Universiteit Leiden

Master Media Technology

Probing the World for Groove: A comparative study of Transfer Learning for Drum audio Style Classification

Name:	Trent Eriksen
Student ID:	s3746887
Date:	30/06/2025
1st supervisor:	Edwin van der Heide
2nd supervisor:	Dr. Robert Saunders

Master's Thesis in Media Technology

Leiden Institute of Advanced Computer Science
Leiden University
Einsteinweg 55
2333 CC Leiden
The Netherlands

Probing the World for Groove: A comparative study of Transfer Learning for Drum audio Style Classification

Trent Eriksen

s3746887

Thesis Advisors Edwin van der Heide and Robert Saunders

Graduation Thesis Media Technology MSc program, Leiden University

Date: 30/6/2025

1 Abstract

Percussion and Drums have been the subject of many works in the field of Music Information Retrieval (MIR) but research has trended in two key directions; (1) audio based single instrument classification, or (2) pattern recognition from symbolic representations such as standard notation and MIDI (Musical Instrument Digital Interface). This leaves a gap for exploration in the audio domain that interrogates drum styles which are composed of patterns over time. When CNNs (Convolutional Neural Networks) are applied to drum classification tasks they perform well but have not been extensively compared to pretrained transformer-based models for drum audio style classification in a supervised transfer learning context. This project seeks to examine classification of drum audio style by comparing a baseline CNN trained only on the GMD (Groove Midi Dataset) with a pretrained transformer based model; PaSST (Patchout Audio Spectrogram Transformer) with frozen general audio embeddings from AudioSet. Understanding this comparison reveals the ways in which general audio knowledge can effect drum audio style classification. Experiments with model depth, augmentation and padding show that PaSST with these frozen embeddings reduces performance in terms of

accuracy but reveals robust feature representation distinct from the baseline CNN.

Keywords

Machine Learning, Deep Learning, Music Information Retrieval, Transfer Learning, Supervised Learning

2 Table of Contents

Introduction, Related Work, Methodology, Experiments, Analysis, Discussion, Conclusion, References, Appendices

3 Introduction

Inspiration and Relevance

This project grew out of an inspiration to detect the motorik drum pattern; an iconic 4/4 time signature groove popularized by 1970's German rock, in musical recordings. Although this topic is uniquely compelling it presents several challenges. To execute a project with this aim one would need to create a custom dataset which would involve source separation to isolate the drum audio, beat detection algorithms to segment the audio and extensive annotation or utilization of few shot/zero shot learning to create ground truth labels. In order to avoid the task of dataset creation the GMD (Groove MIDI Dataset) was chosen as it is the largest drum audio dataset with primary and secondary style annotations and regularized segmentation allowing effective usage in this context (15). In addition the GMD was compiled from performances by professional drummers that included additional metadata for BPM (beats per minute), beat type (beat or fill), time signature among other data points relevant to audio and MIDI size. In terms of model architectures to handle this classification task, first a baseline CNN was chosen as it had been shown that this architecture type performed well on drum instrument classification (11). Then to explore the effect of transfer learning from general audio embeddings, PaSST was chosen as it is a lightweight

and efficient adaption of the AST (Audio Spectrogram Transformer) that has shown consistent performance improvement over baseline CNN models in a variety of downstream audio classification tasks (5). These two models set the groundwork for initial experimentation and data set exploration which led to the style scope of the project broadening to focus on the different performance of the models across all styles present in the GMD rather than just one subset of styles.

The Problem

Several musical audio and general audio classification tasks have involved a variety of CNN, RNN and transformer based architectures. Simple CNNs have shown to be generally performative for single instrument classification while transformer based approaches continue to improve results on a variety of musical audio classification tasks. For this project our musical audio classification task is drum audio style which will be defined as a set of patterns and musical articulations exemplified by combinations of onsets and timbral structures over time that are categorized to certain annotations determined by expert or professional drummers that contributed to the GMD. Often these styles share some combination of semantic labels from both music genre classification and drum pattern designations set out by musical cannon knowledge explicated by subject matter experts, in our case professional drummers. Because drum styles are linked to but independent from specific tempo or time signature constraints, a key challenge in this research area is capturing the features of these styles from variable-length audio (1). Recent applications in CNNs and transformer-based models (2; 3) show that a transfer learning approach, where a model trained on source task is then utilized on a target task like a smaller specialized dataset, offers increased classification accuracy in a variety of audio file lengths. For example Quelennec et al. evaluated a series of pretrained models across musical instrument, urban and environmental sounds noting differing performance in mean average precision and accuracy depending on the file length (4). In addition, Ding et al. compared various CNN baseline models with multiple pretrained models and displayed that the PaSST model showed significant improvement over the CNN baseline in both music instrument detection and genre classification (5). These studies make it compelling to explore a comparison between PaSST and a baseline CNN. To more clearly

explicate the context of this thesis the next section on related audio classification tasks will focus first on music then towards drums while providing additional foundations for transfer learning and augmentation.

