Spectrogram pre-processing for sound classification

Erik Nilsson,
Sebastian Sjögren

I. Abstract

Sound classification is being used every day for speech recognition, security and more. The goal of this project is to develop a model for sound classification, with a focus for researching different outcomes for different methods of audio processing. In particular our model will aim to classify atleast three different instrument sounds, and we will train two different models where one uses spectrograms and the other uses mel spectrograms to learn from.

The results show that we have created a model capable of accurate sound classification managing a peak of 90% accuracy rate, classifying three different instrument sounds. Furthermore, it was found that using mel spectrograms as a processing technique did not outperform using spectrograms when using the test set. However, only a limited amount of data and classes to classify was used to arrive at this conclusion. Since mel spectrograms did perform better then spectrograms when using a larger validation set, some future development is needed to arrive at any conclusive results in the form of more data being used and potentially expanding the model to classify more classes.

II. Introduction

From speech recognition to security and monitoring, automatic sound classification without the need for human input is used everyday all over the world. The goal of this project is to create a model for sound classification, specifically instrument sounds and to research which sound processing methods yields the best performance. We chose to work with sounds to explore methods of processing sound data into useful information for a model and how to optimize a model for this purpose. Methods for feature extraction from sound data that will be used are normal spectrograms and the mel spectrogram, and we strive to learn which method provides the best performance. The minimum target goal for performance to reach is for the model to accurately distinguish between two different sounds and correctly classify them.

A. Related works

There are several papers that have explored sound classification and sound data processing. A paper that explored sound classification and processing somewhat in depth is Rudberg [2022]. As for general sound classification these two papers were found: Das et al. [2020] and Massoudi et al. [2021]. A relevant paper that explored music genre classification but also was published in IITM Journal of Management and IT, Poonia et al. [2022] and another paper is Ghosh et al. [2023]. Knowledge gained from the methods explored in these papers can help when developing the model for our project as well as the papers citing other papers that can could be used if need be. Additionally these papers on the topic were found and deemed to be useful: Doshi [2021], Ketan [2021], Chaudhary [2021], Gartzman [2020]. Some general conclusions drawn from these papers is that a CNN architecture is a good architecture to use, although it may not always be the best. It also shows general methods for developing a model for sound classification.

III. Background

What distinguishes a model for sound classification from other machine learning models is the unique transformation which needs to be applied for sound data to be useful. Therefore this background section will only cover these methods.

To train a model for sound classification some method of data processing needs to be used to transform a given audio input into something usable by a machine learning model. Two common methods of doing this which have been explored in this project are the spectrogram and the mel spectrogram.

A. Spectrogram

A spectrogram is a visual representation of the spectrum of frequencies in a signal as they vary with time. It is a way to analyze and display the frequency content of a signal over time. A spectrogram typically displays the entire range of frequencies in a signal, using a linear frequency scale.

Spectrograms are created by applying a mathematical transformation, such as the Fast Fourier Transform (FFT), to a short segment of the signal at a time. This process decomposes the signal into its constituent frequency components, and the resulting data is then used to create the spectrogram.

B. mel spectrogram

A mel spectrogram uses a mel scale for the vertical axis. The mel scale is a nonlinear scale that models human auditory perception. The mel scale emphasizes lower frequencies more than higher frequencies because humans are more sensitive to changes in pitch at lower frequencies. Mel spectrograms are particularly useful in audio and speech processing, as they provide a representation that aligns more closely with how we hear and distinguish different frequencies.

To create a mel spectrogram, the signal is first divided into short time frames, and a Discrete Cosine Transform (DCT) is applied to the magnitude spectrum (computed via a Fourier transform) within each frame. The DCT coefficients are then mapped to the mel scale to create the mel spectrogram.

IV. Method

Creating a model for sound classification follow three main steps: Processing the data, constructing and training the model and then evaluating performance. After evaluation the model will be improved and adjusted according to the information gathered until a final evaluation is obtained which is deemed to be accurate enough for the purposes of this project.

A. Data processing

Data processing in the context of a sound classifier is going from raw sound data to something viable as input to a model. For our model, we used the following methods:

1) First we resample the audio data, ensuring that all data has the same sampling rate
2) We then use zero-padding or remove data to ensure all data is of the same length.
3) Lastly, we create mel spectrograms and spectrograms to represent the now processed data.

