a slightly smaller discrepancy than 2-channel inputs, the former was established as the baseline for the Project. Thus, a mean accuracy of **60.39%** became the benchmark score against which to evaluate subsequent architectures. The discrepancy of **12.71%** from (Piczak, 2015) seems inevitably to be the result of differences in the implementation from the source code, likely a combination of design and technical factors.

Baseline	3-channel	2-channel
Mean	**60.39%**	56.38%
Std	9.09%	9.11%
Convergence	90mins	90mins

Table 15. Baseline results. Convergence times are approximate.

Like (Piczak, 2015), the 3-channel baseline struggled to distinguish between classes dominated by short temporal features (Drill, Jackhammer and Engine) but it is also apparent from the confusion matrix (Figure 12) that the baseline's decline in accuracy is also attributed to problems distinguishing between the Children Playing, Siren and Street Music classes. Nevertheless, it appears that the baseline score could have continued improving since it had not yet reached convergence by the 150th epoch. Increasing the number of epochs could have led to higher accuracies but it was decided that any further departures from (Piczak, 2015) would be undesirable at this stage.

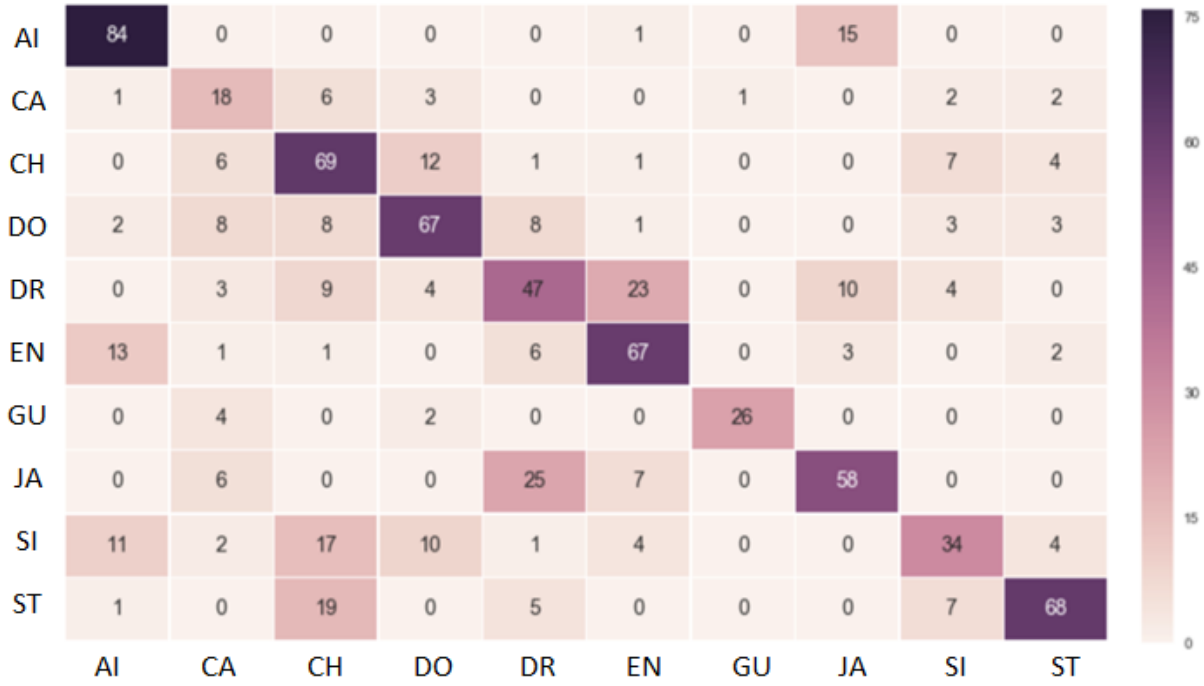


Figure 12. Baseline confusion matrix in absolute numbers. Ground truth classes are displayed horizontally and predicted classes are displayed vertically. For abbreviations see Table 1.

3-channel Baseline	
Best Classes	Car horn, Gun shot
Worst Classes	Drill, Jackhammer, Siren, Street Music

Table 16. Best and worst performing classes in the baseline model.

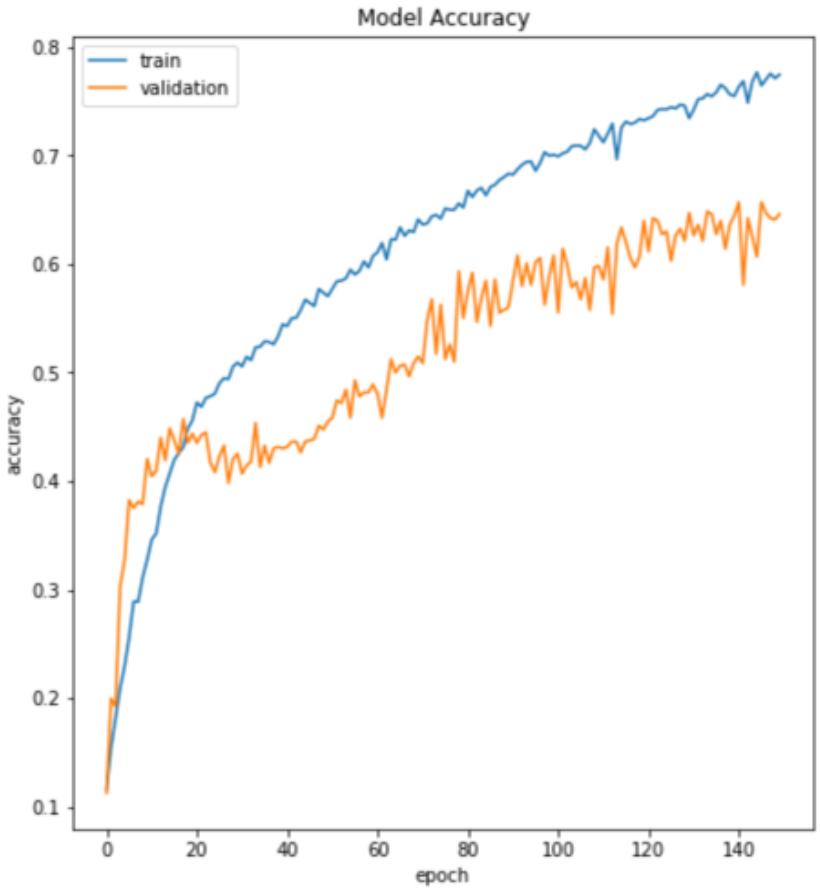


Figure 13. Baseline training and validation accuracies. It appears that the model could have potentially continued improving with additional epochs.

4.2 Results of MRSL CNNs

The first set of experiments examined MRSL CNNs modelling either frequency or time features. Consequently, the architecture prioritizes information along one axis and neglects the other, lowering expectations for its accuracy. Of the two variants, the architecture modelling frequency produced the better score with **61.98%** (an improvement of 1.59% over the baseline) while the architecture modelling time achieved **60.85%**, just slightly less. What stands out from these results is that even when only one feature is prioritized, the scores are still comparable to the baseline, thus the impact of the kernel shapes was negligible.

Experiment	1	2
Feature	Frequency	Time
Mean	**61.98%**	60.86%
Std	4.65%	4.24%
Convergence	50mins	50mins

Table 17. MRSL CNN Results. All convergence times are approximate.

There were also only minor differences between the two models in terms of class-specific performance. Both variants encountered most difficulties with the same classes, namely the Air Conditioner, Jackhammer, Siren and Street Music sounds, while they had least trouble with the Car Horn and Gun Shot classes. This outcome was similar to the baseline. By comparison, in (Piczak, 2015) the Air Conditioner was among the most accurate classes.

Some of the misclassifications observed were asymmetrical. For example, there were many instances of Engine Idling and Jackhammer classes labelled as Air Conditioner classes, whereas the same was not true for Air Conditioner classes labelled as Engine Idling and Jackhammer classes. This is interesting because similar classes are expected to be symmetrical.

Moreover, while baseline training ended after 150 epochs, the MRSL CNNs stopped training early, between the 21st and 26th epochs. In approximately 50mins, these architectures achieved comparable results to the baseline, which required closer to 90mins. Their faster convergence rate indicated that MR-SL CNNs learned more efficiently than the baseline.

MRSL (Frequency)		MRSL (Time)	
Best Classes	Worst Classes	Best Classes	Worst Classes
Car horn	Siren	Car horn	Air Con
Gun shot	Air Con	Gun shot	Jackhammer
	Jackhammer		Siren
	Street Music		Street Music

Table 18. Best and worst performing classes in the MRSL CNN models.

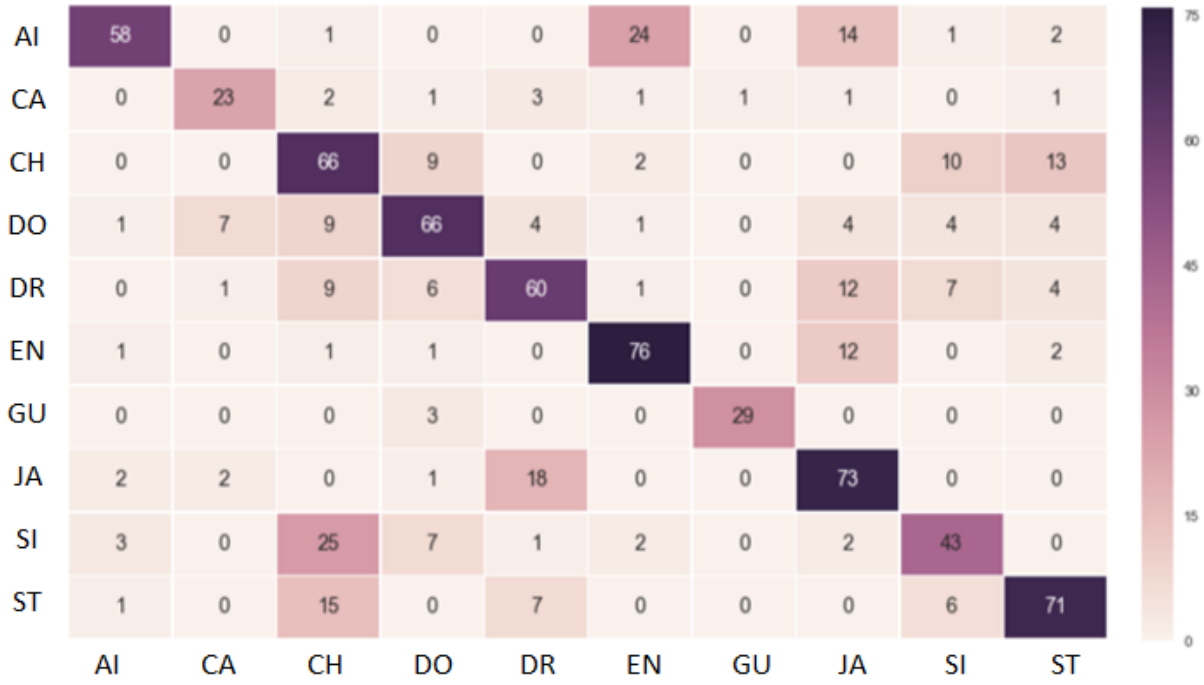


Figure 14. MRSL CNN (Frequency) confusion matrix showing absolute numbers. For abbreviations see Table 1.