4 Related Work

Polyphonic Audio Classification

Modern deep learning has driven substantial progress in classifying polyphonic music, typically focusing on tasks like genre recognition or music tagging over entire tracks. Earlier work has often employed CNNs that segmented audio into short frames, classified each segment, and aggregated predictions (6). A major shift included the use of pretrained audio representation models like VGGish trained on a large YouTube dataset (7). Another influential model, OpenL3, is a deep audio embedding learned via self-supervised audiovisual correspondence on a music subset of the large AudioSet dataset (8). In addition more recent research has pivoted toward AST's, which use self-attention to model long-range dependencies directly (9; 10).Although these classifiers can detect broad musical styles, they are not necessarily optimized for analyzing drum specific patterns in isolated form. Moreover, the focus in polyphonic classification is often on global attributes like timbral evolution over time in conjunction with melody, harmony and instrumentation, and not necessarily percussion or drum focused.

Single drum instrument audio

In contrast, another body of work focuses on classifying individual drum onsets or instruments where, the goal is to recognize whether a given audio excerpt corresponds to a snare, kick, cymbal, or other drum component (11). For single instrument detection it has been shown that CNNs outperform LSTMs in small-data regimes as these convolutional filters efficiently capture spectral features without the heavier parameter load of recurrent layers (12). While such single onset systems achieve high accuracy and highlight the timbral distinctions of individual drum sounds, they provide limited insight into how a sequence of

onsets forms a style over time. As a result these methods applied to single instrument detection can be investigated for application to drum audio style classification that involves multiple instruments in a drum kit creating patterns over variable temporal lengths.

Drum style classification

Capturing drum style which encompasses both rhythmic patterns and dynamic nuances requires a more temporally informed view than single hit detection. In the symbolic domain, ADT (Automatic Drum Transcription) has made strides in transcribing drum onsets into MIDI , but this relies on accurate MIDI file supplements that are usually not present in audio files unless explicitly provided with the audio data (13; 14). Converting raw audio to MIDI also introduces an intermediate representation that may lose timbral and performance details relevant to style and by contrast, directly classifying style from audio aims to retain these features. ADT is a way of detecting, classifying and notating a series of drum onsets without applying additional semantic labeling to this captured sequence. Drum audio style on the other hand takes a more global view of the given audio and learns the features in relation to its style label without transcribing individual onsets over a given time series. This makes it a higher level interpretation of a pattern as these semantic labels assigned represent a collective grouping of patterns that share underlying characteristics. One can view this as similar to the task of genre classification in music as genre is often defined by certain qualities that cannot necessarily be reduced to components of notation. Style classification for drum audio remains relatively unexplored in a comparative context with transfer learning.

Transfer Learning

In machine learning the process of transfer learning in a classification context is generally defined as the re-utilization of parameters trained on a source task to accomplish a target task (16). The motivation behind utilizing this technique is often around data scarcity as a network employing pretrained weights can often have its parameter size reduced. Although transfer learning is applied to many machine learning applications outside of music or audio, when applied to this domain the process of initializing models with weights usually involves

learned representations from large-scale audio corpora (17; 5).

For example, in Choi et al. they used a CNN based VGG (Visual Geometry Group) like architecture with mel spectrogram inputs which utilize the mel frequency scale to compute a high resolution image of the input audio, from the Million Song Dataset[1] consisting of a variety of musical audio to be used for target tasks which are downstream classification tasks like GTZAN[2] music genre classification in which they found that the pretrained models outperformed the random weight initialized baseline without pretraining (16). Although it can be said the source and target tasks here are quite similar, this study indicates that transfer learning could be successfully applied to a variety of data scarce MIR tasks. These tasks are similar in the sense that the pretrained data is of a similar nature to the data set which the model is being applied for in a classification context ie. using a model trained on musical audio for musical genre classification represents a likely similarity in underlying features. In our case we want to explore if general audio embeddings will translate to drum audio style classification representing less similarity between pretrained audio and target audio.