After these three steps we have transformed our raw audio data into something our model can use to train and predict.

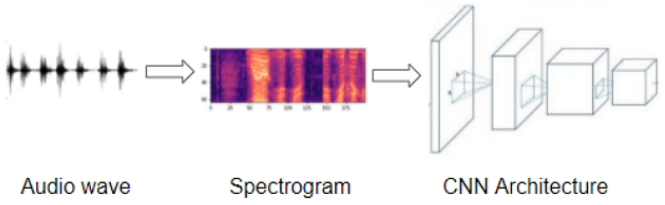


Fig. 1: Sound to Model

B. Training and model structure

The model architecture that was chosen for the sound classification part is a Convolutional Neural Network (CNN) and the quarry in designing the network was to make the model extract patterns and features well from the spectrograms and learn them in order to then be able to classify a spectrogram to be one of three classes. The model comprises the following layers and functions (can also be seen in the attatchments [2]):

```
from tensorflow.keras.layers import LeakyReLU

modelLeaky = Sequential([
    Conv2D(32, (3, 3), input_shape=(128, chosen_frames, 1)),
    LeakyReLU(alpha=0.1),
    MaxPooling2D((2, 2)),
    Conv2D(64, (3, 3)),
    LeakyReLU(alpha=0.1),
    MaxPooling2D((2, 2)),
    Flatten(),
    Dense(128),
    LeakyReLU(alpha=0.1),
    Dropout(0.5),
    Dense(3, activation='softmax')
])

custom_learning_rate = 0.001

modelLeaky.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=custom_learning_rate),
                   loss='categorical_crossentropy',
                   metrics=['accuracy'])

modelLeaky.summary()
```

Fig. 2: The model's architecture

Input Layer: The input layer has a shape of (128, chosen frames, 1), which was chosen to match the shape of the spectrograms. The inputt shape has a "1" that indicates that we are dealing with grayscale images.

Convolutional Layers: Two convolutional layers are used in the model. The first layer has 32 filters with a 3x3 kernel size and the second convolutional layer has 64 filters and the same kernel size. Both of the convolutional layers are followed by a LeakyReLU activation function with an alpha parameter of 0,1, this is because we wanted to avoid the "dead relu" issue.

MaxPooling Layers: After each convolutional layer there is a MaxPooling2D layer with a 2x2 pool size to downsample the feature maps thus reducing the spatial dimensions while still preserving the most vital and important information. This was also done to help with overfitting and speeding up the training.

Flatten Layer: This layer flattens the feature maps into a 1D vector which is needed for the following fully connected layer "Dense" layers.

Dense Layers: Two dense layers are used. The first dense layer has 128 units and is followed by a LeakyReLU activation. A dropout layer with a rate of 0,5 is used. This is to help prevent overfitting where the number 0,5 represents the likelihood that a specific neuron in the network will be dropped out, not considered, which helps prevent overfitting and this is also just applied during training while during prediction all the neurons are used. The final dense layer consists of three units with the softmax activation function that outputs the probabilities for the three sound classes (Drum, Guitar, Piano).

Loss Function: The loss function used for the model is the categorical cross-entropy loss function which is often used and is suitable for multi-class classification tasks.

Optimizer: The optimizer used is the Adam optimizer with a learning rate of 0,001. This optimizer adapts the learning rate during trainingw which can help the model converge better.

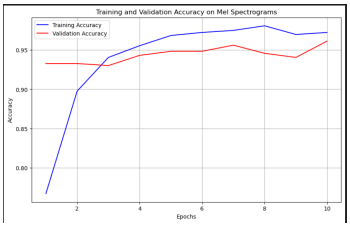
Metrics: The primary metric for monitoring the training progress is accuracy. The loss is also tracked to survey how the model is converging.

C. Evaluation

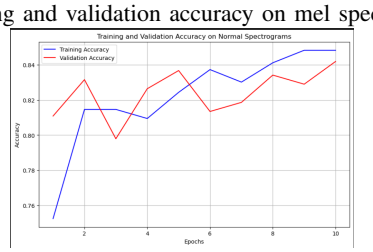
For evaluation the data will be split into a training set, a validation set and a test set according to 78/19/3 to fit the data given. We will then train the model and test for accuracy and produce accuracy plots and confusion matrices using the validation set to see which sounds may overlap with each other and which sounds are hard to classify to aid in the optimization process. Finally we will get a final accuracy plot and confusion matrix using the test set to evaluate performance.