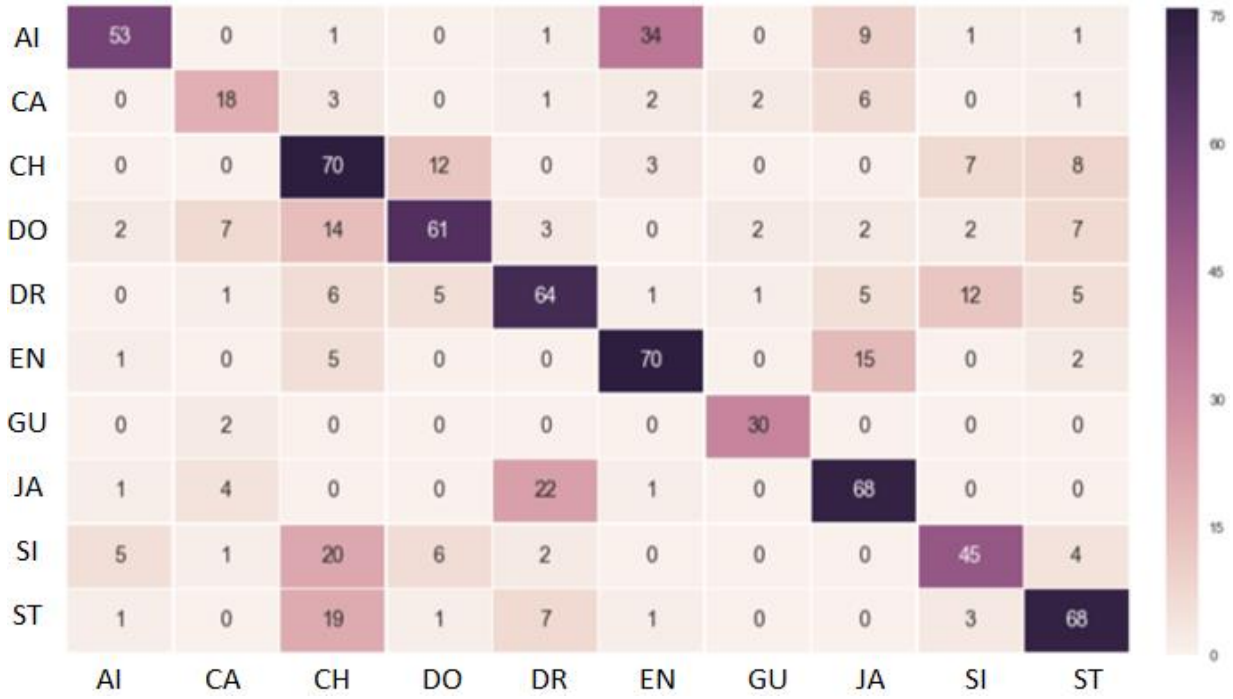


Figure 15. MRSL CNN (Time) confusion matrix showing absolute numbers. For abbreviations see Table 1.

MRSL CNN Results Summary

- Prioritizing one axis over the other impacted overall classification accuracy.
- Regardless of the axis neglected, the worst classes were the Air Conditioner, Jackhammer, Siren and Street Music sounds. This suggests that the representation of these classes depend heavily on both frequency and time features.
- Asymmetrical misclassifications were detected.
- Compared to the baseline, MR-SL CNNs had greater difficulty predicting the Air Conditioner class bhan the Drill class.
- Compared to the Baseline, MR-SL CNNs converged faster.

4.3 Results of MRPP CNNs (Single Category)

In experiments 6 to 11, Multi-Resolution Parallel Pipeline CNNs learned both frequency and temporal features but were restricted to using only a single kernel size category (short, medium or long). The objective was to determine whether the restriction of kernel sizes would be more detrimental to performance than neglecting an entire axis and to identify which kernel sizes appeared to be most significant to this dataset. As with the previous architecture, expectations were limited given the significant amount of information deliberately excluded from network learning.

Out of the three variants, the short variant (SF + ST) obtained the highest accuracy with **61.95%**. At the same time, results from the medium variant were very close, with a score of **61.69%**, only 0.26% less, and even the accuracy of the long variant was not far behind with **60.61%**. These scores were comparable both with the baseline and the results of the MRSL CNN.

Experiment	3	4	5
Feature	SF + ST	MF + MT	LF + LT
Mean	**61.95%**	61.69%	60.61%
Std	4.58%	5.33%	5.35%
Convergence	40mins	40mins	40mins

Table 19. Results of MRPP CNNs using a single kernel size unit. Convergence times are approximate.

Although these are close scores, the comparison seemed to suggest that short and medium kernel units are better at representing the features in the UrbanSound8K dataset. It was possible to speculate at this stage that combining the short and medium filters could lead to the highest results across the entire dataset.

To observe class-wise performance easily across variants, the confusion matrices have been combined (Figure 16). While all three groups generated large numbers of misclassifications, distinctions between each group are interesting. For example, although the short variant outperformed the others, it was also the only model to misclassify Dog barks as Drill sounds and Jackhammer sounds as Engine Idling noises. The results also seemed to support an earlier assumption that the short variant would outperform the baseline in classes with short temporal features. This appeared correct for the Drill and Jackhammer classes and with regards to the Engine class, the short variant achieved the same number of correct predictions as the baseline.

SF + ST		MF + MT		LF + LT	
Best Classes	Worst Classes	Best Classes	Worst Classes	Best Classes	Worst Classes
Car horn	Air Con	Car horn	Air Con	Car horn	Siren
Gun shot	Siren	Gun shot	Siren	Gun shot	Children
	Jackhammer		Jackhammer		Air Con
	Music/Dog		Drill		Drill/Music

Table 20. Best and worst classes in MRPP CNNs using only a single size category.

		air con	car horn	children	dog bark	drill	engine	gun	hammer	siren	music
air con	LF+LT	64	0	4	0	0	15	0	13	1	3
	MF+MT	49	0	1	0	0	35	0	14	1	0
	SF+ST	57	0	0	0	0	31	0	10	1	1
car horn	LF+LT	0	19	1	2	1	1	3	4	0	2
	MF+MT	0	22	1	1	1	0	2	4	0	2
	SF+ST	0	25	0	3	3	1	0	0	0	1
children	LF+LT	0	0	63	13	0	1	0	0	7	16
	MF+MT	0	0	66	10	0	3	0	0	8	13
	SF+ST	0	0	71	11	0	3	0	0	5	10
dog	LF+LT	1	5	12	69	2	1	0	6	1	3
	MF+MT	1	2	8	69	3	1	2	5	4	5
	SF+ST	1	7	17	63	1	1	1	3	2	4
drill	LF+LT	0	0	5	7	64	0	0	10	10	4
	MF+MT	0	1	8	7	55	3	0	13	12	1
	SF+ST	2	0	5	10	49	7	0	15	6	6
engine	LF+LT	3	0	6	1	0	74	0	8	0	1
	MF+MT	1	0	3	1	0	80	0	5	1	2
	SF+ST	4	0	6	2	0	67	0	13	0	1
gun	LF+LT	0	0	0	3	0	0	29	0	0	0
	MF+MT	0	2	0	3	0	1	26	0	0	0
	SF+ST	0	2	0	3	0	0	27	0	0	0
hammer	LF+LT	1	2	1	1	19	0	0	72	0	0
	MF+MT	0	2	0	0	26	2	0	66	0	0
	SF+ST	1	2	0	0	18	1	0	74	0	0
siren	LF+LT	3	1	24	11	0	2	0	1	39	2
	MF+MT	2	0	19	14	0	1	0	2	42	3
	SF+ST	2	1	26	5	2	2	0	0	43	2
music	LF+LT	0	0	20	0	7	0	0	0	4	69
	MF+MT	1	0	14	0	5	0	0	0	3	77
	SF+ST	1	0	17	0	6	0	0	0	5	71

Figure 16. Confusion matrices showing absolute numbers. Misclassifications over 10 are highlighted in pale red.

Multi-Resolution Parallel Pipeline CNN (Single Category) Results Summary

- The short variant outperformed both the medium and long variants.
- The long variant performed the worst of the three.
- Again, asymmetrical misclassifications were detected.
- The architectures converged faster than the baseline.

4.4 Results of MRPP CNNs (Combined Categories)

In an extension of the previous experiments, these architectures made it possible to measure the effect of spreading out the range of kernel sizes by combining adjacent size categories or all size categories. This gave rise to the option to concatenate the pipelines either by feature type (as in the previous Single-Category architectures) or by size category. The expectation was for the variants with a full combination of size categories to achieve the best performance since they utilized a larger range of kernel sizes and could therefore potentially capture more information. The objective was to determine if combining adjacent categories or all size categories improved the accuracy and to observe the effect of each concatenation option.