Pretrained embeddings like VGG or OpenL3 (Open-source deep audio and image embeddings) were once popular, but more recent architectures such as PaSST and PANNs (Pretrained Audio Neural Networks) have generally outperformed these older systems in downstream audio tasks (3; 4). Although this technique can be applied to both CNN and transformer based architectures it is noted that the current state of research tends to focus on utilizing transformers in a transfer learning context. For example, transformers pretrained on diverse data like AudioSet can capture both local and global audio patterns relevant to different sound events (2). Similarly, multi-modal models like CLAP (Contrastive Language Audio Pretraining) have demonstrated robust classification abilities in music datasets by learning audio representations aligned with text descriptions (18). Because of the application of this technique to genre classification, musical instrument classification and a variety of other downstream audio classification tasks it is compelling to explore the potential benefits in a drum audio style classification context.

[1] `http://millionsongdataset.com/`

[2] `https://www.kaggle.com/datasets/andradaolteanu/gtzan-dataset-music-genre-classification`

Application to our Problem

Drawing on both single instrument studies and broader transfer learning advances, this work is structured to investigate how well variations of the AudioSet pretrained PaSST model performs in classifying drum audio by style in relation to a baseline CNN trained on the GMD. This will elucidate if and and to what extent the pretrained embeddings offer an advantage over a baseline CNN baseline trained from scratch on GMD audio as has been shown in prior studies with data sets of similar nature and size (5). This is compelling because transferring knowledge from the large general audio corpora AudioSet to the GMD, highlights not only the efficacy of PaSST in this context but also has the potential to reveal if some form of drum audio style information is encoded in a 'world audio model'.

On Augmentation

Data augmentation in an MIR context generally refers to modifying the input audio waveforms, computed spectrograms or other representations with the intent to change the frequency content over time in the given input. Some applied choices are time-frequency masking, pitch shifting, time stretch or mixup which have been employed to improve model robustness and generalization in audio tasks (19). Adding these perturbations to the audio or the spectrogram allows for the model to learn from more varied conditions although results are varied in terms of classification accuracy ranging from modest improvements of 1 to 3 percentage points (20). Augmentations in the context of pretrained models compared to trained from scratch baselines show that they can be beneficial in data scarce environments in both contexts (26).

5 Research Question

What is the effect of pretrained general audio embeddings in PaSST when compared to a baseline CNN for drum audio style classification?

To address this, we investigate two main model architectures: A baseline CNN trained solely on the GMD and PaSST with frozen embeddings pretrained on AudioSet. We specif-

ically aim to primarily investigate if the AudioSet embeddings are beneficial for drum audio style classification in terms of accuracy and underlying feature representation.

6 Methodology

Data and Preprocessing Overview

The GMD[3] contains over 18,000 indexed audio files with extensive metadata that are segmented based on the MIDI notation giving a uniform musical length of two measures irrespective of BPM recorded and labeled by expert human drummers. This musically informed regularization of two measures is valuable as it allows for elucidation of whether this relatively short amount of musical audio is sufficient for classification across different absolute time durations. The style annotations include 'primary-style' and/or 'secondary-style' which is concatenated into a single tag to mitigate the effect of half of the dataset lacking secondary style annotations which eliminates the need for multi-label classification while still retaining the ability to analyze at a hierarchical level. Due to the fact these labels are human created there is some potential for injection of subjectivity here but it is reasonable to suppose that human drumming professionals are a plausible source for ground truth labeling as they are domain experts. The drum audio has a sample rate of 16kHz, which means it does not include frequencies above 8kHz; however, this reduced bandwidth is not a significant limitation in this context.

Multiple data configurations were explored to evaluate performance across different model architectures. Based on the results, only audio files under 10 seconds were used, as they comprised the majority of the GMD and allowed for a fair comparison between models, given that PaSST was pretrained on 10-second clips from AudioSet. To facilitate experimentation, analysis, and exploration of the data several modifications to the default GMD metadata were made including; revising the splits so that proportional class representation was present in all splits, audio file renaming to efficiently incorporate the existing indexing, file path columns, primary and secondary style concatenation to a new style-class column and a

[3] `https://www.tensorflow.org/datasets/catalog/groove#groove2bar-16000hz`

length-bin column to examine the duration distribution (1-3 seconds, 3-5 seconds, 5-7 seconds and 7-10 seconds). The waveforms of the selected audio recordings were transformed into log-Mel spectrograms as this time-frequency representation has proved effective for audio classification tasks (26). Input regularization was applied during preprocessing for both architectures by padding each input to a fixed length representing 10 seconds of audio initially using zero-padding, with alternative padding methods introduced in later experiments.