V. Results

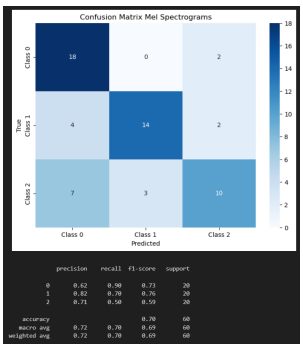
The model is trained over 10 epochs and then tested on the testing data which contains 20 samples of each class. After this a confusion matrix is made to see the performance of the model on the different classes and see how the classes were correctly or incorrectly classified. During training and testing the following observations were made:



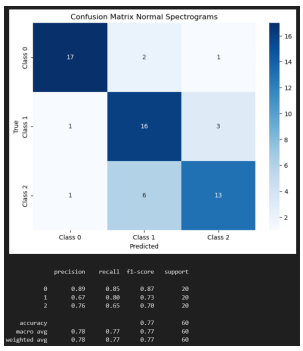
(a) Training and validation accuracy on mel spectrograms



(b) Training and validation accuracy on normal spectrograms



(a) Confusion matrix for mel spectrograms



(b) Confusion matrix for normal spectrograms

A. *Mel spectrograms*

For the mel spectrograms we see in the accuracy plot [3a] that the model's accuracy improves greatly during the first four to five epochs whereby it subsequently starts to taper off but still generally hovers around the same value of 97%.

The validation accuracy starts off quite well and increases very slowly during the epochs for a final value of around 94%. This indicates that the model is learning to classify the sounds from the mel spectrograms efficiently and the validation accuracy tells us that it has good generalization.

The testing results, that can be seen in the confusion matrix [4a], show the model's performance on the test dataset that is data that the model has not seen before. Provided are also the precision, recall, F1 scores, and support(number of samples in the testdata) for each class. The model's overall accuracy on the test data is 70%, and it demonstrates varying performance across the three sound classes.

The precision basically tells us how low the rate of false positives is (how many of the ones predicted as class x were actually of that class) and the recall how low the rate of false negatives is (how many of the actuall class x were actually predicted as that class). Finally the F1-score tells us the harmonic mean of the precision and recall and basically gives us a score that takes both of them into account.

Class 0: The model demonstrates okay precision 0,62 and great recall 0,9 resulting in an F1-score of 0,73.

Class 1: Precision 0,82 and recall 0,7 are both relatively high, resulting in an F1-score of 0,76.

Class 2: Precision 0,71 is relatively high and recall is a moderate 0,5, resulting in an F1-score of 0,59.

The macro and weighted averages are also provided, indicating overall performance by taking the averages for the precision, recall and f1-score for each class. The F1-score is 0,69 indicating an acceptable classification performance.

B. *Normal spectrograms*

For the normal spectrograms, the model's accuracy plot, shown in [3b], indicates a different learning pattern compared to the Mel spectrograms. In this case, the model's training accuracy improves consistently during the first two epochs but go down slightly until epoch four whereby it increases again and then tapers off around epoch eight and nine and reaches a final value of approximately 85%.

The validation accuracy plot also exhibits an upward trend, reaching a final value of around 84% but is very zigzaggy. This suggests that the model learns to classify the sounds from the normal spectrograms, and the relatively validation accuracy indicates good generalization capabilities although not as good as for the mel spectrograms.

The testing results as displayed in the confusion matrix [4b], showcase the model's performance on the unseen testing dataset. The model demonstrates an overall accuracy of 77% with precision, recall, and F1 scores provided, along with the support, for each class.

Class 0: The model shows good precision (0,89) and recall (0,85), resulting in a high F1-score of 0,87. Class 1: Okay precision (0,67) and high recall (0,80) leading to an F1-score of 0,73. Class 2: The model exhibits good precision (0,76) and okay recall (0,65), resulting in an F1-score of 0,70. The average F1-score is 0,77, showing the balance between precision and recall across all classes.

Succinctly, the model trained on normal spectrograms achieves an accuracy of 77% on the test data, with varying but generally balanced performance across the three sound classes. The F1-scores suggest an acceptable classification performance.