Against expectations, variants combining all size categories did not outperform those excluding a size category. In fact, it appeared that short and medium kernels concatenated according to feature type (SMF + SMT FT) achieved the highest

performance with **63.66%**. This outcome supported the earlier assumption that combining short and medium variants could contribute to higher scores, and it suggested that the most important features in the UrbanSound8K dataset were represented by these kernel shapes. The next highest score, **63.10%**, was obtained by a variant combining all size categories concatenated according to size category. This suggests that longer features may have been difficult for the model to learn (even with access to a wider range of kernel size) but concatenating the pipelines according to size category allowed the network to better learn these differences. Confusion matrices have been combined for variants concatenated by feature type in Figure 17 and size category in Figure 18. Misclassifications over 10 have been highlighted in pale red.

Experiment	6	7	8	9	10	11
Sizes	SMF + SMT	SMF + SMT	MLF + MLT	MLF + MLT	SMLF + SMLT	SMLF + SMLT
Concat	Feature	Size	Feature	Size	Feature	Size
Mean	**63.66%**	62.53%	62.69%	62.87%	62.93%	63.10%
Std	5.20%	5.28%	4.41%	4.27%	5.30%	5.12%
Convergence	40mins	40mins	40mins	40mins	40mins	40mins

Table 21. Results from the second set of parallel pipeline architectures, where the kernel size combinations are denoted in the second row and the concatenation option is denoted by Feature or Size.

SMF + SMT (Feature)		MLF + MLT (Feature)		SMLF + SMLT (Feature)	
Best Classes	Worst Classes	Best Classes	Worst Classes	Best Classes	Worst Classes
Car horn	Jackhammer	Car horn	Air Con	Car horn	Siren
Gun shot	Siren	Gun shot	Siren	Gun shot	Air Con
	Children		Drill		Jackhammer
	Drill		Jackhammer		Drill
SMF + SMT (Size)		MLF + MLT (Size)		SMLF + SMLT (Size)	
Best Classes	Worst Classes	Best Classes	Worst Classes	Best Classes	Worst Classes
Car horn	Siren	Car horn	Air Con	Car horn	Air Con
Gun shot	Air Con	Gun shot	Drill	Gun shot	Jackhammer
	Drill		Siren		Children
	Street Music		Jackhammer		Siren

Table 22. Best and worst classes in MRPP CNNs with combined size categories.

Multi-Resolution Parallel Pipeline (Combined Categories) CNN Results Summary

- The combined short and medium variant, concatenated by feature type achieved the highest score in the Project without augmenting data, with **63.66%.**
- The significance of the concatenation selection was questioned as the medium and long variants performed slightly better when concatenated according to size categories but the short variant concatenated by feature type outperformed all variants.
- Again, the presence of asymmetrical misclassifications were detected.
- Consistent with MRPP CNNs (Single Category), these architectures converged faster than the baseline, stopping between the 21st and 26th epoch.

		air con	car horn	children	dog bark	drill	engine	gun	hammer	siren	music
air con	SM	76	0	4	0	0	13	0	4	1	2
	ML	56	0	5	0	0	20	0	16	1	2
	SML	65	0	0	0	0	26	0	7	1	1
car horn	SM	0	23	2	1	3	1	1	1	0	1
	ML	0	23	0	1	0	1	2	4	0	2
	SML	0	24	0	0	2	1	2	2	0	2
children	SM	0	0	69	11	0	3	0	0	7	10
	ML	0	0	73	9	0	3	0	0	6	9
	SML	0	0	69	12	0	2	0	0	6	11
dog	SM	1	4	13	64	7	0	2	4	2	3
	ML	1	5	10	63	3	1	1	6	5	5
	SML	1	5	11	68	4	1	1	4	3	2
drill	SM	0	0	5	8	62	3	0	11	10	1
	ML	2	1	5	9	47	2	0	22	9	3
	SML	3	0	4	6	61	2	0	11	10	3
engine	SM	3	0	1	1	1	71	0	14	1	1
	ML	2	0	2	2	0	73	0	14	0	0
	SML	1	0	2	0	1	78	0	10	0	1
gun	SM	0	0	1	2	0	0	27	2	0	0
	ML	0	0	0	3	0	0	29	0	0	0
	SML	0	0	0	3	0	0	29	0	0	0
hammer	SM	3	2	1	0	45	1	0	44	0	0
	ML	0	1	0	0	21	1	0	73	0	0
	SML	1	2	1	0	25	0	0	67	0	0
siren	SM	3	1	17	12	0	2	0	2	44	2
	ML	2	1	23	9	0	1	0	3	43	1
	SML	4	0	19	11	1	2	0	0	43	3
music	SM	1	0	12	0	5	0	0	0	5	77
	ML	1	1	12	0	3	0	0	0	8	75
	SML	2	0	12	0	5	0	0	0	8	73

Figure 17. Confusion matrices for MRPP with combined categories, concatenated by feature type.

		air con	car horn	children	dog bark	drill	engine	gun	hammer	siren	music
air con	SM	66	0	1	0	0	20	0	11	1	1
	ML	57	0	3	0	0	20	0	17	1	2
	SML	64	0	1	0	0	27	0	6	1	1
car horn	SM	0	22	2	3	1	1	1	2	0	1
	ML	0	22	0	0	1	1	2	5	0	2
	SML	0	23	0	1	3	1	2	1	0	2
children	SM	0	0	77	9	0	2	0	0	4	8
	ML	0	0	70	12	0	3	0	0	5	10
	SML	0	0	72	11	0	2	0	0	3	12
dog	SM	3	3	15	65	3	1	2	4	2	2
	ML	1	5	14	63	3	1	2	5	3	3
	SML	1	6	13	66	3	1	0	5	4	1
drill	SM	1	0	4	6	51	4	0	18	11	5
	ML	0	1	3	11	49	1	0	21	11	3
	SML	3	0	5	5	60	1	0	12	9	5
engine	SM	1	0	6	1	0	76	0	9	0	0
	ML	2	0	1	2	0	75	0	12	0	1
	SML	2	0	2	0	1	77	0	10	0	1
gun	SM	0	0	0	3	0	0	29	0	0	0
	ML	0	0	0	3	0	0	29	0	0	0
	SML	0	0	0	3	0	0	29	0	0	0
hammer	SM	1	2	0	0	15	2	0	76	0	0
	ML	0	2	0	0	21	2	0	71	0	0
	SML	1	2	1	0	24	0	0	68	0	0
siren	SM	2	0	24	11	0	0	0	0	46	0
	ML	3	1	24	8	1	1	0	2	43	0
	SML	4	0	22	9	2	1	0	0	42	3
music	SM	1	0	16	0	3	0	0	0	5	75
	ML	1	0	12	0	4	0	0	0	8	75
	SML	1	0	10	0	6	0	0	0	9	74

Figure 18. Confusion matrices for MRPP with combined categories, concatenated by size category.

4.5 Model Depth

Adding a second hidden layer before the Softmax output layer produced a slight improvement to the Multi-Resolution Parallel Pipeline CNNs using a Single Size Category. The additional layer was identical to the first one and contained 5000 ReLU units. Again, the short variant obtained the highest accuracy, with **62.43%**, outperforming both medium and long variants. The order in which the variants ranked was consistent with the previous set of results for this variant.

Increasing the number of hidden layers in experiments 15 to 20 produced an inconsistent effect on accuracy. Only half of the architectures benefitted from this inclusion and in fact the highest overall score in this set underperformed that of the corresponding result from a single hidden layer. This outcome suggests that there is limited potential for improvement by increasing the model depth further with hidden layers. Consistent with their single-layer versions, these architectures converged faster than the baseline, stopping between the 21st and 26th epoch.

Experiment	12	13	14
Feature	SF + ST	MF + MT	LF + LT
Mean	**62.43%**	61.69%	60.97%
Std	4.54%	5.33%	5.70%
Convergence	40mins	40mins	40mins

Table 23. MRPP (Single Category) CNN with two hidden dense layers before the Softmax output layer. All convergence times are approximate.

Experiment	15	16	17	18	19	20
Sizes	SMF + SMT	SMF + SMT	MLF + MLT	MLF + MLT	SMLF + SMLT	SMLF + SMLT
Concat	Feature	Size	Feature	Size	Feature	Size
Mean	62.81%	63.07%	62.74%	62.71%	**63.45%**	63.03%
Std	4.71%	4.06%	5.45%	5.39%	5.23%	4.68%
Convergence	40mins	40mins	40mins	40mins	40mins	40mins

Table 24. MRPP (Combined Categories) CNN with two hidden dense layers before the Softmax output layer. All convergence times are approximate.

4.6 Data Augmentation

The UrbanSound8K dataset was augmented in real-time batches with random horizontal shifts up to 40% of the input width and randomly flipping the inputs horizontally. Due to time constraints and the longer training time required, only an additional experiment, using one of the combined unit variants of the Multi-Resolution Parallel Pipeline was tested on the augmented dataset. This obtained a **64.03%** accuracy. Although this was the highest score of all experiments in the Project, the network still struggled to differentiate between the same classes that it did in the original dataset. For example, there were instances of the Jackhammer class misclassified as Drill and Engine noises, as well as instances of Children Playing labelled as Sirens. This result is attributed to the choice of augmentation technique. On reflection, these augmentations possessed limited representational value since inverted audio samples do not resemble their original sounds. It would be beneficial to investigate other choices of augmentation technique in further work.

Experiment	21
Augmented	Yes
Sizes	SMLF + SMLT
Merge	Size
Mean	**64.03%**
Std	3.87%
Convergence	6.27 hours

Table 25. Experiment 20 repeated on augmented data.