In this project, augmentation methods were not initially utilized, however for the sake of interpretability of the latent representations Timestrech, Time Patchout (from PaSST), Gaussian Noise and Room Simulation were implemented to observe if these produced movement of the embeddings. These augmentations were chosen in part for their feasibility and because of the variable length clips in the GMD. The python library audiomentations[4] is a go-to tool for many audio machine learning researchers which offers a wide variety of augmentations aimed at making models more useful for real world applications. By choosing augmentations that can be implemented during training rather than through re-rendering audio files the scope of this part of the project is contained. For definitions, Gaussian Noise will add random noise to the audio waveform, Timestretch changes the speed of the audio waveform without changing the pitch, Time Patchout removes vertical patches from the spectrogram and Room Simulator creates the effect of a room on the audio waveform. Crucially, these augmentations are implemented either on the raw waveform or the computed spectrogram to explore how the models will deal with time, noise and reverberant conditions which can provide insight into performance in more real-world conditions.

CNN Baseline Model

1. **Architecture Overview**

   Initially Hiner's model for drum instrument classification that achieved 93.4 percent accuracy was adapted (11). Then an iterative process was undertaken to modify the architecture with several changes primarily focused on increasing the depth of the network then later with different augmentation and padding refinements.

[4] `https://github.com/iver56/audiomentations`

As detailed in the experiments, changes in the architecture involved primarily adding 2d convolutional layers, each followed by max pooling, dropout and global average pooling, implemented with PyTorch[5]. Then the network passes the inputs to two fully connected with a 512 input size after which follows the feedforward network with Rectified Linear Unit (ReLU activation) which introduces non-linearity and helps the network learn more complex relationships within the data. After this global pooling occurs along with flattening, RelU activation, and dropout concluding with the final dense layer which takes the 512-dimensional output and further transforms it into a final output that represents the predicted probabilities for each drum style class.

2. **Preprocessing and Feature Extraction**

   For feature extraction the original sample rate of 16 Khz is maintained and no resampling is performed with NFFT=1024, 128 mel bins and hop=NFFT/4. These settings are chosen because they performed well in the original code we adapted and provide a sufficient setup for the classification task. A preprocessing notebook was also utilized to create several dataset configurations with different columns added to facilitate experiments in the Google Colab virtual environment (28).

3. **Variable-Length Inputs and Training**

   To handle the varied audio file durations all files are initially padded with zeros up to 10 seconds which helps the CNN perform better and will match our comparison with PaSST. For training a 50 epoch maximum is set with early stopping patience of 10 epochs to exist within a limited compute budget while also being sufficient for a dataset of modest size like the GMD. During training the Adam optimizer with batch size 16 and learning rate of 1e-4 helped constrain the training time to 3-8 hours depending on GPU type utilization and after concluding training inference is ran on the test set and compute the precision, recall and F1 scores for the both the entire test split and per class which is then visualized with confusion matrices per experiment.

4. **Model Interpretability**

[5] `https://pytorch.org`

Once arriving at the optimal primary configuration of the CNN baseline starting at experiment round 6, t-distributed stochastic neighbor embedding (t-SNE) is employed to visualize and examine the clustering in the embedding space both by individual style class and primary/secondary styles. This technique is particularly useful here as it provides a concise dimensionality reduction of the higher dimensional embeddings allowing additional insights beyond accuracy metrics which becomes particularly insightful when analyzed in conjunction with the confusion matrices.

5. **Expected Results**

We expect the classification accuracy of the model to increase with the number of Conv2d layers but only up to a point that will need to be experimentally tested in conjunction with dataset size and hyperparameter tuning. The experiments will examine different model depths and augmentation then report results. In the t-SNE visualizations it is expected to see clustering reflective of the accuracy scores.

PaSST Pretrained Model

1. **Overview and Motivation**

To compare with the CNN baseline and observe the influence of transfer learning from large scale general audio data, we adopt PaSST as our pretrained model (1). PaSST is trained on AudioSet; one of the largest general audio datasets that includes among other categories, environmental sounds, speech audio, and musical sounds with a regularized 10 second duration and human created labels (21). PaSST is well suited to this project due to it's efficient adaptation of the AST architecture, implementation of time and frequency patchout along with disentangled positional encodings and reliable ease of implementation. Additionally PaSST is fairly well cited and utilized in audio classification research with well maintained code[6]including several different pretrained configurations that facilitate experimentation. In the second round of experiments with different configurations, it is determined that PaSST-L; a light version of PaSST with 7 instead of 12 attention blocks trains faster and achieves higher accuracy than

[6]`https://github.com/kkoutini/passt_hear21`

alternatives tested in this project. For these reasons we are steadfast in the judgment that PaSST is a reasonable architectural choice in which to explore the relationship between pretrained frozen general audio embeddings and drum audio style classification in the GMD.