VI. Conclusions

The model trained on Mel spectrograms achieved high training accuracy and demonstrated good generalization, with a testing accuracy of 70%. While precision, recall, and F1 scores varied across classes but were relatively good, the average F1-score indicated acceptable classification performance.

The model trained on normal spectrograms exhibited fluctuating learning with training and validation accuracies significantly lower than with the mel spectrograms. It achieved a testing accuracy of 77% and balanced precision and recall and F1-scores for all classes.

The normal spectrogram model performed worse during training (both on training and validation data) but better during testing which could mean that it is better at classifying on unseen data than the one trained on mel spectrograms. However, the testing data was quite small of only twenty instances of each class as well as seeming to be a bit different than the data found in the training dataset (which was split into training and validation 80-20) if we compare the performance of the model on the validation data and the testing data. Thus we have a suspicion that the model trained on mel spectrograms might be better if we had a lot more data to test it on. It would also be good to have more training data.

Also notable is that originally in the dataset there was a fourth class, Violin, that we decided to exclude after we quite late into the project discovered that the dataset for this class had sound files that were not violins which made us have to exclude it as we did not have the time during this project to systematically go through the thousands of audiofiles in order to verify which are correct and which are not. Additionally we thus hope that there are not any other missclassified data in the dataset for the other classes that could impact the model's performance.

Generally the choice between Mel spectrograms and normal spectrograms may depend on the specific requirements of the application. Mel spectrograms offer an efficient way to capture spectral information in a format well-suited for neural networks, while normal spectrograms may provide reliable results with straightforward processing. The decision should consider trade-offs between computational efficiency, model performance, and the complexity of preprocessing.

Ultimately, this project demonstrates the efficacy of both preprocessing methods for sound classification where we judge the mel spectrograms to be somewhat better for the reasons mentioned above.

Future work could explore the addition of a lot more data for training and testing for clearer and more indicative results as well as investigating further prae-processing techniques and model architectures to achieve yet higher classification performance.

References

Kartik Chaudhary. Understanding audio data, fourier transform, fft, spectrogram and speech recognition, Jun 2021. URL https://towardsdatascience.com/understanding-audio-data-fourier-transform-fft-spectrogram-and-speech-recognition-a4072d228520.

Joy Das, Arka Ghosh, Abhijit Pal, Sumit Dutta, and Amitabha Chakrabarty. Urban sound classification using convolutional neural network and long short term memory based on multiple features. 11 2020. doi: 10.1109/ICDS50568.2020.9268723. URL https://www.researchgate.net/publication/346659500_Urban_Sound_Classification_Using_Convolutional_Neural_Network_and_Long_Short_Term_Memory_Based_on_Multiple_Features.

Ketan Doshi. Audio deep learning made simple: Sound classification, step-by-step, May 2021. URL https://towardsdatascience.com/audio-deep-learning-made-simple-sound-classification-step-by-step-cebc936bbe5.

Dalya Gartzman. Getting to know the mel spectrogram, May 2020. URL https://towardsdatascience.com/getting-to-know-the-mel-spectrogram-31bca3e2d9d0.

Partha Ghosh, Soham Mahapatra, Subhadeep Jana, and Ritesh Jha. A study on music genre classification using machine learning. *International Journal of Engineering Business and Social Science*, 1:308–320, 04 2023. doi: 10.58451/ijebss.v1i04.55. URL https://www.researchgate.net/publication/370546962_A_Study_on_Music_Genre_Classification_using_Machine_Learning.

Ketan. Audio deep learning made simple - data preparation and augmentation, Feb 2021. URL https://ketanhdoshi.github.io/Audio-Augment/.

Massoud Massoudi, Siddhant Verma, and Riddhima Jain. Urban sound classification using cnn. In *2021 6th International Conference on Inventive Computation Technologies (ICICT)*, pages 583–589, 2021. doi: 10.1109/ICICT50816.2021.9358621. URL https://ieeexplore.ieee.org/document/9358621.

Sahil Poonia, Chetan Verma, and Nikita Malik. Music genre classification using machine learning: A comparative study. 13:15–21, 08 2022. URL https://www.researchgate.net/publication/362619781_Music_Genre_Classification_using_Machine_Learning_A_Comparative_Study.

Olov Rudberg. Compare accuracy of alternative methods for sound classification on environmental sounds of similar characteristics. 2022. URL https://www.diva-portal.org/smash/get/diva2:1689775/FULLTEXT01.pdf.