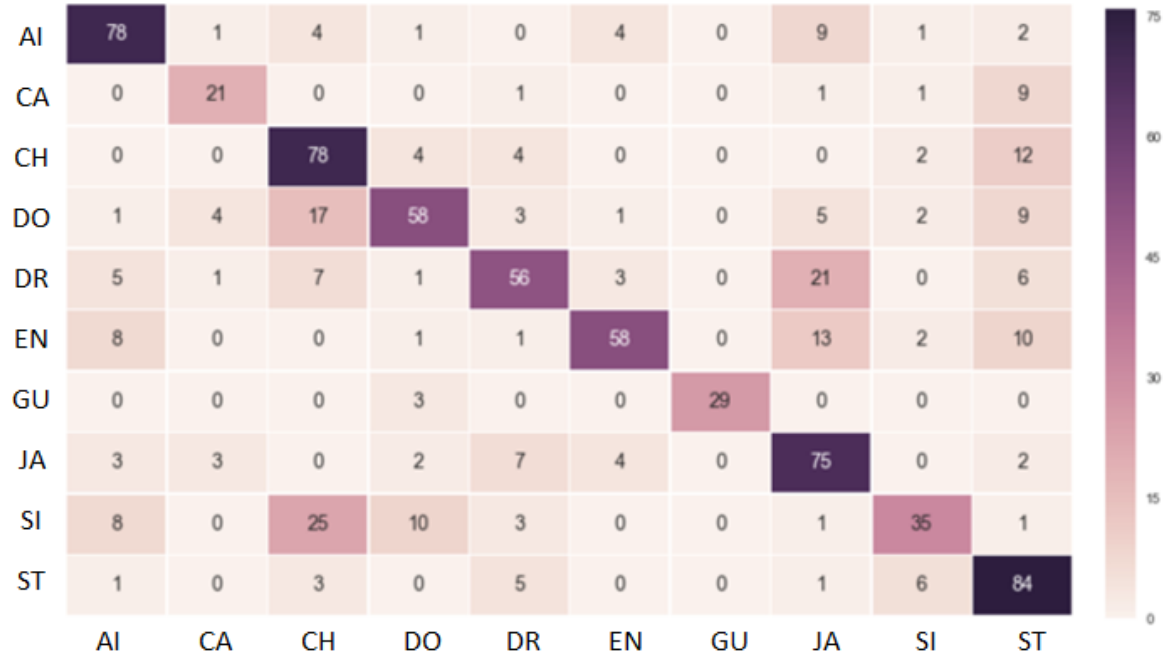


Figure 19. Confusion matrix from results of (SMLF + SMLT) concatenated by size, on augmented data.

4.7 Analysis of Results

Kernel size

Short and medium variants appeared to be the most important kernel sizes for learning features in the UrbanSound8K dataset. This was observed consistently over the four sets experiments without data augmentation, over which architectures contained either a single hidden layer or two hidden layers. This suggests that features in the UrbanSound8K dataset tend to be smaller and less dependent on long patterns on either axis. This supports intuition since, unlike music, urban sounds lack tempo or rhythm. The exception to this would be the Street Music class, in which rhythms from different music genres could be perceived by ear. However, some background noise was identified in an earlier review of the dataset, which may have increased the difficulty for the network to ascertain longer time-scale features in this class. The assessment across the whole dataset is that short and medium kernel sizes were preferable for learning the distinctions between urban sounds.

Kernel Shape

MRSL CNNs demonstrated that re-shaping kernels to prioritize one axis over the other had a negligible impact on the overall classification, with results that were comparable

with the baseline. Moreover, regardless of the axis prioritized, the worst-suffering classes were the same, suggesting that the representation of these classes depended heavily on both frequency and time features. It appears that for urban noise, there is little to suggest that changing the kernel shape to prioritize frequency or time could improve accuracy. This outcome diverged from the results in (Pons, Lidy and Serra, 2016) and (Pons and Serra, 2017) and again highlights the difference between music and general noise.

Concatenation Selection

The results seem to challenge the impact of the concatenation choice to the performance. In the single-layer variants of the Multi-Resolution Parallel Pipelines, in two out of three instances, the variants concatenated by size outperformed the variants concatenated by feature type. These were the (MLF + MLT) and (SMLF + SMLT) variants. The reverse effect was observed in a second set of experiments in which a second hidden layer was added. In two out of three instances, variants concatenated by feature type outperformed those concatenated by size and these were again the (MLF + MLT) and (SMLF + SMLT) variants. This seems to indicate that concatenating the parallel pipelines by feature or size makes negligible impact on the outcome of the models. This makes intuitive sense given that the models only have a single convolutional layer, from which potentially only low-level features may be been captured. It is possible that this option could be more significant for deeper models modelling higher level features.

Convergence Rates

Multi-Resolution CNNs of all variants converged faster than the baseline when trained on the original dataset. Augmenting the dataset in real-time batches increased the training time significantly.

Model Depth

In the MRPP (Single) architectures, adding a second hidden layer appears to have improved the performance slightly in the short and long variants and made no impact at all on the medium variant. The short variant obtained the highest accuracy in this set regardless of whether it possessed one or two hidden layers. By comparison, only half of the variants improved with an additional hidden layer in the MRPP (Combined) architectures. This suggests limited potential for improvement in continuing to increase the number of hidden layers. Conclusions from the current state-of-the-art systems (Salamon and Bello, 2017) and (Tokozume, Ushiku & Harada, 2018), underline the imperative for data augmentation techniques, which may then subsequently validate the application of deeper models.

Data Augmentation

The effect of using augmented data made a slight improvement to the accuracy of the variant tested (SMLF + SMLT, concatenated by size). Although the network achieved the highest score in the Project (64.03%), it still struggled to distinguish between the same classes as on the original dataset, which underlines the similarity of these classes but also points out limitations in the choice of augmentation technique. From reviewing the results and upon reflection, it is evident that augmentation by random horizontal shifts (up to 40% of the width) and flips added limited representational value to the model.

5 Discussion and Implications

5.1 Evaluation of Results and Objectives

The objectives were:

- Establish a suitable baseline, informed by findings in relevant literature.
- Measure the effect of various kernel shapes, sizes and any combinations thereof on classification accuracy.
- Determine any compelling findings that could be used to compliment other methods in urban sound classification.

  The following three methods were outlined to aid the fulfilment of objectives. The results have been reviewed against these methods in 5.1.1. to 5.1.3.

- Establish a baseline CNN, modelled on (Piczak, 2015)
- Design and develop CNN architectures to investigate the effect of applying different kernel shapes, sizes and combinations on classification accuracy in urban sound classification. To this end, Multi-Resolution CNNs will be developed. In exploring different implementations and scripts, consider appropriate options for building these architectures in the chosen deep learning framework. For example, different combinations could be modelled and interpreted more easily with the use of separate pipelines.
- Train CNNs and evaluate their performance against the baseline.
  Use the same implementation script for training, validation and testing as the baseline, if possible from the original code used in (Piczak, 2015).

5.1.1 Establish a Baseline, Modelled on (Piczak, 2015)

Full replication of (Piczak, 2015) was unsuccessful but this objective was completed by adapting an alternative implementation that had been identified at an earlier stage. A viable baseline was established, preserving as much of Piczak's original methods as possible. One area where the implementation diverged from these methods was in choice of deep learning framework. Another was in the use of a 3-channel input, rather than two. Justifications were provided for these changes. The greatest concern undermining this baseline remains the significant discrepancy between its performance and that of (Piczak, 2015). Although this stage was carried out to the best of the Author's ability at the time, a closer replication of the intended benchmark would have added significant value to subsequent analysis. Furthermore, the difficulties associated with this task caused delays which impacted the work schedule. Completion of this task could have been potentially accelerated by seeking earlier assistance from university staff or contacting Piczak. However, a viable baseline was established, modelled on the methods described by Piczak, and has provided value as a baseline for evaluating the Multi-Resolution CNN architectures. Therefore, the task has fulfilled this objective.

5.1.2 Design and Develop Multi-Resolution CNNs

Different variants of Multi-Resolution CNN were designed to investigate the effect of multiple kernel resolutions, shapes and concatenation choices on convolutional

neural networks for urban sound classification. In addition, model depth and data augmentation were tested.

There is also further scope for work on this task, regarding the choice of kernel size increments. Since the maximum kernel size was set at 60, whereas the width of the input extended to 101 frames, further information could have been learnt from the input by increasing the number of kernels prioritizing temporal features. On balance, given that the resulting architectures were comprehensive, interpretable and easily customized, this task fulfilled the objectives.

5.1.3 Train Models and Evaluate their Performance

Training took place over 13 days sharing access to a server with a GPU. This objective was completed without issues. However, a considerable amount of time was required to complete tests on all variants. Occasionally, server connection was lost, requiring training to be restarted. Moreover, augmenting the data in real-time batches significantly added to total time needed overall, curtailing the testing of any other variants with this augmentation technique. Evaluation of the architectures was carried out without issues but the visual clarity of the confusion matrices could have benefitted from the use of percentages, given there was some class imbalance.

5.2 New Findings

When comparing the results of the Project with (Pons, Lidy and Serra, 2016) and (Pons and Serra, 2017), it's important to reinforce the fact that urban sounds are inherently different to music and that this is expected to reflect in the results.

One of the most compelling findings from the results was that short and medium kernels were consistently better at learning the UrbanSound8K dataset than larger kernels. Since noises generally lack tempo or rhythm, which are characteristic of music, they are less dependent on long temporal patterns. The Street Music class is an exception to this.

A related finding is that in contrast to results in music classification (Pons, Lidy & Serra, 2016), changing the kernel shape to prioritize features along one axis appeared to have negligible impact on classification of urban noise.

Moreover, increasing the number of hidden layers pointed to limited improvement in network learning. Awareness of this outcome is useful since this can inform decisions to divert time and resources to other, more promising lines of research.

Other findings included the absence of support for the relevance of the concatenation choice on these architectures, presumably because there was only one convolutional layer, from which perhaps only low-level features would have been learned. However, even after adding a second hidden layer, the inconsistent results suggested that the concatenation choice was inconsequential for the architectures presented. Nevertheless, the possibility that this option could still be relevant for significantly deeper models capturing higher-level features has not been tested and should not be excluded.

5.3 Implications of Findings

The purpose was of the Project was to explore the effect of kernel shape, size, size range and concatenation type to determine if these could be used to maximize the

representational capacity of first convolutional neural network layer. The motivation was to use findings to compliment state-of-the-art methods that depend on data augmentation and deep learning methods and potentially advance their performance.

The finding that appears most applicable to other methods in urban sound classification is a preference for short kernels or short and medium kernels combined. In almost every experiment conducted without augmented data, the short variants achieved the highest accuracy. It would be interesting to take this finding and explore its effect on other popular CNN architectures for audio classification, also in conjunction with different data augmentation techniques.

5.4 Response to the Research Question

The Research Question was:

What is the effect of Multi-Resolution CNNs on urban sound classification?

The Multi-Resolution CNNs were developed to test the impact of kernel shape, size, size range and concatenation type. The most compelling finding was that short and medium sized kernels consistently achieved higher classification accuracies than long kernels. Changing the kernel shape to prioritize one axis over the other had a negligible impact. Support for choice of concatenation in parallel pipelines was absent in the results, both in variants with a single hidden layer before the output layer and with two hidden layers. Finally, increasing the number of hidden layers showed limited improvement to accuracy.

6 Evaluation, Reflections, and Conclusions

6.1 Choice of Objectives

The choice of objectives was achievable and though there was room for improvement, sufficient progress was made in all of them to satisfy the research question fully.

6.2 What could have been improved?

With hindsight, it is easy to recognize the benefit of seeking assistance for setbacks in replicating Piczak's source code. Moreover, in the earlier Project planning stages, it seems that progress could have been accelerated with prioritizing the review of datasets and materials ahead of or at least in conjunction with the literature survey. This would have made it possible to avoid some of the unnecessary changes and could potentially have enabled some of the technical challenges to be identified and addressed earlier.

6.3 What was learned from the Project?

The Project's objectives offered an opportunity to become more familiar with deep learning in audio data. This is an area of research that has undergone rapid growth in recent years, which has incentivized the Author to understand the latest developments as well as invest time to developing useful technical and research skills. Carrying out this Project has also deepened the appreciation for current achievements in this field and inspiration to continue following the latest developments in this area.

6.4 Proposals for further work

It would be highly beneficial to explore the effects of different data augmentation techniques. These could include the addition of background noise, including echoes and reverb, or changing the volume of entire samples or within samples. Learning from the findings, the guiding principle should be to select augmentations that result in realistic representations of the classes. To this effect, it would also be useful to include the time-stretching and pitch-shifting techniques available in *librosa* to this list and explore the effect of applying all techniques to the Multi-Resolution CNNs presented in this Project. Another interesting area to explore in further work would be to determine the effect of using other audio representations. Alternatives include the STFT spectrogram and CQT spectrogram, each offering different properties. While mel-spectrograms are optimized for the human auditory system and thus a popular choice for perceptual and musically-motivated tasks such as audio-tagging (Choi, Fazekas & Sandler, 2016) and music recommendations systems (van den Oord, Dieleman, Schrauwen, 2013), the Short Time Fourier Transform (STFT) spectrogram uses linearly-spaced center frequencies and has the advantage of being able to invert the audio signal, making it a suitable for source separation and the Constant-Q Transform (CQT) uses geometrically spaced frequency bins and equal ratios of centre frequencies to bandwidths (Q-factors). While this has been shown to be appropriate for musically-motivated tasks, it was also been recently used for the challenge of acoustic scene classification (Lidy and Schindler, 2016) where it outperformed mel-transform spectrograms. Like urban sound classification, this task addresses the classification of noises in a given environment. Therefore, any promising results in this area could also be useful to explore on the UrbanSound8K dataset.

7 References

Aiello, L. M., Schifanella, R., Quercia, D., Aletta, F. 2016. *Chatty maps: constructing sound maps of urban areas from social media data*. Available from http://rsos.royalsocietypublishing.org/content/3/3/150690, doi: 10.1098/rsos.150690 [Accessed on 20th June 2018].

Bae, S. H., Choi, I. & Kim, N. S. 2016. *Acoustic Scene Classification using Parallel Combination of LSTM and CNN*. Proceedings of the Detection and Classification of Acoustic Scenes and Events (DCASE) workshop 2016. 3 September, Budapest, Hungary.

Brandl, F. and Holleis, A. SILENCE e-learning, available from http://www.silence-ip.org/site/ [Accessed 20 June 2018].

Chachada, S. and Kuo, J. C. C. 2013. *Environmental Sound Recognition: A Survey.* Proceedings from APSIPA Transactions on Signal and Information Processing. 3. 1-9. 10.1109/APSIPA.2013.6694338.

Choi, K., Fazekas, G. and Sandler, M and Cho, K. 2018. *A Comparison of Audio Signal PreProcessing Methods for Deep Neural Networks on Music Tagging.* Available from https://arxiv.org/abs/1709.01922 [Accessed 22nd June 2018].

Choi, K., Fazekas, G. and Sandler, M and Cho, K. 2018. *A Tutorial on Deep Learning for Music Information Retrieval.* Available from https://arxiv.org/pdf/1709.04396.pdf. [Accessed 25th June 2018].

Choi, K., Fazekas, G. and Sandler, M. 2016. *Automatic Tagging using Deep Convolutional Neural Networks*. Proceedings of the 17th International Society for Music Information Retrieval Conference IMSIR, New York City, USA, August 7-11, 2016. Available from https://wp.nyu.edu/ismir2016/wp-content/uploads/sites/2294/2016/07/009_Paper.pdf. [Accessed 29th June 2018].

Choi, K., Fazekas, G. and Sandler, M. 2016. *Explaining Deep Convolutional Neural Networks on Music Classification*. Available from https://arxiv.org/abs/1607.02444. [Accessed 23rd June 2018].

Daisuke, N. 2014*. ml-urban-sound: UrbanSound classification attempts*, available from https://github.com/daisukelab/ml-urban-sound [Accessed 16th July 2018].

Dieleman, S., Shrauwen, B. 2013. *Multiscale approaches to music audio feature learning*. Proceedings of 14th International Society for Music Information Retrieval Conference (ISMIR), Curitiba, Brazil. Available from http://ismir2013.ismir.net/wp-content/uploads/2013/09/69_Paper.pdf [Accessed 8th July 2018].

Dixon, S., Gouyon, F., Widmer, G. 2004. *Toward Characterisation of Music via Rhythmic Patterns*. In proceedings of the ISMIR International Conference on Music Information Retrieval, Universitat Pompeu Fabra, Barcelona, Spain.

Gencoglu, O., Virtanen, T. and Huttunen, H. 2014. *Recognition of Acoustic Events Using Deep Neural Networks.* Proceedings from the 2014 European Signal Processing Conference (EUSIPCO). Available from http://www.cs.tut.fi/sgn/arg/music/tuomasv/dnn_eusipco2014.pdf [Accessed 13th July 2018].

Goodfellow, I. J., Warde-Farley, D., Lamblin, P., Dumoulin, V., Mirza, M., Pascanu, R., Bergstra, J., Bastien, F., Bengio, Y. 2013. *Pylearn2: a machine learning research library.* Available from https://arxiv.org/abs/1308.4214. [Accessed on 19th July 2018].

Gouyon, F., Dixon, S., Pampalk, E. and Widmer, G. 2004. *Evaluating rhythmic descriptors for musical genre classification*. Proceedings of the 25th AES International Conference, pages 196-204.

Hettiola, T n.d., *Datasets, Environmental sounds*, available from www.cs.tut.fi/~heittolt/datasets.html [Accessed 24th June 2018]

Humphrey, E. J., Bello, J. P. and LeCun, Y. J. 2013. *Feature Learning and deep architectures: new directions for music informatics.* Journal of Intelligent Information Systems, 41(3): 461-481, 2013. https://doi.org/10.1007/s10844-013-0248-5.

Jansson, A., Humphrey, E., Montecchio, N., Bittner, R., Kumar, A. and Weyde, T. 2017. *Singing voice separation with Deep U-Net convolutional networks*. Proceedings of the 18th International Society for Music Information Retrieval Conference. Suzhou, China.

Khunarsal, P., Lursinsap, C. and Raicharoen, T. 2013. *Very short time environmental sound classification based on spectrogram matching*. Information Sciences 243 (2013) 57-74.

Kons, Z. and Toledo-Ronen, O. 2014. *Audio Event Classification Using Deep Neural Networks.* Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH. 1482-1486.

Lidy, T. 2015. *Spectral Convolutional Neural Network for Music Classification*. Paper submitted to Music Information Retrieval Evaluation eXchange (MIREX), Malaga, Spain. 2015.

Lidy, T. and Schindler, A. 2016*. Parallel Convolutional Neural Networks for Music Genre and Mood Classification Parallel Convolutional Neural Networks for Music Genre and Mood Classification*. Paper submitted to the Music Information Retrieval Evaluation eXchange (MIREX), 2016. Available from http://www.ifs.tuwien.ac.at/~schindler/pubs/MIREX2016.pdf. [Accessed on 23rd June 2018].

Lidy, T and Schindler, A. 2016. *CQT-Based Convolutional Neural Networks for Audio Scene Classification and Domestic Audio Tagging.* Proceedings of the Detection and Classification of Acoustic Scenes and Events (DCASE) 2016 workshop. 3 September, Budapest, Hungary.

Maas, A. L., Hannun, A. Y. and Ng, A. Y. 2013. *Rectifier Nonlinearities improve neural network acoustic models*. ICML 2013. Vol 28.

McFee, B., Raffel, C., Liang, D., Ellis, D. P. W., McVicar, M., Battenberg, E., Nieto, O. 2015. *Librosa: Audio and Music Signal Analysis in Python*, available from http://conference.scipy.org/proceedings/scipy2015/pdfs/brian_mcfee.pdf [Accessed on 24th June 2018].

Phan, H., Hertel, L., Maass, M. and Mertins, A. 2016. *Robust Audio Event Recognition with 1-Max Pooling Convolutional Neural Networks.* Available from https://arxiv.org/pdf/1604.06338.pdf [Accessed on 25th June 2018].

Piczak, K. J. 2015. *Environmental Sound Classification with Convolutional Neural Networks*. IEEE International Workshop on Machine Learning for Signal Processing, Sept 17-20, 2015. Boston, USA.

Piczak, K. J. 2015. *ESC: Dataset for Environmental Sound Classification*. In proceedings of the 23rd ACM international conference on Multimedia, pages 1015-1018, available from http://dx.doi.org/10.1145/2733373.2806390 [Accessed on 20th June 2018].

Pons, J., Lidy, T. and Serra, X. 2016. *Experimenting with Musically Motivated Convolutional Neural Networks, paper presented at the* 14th International Workshop on Content-Based Multimedia Indexing (CBMI), 2016, doi: 10.1109/CBMI.2016.7500246.

Pons, J. and Serra, X. 2017. *Designing Efficient Architectures for Modeling Temporal Features with Convolutional Neural Networks.* In proceedings of the 42nd International Conference on Acoustics, Speech, and Signal Processing (ICASSP), doi.org/10.1109/ICASSP.2017.7952601

Salamon, J and Bello, J. P. 2015. *Unsupervised Feature Learning For Urban Sound Classification*, in proceedings of the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 19-24 April, doi 10.1109/ICASSP.2015.7177954

Salamon, J and Bello, J. P. 2016. *Deep Convolutional Neural Networks and Data Augmentation for Environmental Sound Classification*. IEEE Signal Processing Letters, available from arXiv:1608.04363, doi 10.1109/LSP.2017.2657381 {Accessed 23rd June 2018].

Salamon, J., Jacoby, C and Bello, J. P. 2014. *A Dataset and Taxonomy for Urban Sound Research.* In proceedings of the 22nd ACM International Conference on Multimedia, Orlando USA, Nov. 2014.

Tokozume, Y., Ushiku, Y., Harada, T. 2018. *Learning from Between-class Examples for Deep Sound Recognition*. Available from arXiv:1711.10282 [Accessed 23rd June].

Uchida, Y. 2017. *An implementation of "mixup: Beyond Emperical Risk Minimization"*. Mixup-Generator. Available from https://github.com/yu4u/mixup-generator [Accessed 16th July 2018].

Van den Oord, A., Dieleman, S. and Schrauwen, B. 2013. *Deep content-based music recommendation*. In proceedings from Advances in Neural Information Processing Systems (NIPS) 2013. Available from https://papers.nips.cc/paper/5004-deep-content-based-music-recommendation. [Accessed 6th July 2018].

Zeiler, M. D. and Fergus, R. 2014. *Visualizing and understanding convolutional networks.* Computer Vision (ECCV), pages 818-833. Springer, 2014.

Zhu, X., Kaznady, M., Hendry, G. 2018. *https://github.com/Azure/DataScienceVM/tree/master/Tutorials/DeepLearningForAudio. Available from https://github.com/Azure/DataScienceVM/tree/master/Tutorials/DeepLearningForAudio [Accessed 21st June 2018].*

Appendix A Submitted Project Plan

Singing Voice Separation with Deep Capsule Network

Joy Basford INM363/INM373 Project Proposal April 2018

Introduction: what will be produced, why, where and when: This project aims to investigate the effect of capsule network methods in the task of source separation in music spectrograms. Capsules have some advantages over convolutional neural networks which them suitable candidates for difficult image classification tasks, such as spectrogram source separation. Many approaches employing deep convolutional networks have had successful results including Jansson et al. (2017) [1]. Motivated by these findings, this dissertation is driven by the research question of whether the introduction of capsules can improve the performance of a deep convolutional network in music source separation.

Responding to this question, this project will produce a capsule network trained and tested on the MUSDB18 dataset [2]. Developments in source separation methods are relevant to several beneficiaries including music professionals and researchers in Music Information Retrieval. This proposal will outline the details of the project as follows: critical context, suggested approach with details of the experiment design, analysis and evaluation. Any dependencies and risks identified at this stage will also be highlighted and addressed. Moreover, a timeline with specified milestones will be included to present the sequence and duration in which this work will be carried out.

Critical Context: ideas are informed by relevant information:

Singing Voice Separation: Source Separation is the isolation of component signals from a combined signal. There have been many approaches to this task, although Liu and Li observe that the problem has yet to be conclusively resolved [3]. These can be categorized into two categories: one where auditory events are grouped into streams according to psycho-acoustical cues and the other which relies on the mixing structure. For our purposes, we are concerned with the latter, which can be done in the spectral domain. Traditional methods include Non-negative matrix factorization [5] and Bayesian methods [6] whereas more recently, deep learning [7] and convolutional encoder-decoder architectures [8] have been employed to great effect. Separating sources requires estimating parts in isolation and one of the existing challenges is in obtaining a large dataset containing songs together with parts in isolation. The largest that I am aware of was created by Jansson et al [1] containing 20,000 tracks.

Convolutional Neural Networks (CNNs): Given the datasets used, we can approach the research question as an image classification task, for which a deep convolutional network would be well suited to. Since AlexNet outperformed all previous image classification methods [9], CNNs have become widely adopted as the benchmark in computer vision tasks but they are less apt at determining the spatial relationships among features (including perspective, size and orientation) than the presence of features. This limitation is shown in Figure 1. where two images contain facial components but only the left image represents a fact. CNNs struggle with this due to the process used in passing information from one layer to another called max pooling. This process selects the maximum activation in any region

within a grid of pixels and discards all other information, including spatial relationships (Figure 2).

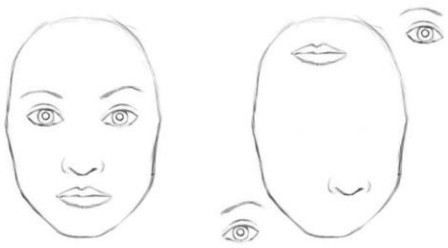


Figure 1. CNNs would recognise both images as faces, having detected the presence of individual facial features [10].

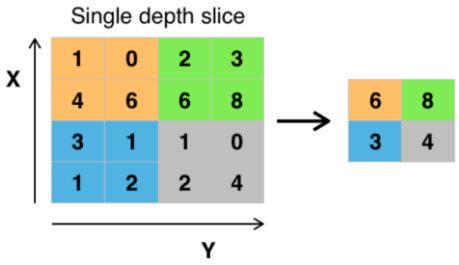


Figure 2. Max pooling selects the maximum activation in any region, losing surrounding information regarding properties [11].

This limitation is particularly problematic in the context of spectrogram data, which requires a high level of detail to be preserved in its reproduction. Even a minor shift in the frequency or time dimensions can lead to a significant impact in the overall perception. To address this issue, CNNS must either account for variants of the features with different properties, which increases the feature grid exponentially with the dimensions, or use a labelled training set that is exponentially larger given the additional dimensions.

Capsules: Capsules are a group of neurons, which produce a vector output rather than a scalar. This permits more information about the input data to be retained and processed. Applying a transformation matrix (W) produces a weighted sum of the input capsule vectors, forming a vote. This enables capsules to be grouped by similar votes and, by agreement, to route the predictions to the most similar parent capsule. Instead of Max Pooling, capsules transfer information from layer to layer using an approach called routing-by-agreement, Sabour, Frosst & Hinton propose two forms of routing mechanism: dynamic routing [12] and EM routing. Since this project is concerned with 2D spectrogram data, dynamic routing appears to be the most suitable routing mechanism. In the example below, capsule i provides the input vector u_i to capsule j and the transformation matrix W_{ij} produces a weighted sum s_j using coupling coefficients, c_{ij}. Capsule i makes multiple likelihood and property predictions. Agreement between multiple predictions in capsule i activates the corresponding capsule v. Through dynamic routing, the capsule sends its output to the capsule one level up, which is most similar to the weighted sum of its predictions. Both dynamic routing and EM routing overcome the limitation in max pooling by preserving more information in the transfer from one layer to another.

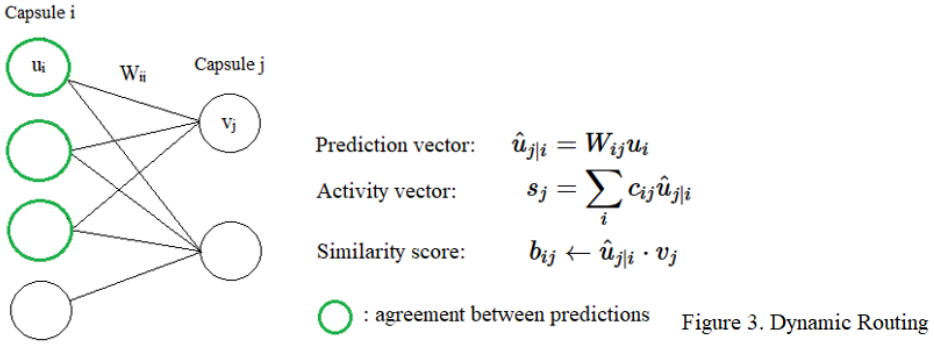


Figure 3. Dynamic Routing

Approaches: Methodology, Experiment Design, Analysis and Evaluation

Literature Review: Given the difference in architecture and routing mechanism, it appears that capsule networks would improve upon the performance of CNNs by retaining and processing more information without incurring exponential computational costs. Subsequently, it would be useful to test and compare these architectures across different datasets. There are still few applications of capsule networks and none so far seen in music source separation. One of the priorities in the literature review therefore is to examine the performance of CNNs to become familiar with their capabilities and constraints as a benchmark. Prioritizing reviews and surveys such as [3] to ensure that key methods are considered and observe best practice. Moreover, it is useful to examine competitions where novel approaches in source separation are introduced, for example the MIREX Singing Voice Separation competition where CNNs were among the best models in 2016 competition [7]. Given the diverse methods for source separation, a standardized evaluation format is desirable. A key contributor in this area is the Signal Separation Evaluation Campaign (SiSEC) [15], which aims to provide data for developing and testing new methods and release regular reports on progress in this space.

Data Gathering and Processing: This project will use the MUSDB18 dataset [2] which consists of 150 full length music tracks, together with isolated components including vocals. The music files require converting into spectrogram format. Various approaches exist to accomplish this, including through the scipy package in python [16]. During the literature search stage, a search to identify appropriate tools for file conversion will be conducted, so that a shortlist of appropriate methods can be tested before selecting the final one for the experiment. While the dataset will be split into training and test set, a subset of the training data will be reserved for validation of parameters for final model selection. **Capsule Network and CNN Development:** The project requires a programming framework and network architecture on which to run the models. There are several possibilities for designing and implementing the capsule network. At this stage, Tensorflow in Python has been identified as a suitable framework given the weight of examples of deep neural networks that have been implemented with it, many of which are publicly available [17]. The final selection, however, will be informed by the literature review and results of initial tests of a shortlist of options.

Model Construction, Training and Parameterisation

The goal of the models is to predict the isolated vocal and instrumental sources from the combined spectrogram as the input. To discriminate between the two, we make the assumptions that the sources are sparse and that the time-frequency bins in the vocal and instrumental sources do not overlap by much. By assigning each time-frequency cell as either close to 1 or close to 0, according to the probability that it belongs to either the vocal or instrumental source, a soft mask can be multiplied element-wise to the mixed spectrogram to create the final estimate. The capsule network architecture will adhere to the structure proposed in [12], adapting it as appropriate to the dataset. At its simplest, there will be two convolutional layers and a fully connected layer, though additional layers will be experimented with during the construction and testing stages. The first convolutional layer converts pixel intensities into 1D inputs for the primary capsules. Thereafter, routing by agreement occurs between the primary capsules and subsequent capsules. As a regularization parameter and to ensure a high-level of reproducibility, we can add a function to minimize reconstruction loss. The capsule output is a predicted spectrogram, from which audio signal

can be generated by integrating the output magnitude with the phase of the original spectrum. The benchmark CNN will consist of convolutional and subsampling layers. Each unit of the convolutional layer is a feature map, which extracts information from a sub-region of the underlying image. Each feature map has a corresponding unit in the subsampling layer that receives and transforms inputs, although there are fewer compared to the convolutional layer. Bishop observes multiple pairs of convolutional and subsampling layers used in practice [18]. Once the basic working prototypes have been trained on a small subset of spectrogram data, the architecture can be optimized. According to Xia and Wang (1998) [19], the following should be observed in a neural network with satisfactory computational performance: global convergence should be guaranteed after weights are arbitrarily initiated, the network design should preferably contain no variable parameter and the equilibrium points should correspond to the exact or approximate solution. Regarding the first of these comments, backpropagation will be used to determine the transformation matrix W and for CNN training. The CNN requires an additional consideration to observe the shared weights in the feature maps [18]. In addition, the number of capsule layers is an important consideration and will be guided by the dataset requirements as well as results observed in the literature. Beyond this, the reconstruction loss would be the only variable parameter, the effect of which will be determined in initial tests before the final model is selected.

Analysis and Evaluation: The models will be evaluated for the following: Normalized Signal to Distortion Ratio (NSDR), Signal to Interference Ratio (SIR) and Signal to Artefact Ratio (SAR). These can be computed using the museval package in Python [19]. The capsule network model will be compared to the CNN model across these metrics. Given the difference in architectures, the computational cost and training time and these will also be measured for evaluation.

Risks specific risks and plans to deal with them: Figure 5 presents the risks associated, ranked by impact (likelihood and consequences) as suggested by Dawson (2009) [20]. For each task completed, the risk register will be updated to reflect this and to re-evaluate the remaining risks.

Risk	Likelihood (1-3)	Consequence (1-5)	Risk Impact (L X C)	Control
Difficulties implementing Tensorflow Programming framework	3	5	15	Practice implementation on a smaller scale project using the MNIST handwritten digits dataset.
Capsule or CNN architecture development exceeds time estimates	3	5	15	1. Implement with University servers 2. Budget for Amazon Web Services hire
Literature review identifies existing implementation of capsules in singing voice source separation	2	5	10	Research question will be altered to implement capsule network using alternative dataset identified in [2]

Music file conversion exceeds time estimate	2	5	10	Identify 3 conversion techniques with sample code and practice conversion ahead of requiring converted data
Loss of work (either data, code or project report)	2	5	10	Back-up copies on external hard drive in addition to online drives (Microsoft Onedrive and Dropbox)
Relevant papers overlooked	2	2	4	After the allocated literature review timeframe, continue monitoring for new papers periodically throughout the project

Figure 5. Risk register

Ethics Review Form: BSc, MSc and MA Projects

Computer Science Research Ethics Committee (CSREC)

Undergraduate and postgraduate students undertaking their final project in the Department of Computer Science are required to consider the ethics of their project work and to ensure that it complies with research ethics guidelines. In some cases, a project will need approval from an ethics committee before it can proceed. Usually, but not always, this will be because the student is involving other people ("participants") in the project.

In order to ensure that appropriate consideration is given to ethical issues, all students must complete this form and attach it to their project proposal document. There are two parts:

Part A: Ethics Checklist. All students must complete this part. The checklist identifies whether the project requires ethical approval and, if so, where to apply for approval.

Part B: Ethics Proportionate Review Form. Students who have answered "no" to questions 1 – 18 and "yes" to question 19 in the ethics checklist must complete this part. The project supervisor has delegated authority to provide approval in this case. The approval may be provisional: the student may need to seek additional approval from the supervisor as the project progresses.

A.1 If your answer to any of the following questions (1 – 3) is YES, you must apply to an appropriate external ethics committee for approval.		*Delete as appropriate*
1.	Does your project require approval from the National Research Ethics Service (NRES)? For example, because you are recruiting current NHS patients or staff? If you are unsure, please check at http://www.hra.nhs.uk/research-community/before-you-apply/determine-which-review-body-approvals-are-required/.	**No**
2.	Does your project involve participants who are covered by the Mental Capacity Act? If so, you will need approval from an external ethics committee such as NRES or the Social Care Research Ethics Committee http://www.scie.org.uk/research/ethics-committee/.	**No**

3.	Does your project involve participants who are currently under the auspices of the Criminal Justice System? For example, but not limited to, people on remand, prisoners and those on probation? If so, you will need approval from the ethics approval system of the National Offender Management Service.	**No**

A.2 If your answer to any of the following questions (4 – 11) is YES, you must apply to the City University Senate Research Ethics Committee (SREC) for approval (unless you are applying to an external ethics committee).		*Delete as appropriate*
4.	Does your project involve participants who are unable to give informed consent? For example, but not limited to, people who may have a degree of learning disability or mental health problem, that means they are unable to make an informed decision on their own behalf?	**No**
5.	Is there a risk that your project might lead to disclosures from participants concerning their involvement in illegal activities?	**No**
6.	Is there a risk that obscene and or illegal material may need to be accessed for your project (including online content and other material)?	**No**
7.	Does your project involve participants disclosing information about sensitive subjects? For example, but not limited to, health status, sexual behaviour, political behaviour, domestic violence.	**No**
8.	Does your project involve you travelling to another country outside of the UK, where the Foreign & Commonwealth Office has issued a travel warning? (See http://www.fco.gov.uk/en/)	**No**
9.	Does your project involve physically invasive or intrusive procedures? For example, these may include, but are not limited to, electrical stimulation, heat, cold or bruising.	**No**
10.	Does your project involve animals?	**No**
11.	Does your project involve the administration of drugs, placebos or other substances to study participants?	**No**

A.3 If your answer to any of the following questions (12 – 18) is YES, you must submit a full application to the Computer Science Research Ethics Committee (CSREC) for approval (unless you are applying to an external ethics committee or the Senate Research Ethics Committee). Your application may be referred to the Senate Research Ethics Committee.	*Delete as appropriate*

12.	Does your project involve participants who are under the age of 18?	**No**
13.	Does your project involve adults who are vulnerable because of their social, psychological or medical circumstances (vulnerable adults)? This includes adults with cognitive and / or learning disabilities, adults with physical disabilities and older people.	**No**
14.	Does your project involve participants who are recruited because they are staff or students of City University London? For example, students studying on a specific course or module. (If yes, approval is also required from the Head of Department or Programme Director.)	**No**
15.	Does your project involve intentional deception of participants?	**No**
16.	Does your project involve participants taking part without their informed consent?	**No**
17.	Does your project pose a risk to participants or other individuals greater than that in normal working life?	**No**
18.	Does your project pose a risk to you, the researcher, greater than that in normal working life?	**No**

A.4 If your answer to the following question (19) is YES and your answer to all questions 1 – 18 is NO, you must complete part B of this form.		
19.	Does your project involve human participants or their identifiable personal data? For example, as interviewees, respondents to a survey or participants in testing.	**No**

References

[1] Jansson, A., Humphrey, E., Montecchio, N., Bittner, R., Kumar, A. and Weyde, T. (2017). Singing voice separation with Deep U-Net convolutional networks. Proceedings of the 18th International Society for Music Information Retrieval Conference. Suzhou, China.

[2] Rafii, Z., Liutkus, A., Stöter, F-R., Mimilakis, S. I. and Bittner, R. (2017). The MUSDB18 corpus for music separation. Doi: 10.5281/zenodo.1117372. Available at: https://doi.org/10.5281/zenodo.1117372 [Accessed 16 April 2018].

[3] *Liu, R.and Li S. (2009) A Review on Music Source Separation.2009. Proceedings of IEEE Youth Conference on Information, Computing and Telecommunication. Beijing, China. Doi:10.1109/YCICT.2009.5382353*

[4] Virtanen, T. (2007) Monaural Sound Source Separation by Nonnegative Matrix Factorization with Temporal Continuity and Sparseness Criteria. IEEE Transactions on Audio, Speech and Language Processing. 15(3), 1066-1074. Doi:10.1109/TASL.2006.885253

[5] Chien, JT. & Hsieh, H-L. (2013) Bayesian group sparse learning for music source separation. *EURASIP Journal on Audio, Speech, and Music Processin*g. Springer International Publishing. Available at https://doi.org/10.1186/1687-4722-2013-18

[6] Cobos, M. and López, J. J. (2008) Stereo audio source separation based on time–frequency masking and multilevel thresholding. Digital Signal Processing. 18 (6) 960-976. Available at https://doi.org/10.1016/j.dsp.2008.06.004. [Accessed 16 April 2018].

[7] Chandna, P., Janer, J., Miron, M., Gómez , E. (2017) Monoaural Audio Source Separation Using Deep Convolutional Neural Networks. Proceedings from the13th International Conference on Latent Variable Analysis and Signal Separation (LVA ICA2017). Grenoble, France.

[8] Grais, E. M and Plumbley, M. D () Single Channel Audio Source Separation using Convolutional Denoising Autoencoders arXiv:1703.08019

[9] Krizhevsky, A., Sutskever, I., Hinton, G. E. (2017) ImageNet Classification with Deep Convolutional Neural Networks. ACM Digital Library. 60(6) 84-90. Doi: 10:1145/3065386 [Accessed on 16 April 2018]

[10] [11] Garg, S. (2017) *Demistifying "Matrix Capsules with EM Routing." Part 1: Overview.* Available at https://towardsdatascience.com/dymistifying-matrix-capsules-with-em-routing-par-1-overview-2026133a8457 [Accessed on 16 April 2018]

[12] Sabour, S., Frosst, N. and Hinton, G. E. (2017) Dynamic Routing Between Capsules. arXiv:1710.09829 [cs.CV]

[13] [14] Hui, J. (2017) Understanding Dynamic Routing between Capsules (Capsule Networks). Available at https://jhui.github.io/2017/11/03/Dynamic-Routing-Between-Capsules/ [Accessed on 16 April 2018]

[15] *Signal Separation Evaluation Campaign 2018.* Available from https://sisec.inria.fr/ [Accessed on 16 April 2018]

[16] Scipy Signal Spectrogram. Available at https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.spectrogram.html [Accessed on 16 April 2018]

[17] Zaccone, G. and Karim, M. R. (2018) Deep Learning with TensorFlow: Explore neural networks and build intelligent systems with Python. 2nd Edition. Packt Publishing Limited.

[18] Bishop, C. M. (2006) Pattern Recognition and Machine Learning. Springer.

[19] Xia, Y. and Wang, J. (1998) A General Methodology for Designing Globally Convergent Optimization Neural Networks. IEEE Transactions on Neural Networks. 9(6) 1331-1343

[20] Museval evaluation toolkit for SiSec 2018 Music Task. Available at https://github.com/sigsep/sigsep-mus-eval. [Accessed on 16 April 2018]

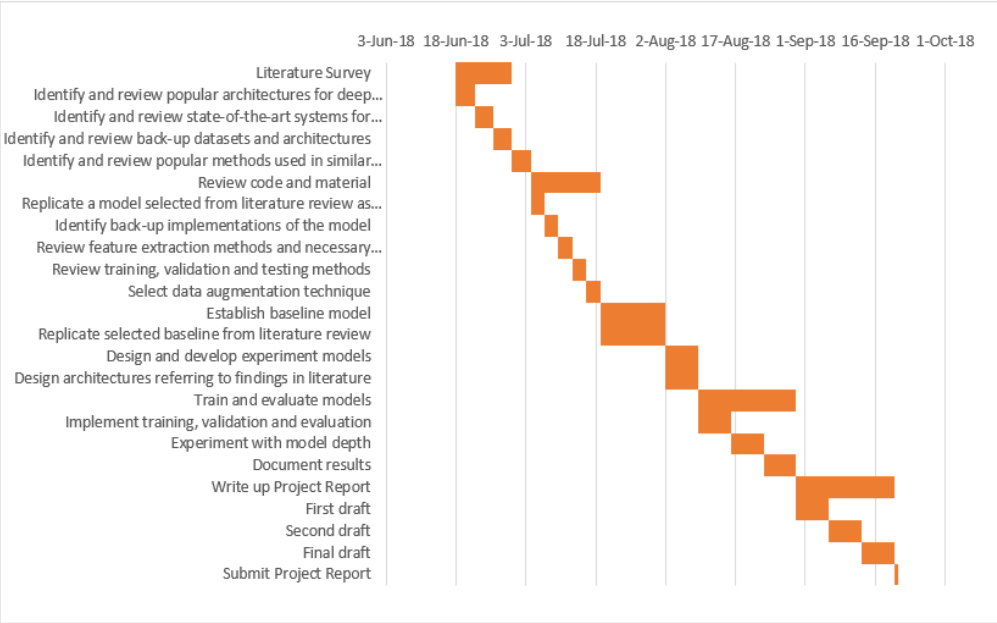
[21] Dawson, C. W. (2015) Projects in Computing and Information Systems. 3rd Edition. Pearson Education

Appendix B Risk Register

Risk	Likelihood (1-3)	Consequence (1-5)	Risk Impact (L x C)	Control
Difficulty replicating baseline model selected from literature review	2	5	**10**	Establish a new baseline, replicating the selected model as closely as possible
Difficulty building neural network models	3	4	**12**	Use examples from documentation and similar implementations
Difficulty implementing experiments in deep learning frameworks	2	3	**6**	Select a widely used framework and become familiar with it
Model training, validation and testing is slow on local machine	3	5	**15**	Learn to use and share access to university server with GPU
Loss of work (either data, code or report)	2	5	**10**	Back-up copies on online drive (Google Drive), in personal directory on university server and local machine
Relevant work in literature overlooked	2	2	**4**	Continue checking for newly published research periodically after the literature review

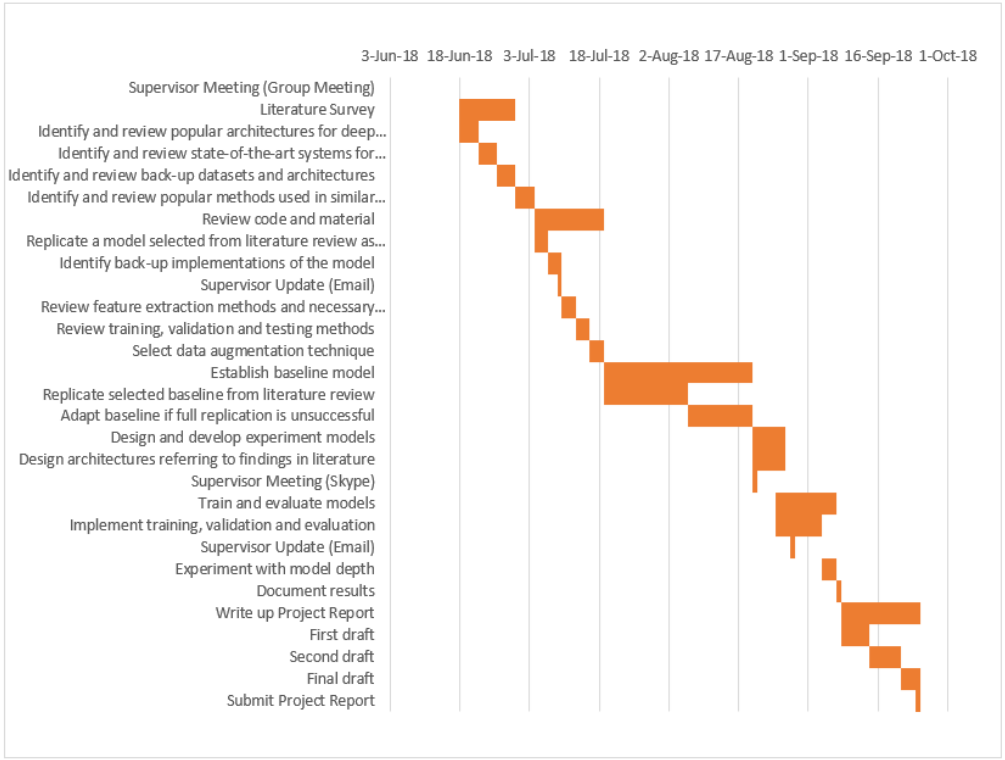
Appendix C Work Schedule

Task	Duration	Start Date
Literature Survey	**12**	**18-Jun-18**
Identify and review popular architectures for deep learning task	4	18-Jun-18
Identify and review state-of-the-art systems for selected task	4	22-Jun-18
Identify and review back-up datasets and architectures	4	26-Jun-18
Identify and review popular methods used in similar tasks	4	30-Jun-18
Review code and material	**15**	**4-Jul-18**
Replicate a model selected from literature review as baseline	3	4-Jul-18
Identify back-up implementations of the model	3	7-Jul-18
Review feature extraction methods and necessary libraries	3	10-Jul-18
Review training, validation and testing methods	3	13-Jul-18
Select data augmentation technique	3	16-Jul-18
Establish baseline model	**14**	**19-Jul-18**
Replicate selected baseline from literature review	14	19-Jul-18
Design and develop experiment models	**7**	**2-Aug-18**
Design architectures referring to findings in literature	7	2-Aug-18
Train and evaluate models	**21**	**9-Aug-18**
Implement training, validation and evaluation	7	9-Aug-18
Experiment with model depth	7	16-Aug-18
Document results	7	23-Aug-18
Write up Project Report	**21**	**30-Aug-18**
First draft	7	30-Aug-18
Second draft	7	6-Sep-18
Final draft	7	13-Sep-18
Submit Project Report	**1**	**20-Sep-18**



Appendix D Updated Work Schedule Reflecting Delays

Task	Duration	Start Date
Supervisor Meeting (Face to Face)	1	5/8/2018
Supervisor Meeting (Group Meeting)	1	31-May-18
Literature Survey	**12**	**18-Jun-18**
Identify and review popular architectures for deep learning task	4	18-Jun-18
Identify and review state-of-the-art systems for selected task	4	22-Jun-18
Identify and review back-up datasets and architectures	4	26-Jun-18
Identify and review popular methods used in similar tasks	4	30-Jun-18
Review code and material	**15**	**4-Jul-18**
Replicate a model selected from literature review as baseline	3	4-Jul-18
Identify back-up implementations of the model	3	7-Jul-18
Supervisor Update (Email)	1	9-Jul-18
Review feature extraction methods and necessary libraries	3	10-Jul-18
Review training, validation and testing methods	3	13-Jul-18
Select data augmentation technique	3	16-Jul-18
Establish baseline model	**32**	**19-Jul-18**
Replicate selected baseline from literature review	18	19-Jul-18
Adapt baseline if full replication is unsuccessful	14	6-Aug-18
Design and develop experiment models	**7**	**20-Aug-18**
Design architectures referring to findings in literature	7	20-Aug-18
Supervisor Meeting (Skype)	1	20-Aug-18
Train and evaluate models	**13**	**25-Aug-18**
Implement training, validation and evaluation	**10**	**25-Aug-18**
Supervisor Update (Email)	1	28-Aug-18
Experiment with model depth	3	4-Sep-18
Document results	1	7-Sep-18
Write up Project Report	**17**	**8-Sep-18**
First draft	6	8-Sep-18
Second draft	7	14-Sep-18
Final draft	4	21-Sep-18
Submit Project Report	**1**	**24-Sep-18**



Appendix E Parallel Pipelines in (Bae, Choi & Kim, 2016)