2. **Model Architecture**

The PaSST architecture borrows its fundamental design from vision transformers, namely the DeiT and ViT upon which the AST was developed (9; 22; 23). In these architectures an image is typically split into 16x16 patches during training. For our purposes this image input is a log-mel spectrogram divided into a sequence of patches and mapped to embedding vectors. The self-attention blocks enable PaSST to compute global relationships among patches by allowing the model to capture both short-term and long-term temporal structures. During training each patch is flattened and passed through a linear projection to form a 768-dimensional embedding which is treated as a high-level feature vector for the audio (9). The sequence of patch embeddings is then augmented with positional encodings to inform the transformer of each patch's time and frequency location. Consequently, the disentangled encodings allow the model to adjust to inputs of varying lengths by cropping or extending the time encoding dimension without altering frequency positional parameters (1). Afterwards, this sequence including patches and prepended tokens is then fed through transformer encoder layers each comprising multi-head self-attention and feed-forward sub-layers. At the last stage PaSST prepends a learned classification token (CLS) embedding to the sequence that aggregates global context for classification. The output class prediction is produced from the transformer's output tokens via a multi layer perceptron MLP that is tuned for optimal depth experiment round 8.

3. **Preprocessing and Feature Extraction**

PaSST was trained on 32kHz sample rate audio therefore GMD audio is upsampled to match the length of the AudioSet embeddings avoiding potential errors that could arise from a sample rate mismatch. Following the original paper's STFT approach we use a 25 millisecond window with 10 ms hop and apply Mel filter bank to convert

linear frequency spectrum into 128-bin Mel-spectrogram. These settings match our preprocessing setup for the baseline CNN thus setting the stage for a fair comparison. We apply the same data filtering done with the CNN in which all files up to 10 seconds in duration are utilized with padding to the fixed 10 second length of 320000 samples at 32Khz.

4. **Patchout Regularization**

   The Patchout mechanism in PaSST comes in two forms, either Structured (PaSST-S) which removes entire horizontal frequency bins or time columns or Unstructured (PaSST-U) which randomly removes individual patches regardless of their time-frequency positions which mimics dropout-like randomness at patch level granularity. The pre-trained examples included in the PaSST github repository [7] demonstrate that structured patchout is the more effective choice in terms of attaining classification accuracy. From this it is reasonable to suggest that time patchout will influence PaSST to rely more on overall rhythmic patterns rather than local temporal positions thus capturing the features related to style better. It is expected that this can be an advantage of PaSST over the CNN and will be tested in the experiments section.

5. **Transfer Learning, MLP Classifier and Training Setup**

   In the PaSST repository the authors provide 14 different configurations with frozen embeddings in which the experiments of this project first use the default linear layer then a deeper MLP. To maintain scope on the research question we only utilize the frozen embeddings. For training we use the defaults within PaSST with batch size of 16, 4 workers, learning rate of 1e-4 with 50 epochs max with early stopping criteria of 10 epochs. The experiment notebooks each have detailed logs that provide data for training curve visualizations (28). These choices elicit a fair comparison with our CNN baseline, and prevent scope creep from extensive hyperparameter tuning.

6. **Model Interpretability**

   In the same fashion as applied to the CNN we utilize t-SNE to examine the clustering

[7] `https://github.com/kkoutini/passt_hear21`

in the embeddings. Based on the different architectures, augmentations and dataset configurations tested, the changes that produce embedding drift will become apparent. The outputs from t-SNE and confusion matrices will help build reasoning for how each architecture deals with the data.

7. **Expected Results**

   We hypothesize that because of the over 1 million general audio files in AudioSet, some information will be present in the frozen PaSST embeddings that will aid in classification accuracy and feature representation over a baseline CNN. Fortification of the MLP on top of these embeddings is expected to produce noticeable increases in accuracy. Besides accuracy a more robust shared feature space between classes is expected to be indicated by nearby clustering of similar classes.

Evaluation Protocol

For both model pipelines the primary metric we evaluate and discuss is F1 / Accuracy even though Precision, Recall, F1 , mAP (Mean Average Precision) and AUC are also computed. F1 is the most appropriate in a single label classification context as we have devised with the primary and secondary style tag concatenation. Confusion matrices will provide at-a-glance views of model performance.

Reproducibility

All notebooks for these experiments are made available in a google drive repository for examination and reproducibility if desired (28). This repository also includes audio files, metadata, model checkpoints, log files, embedding space visualizations and confusion matrices used for and produced by the experiments.

7 Experiments

Description and Summary

At the very beginning of exploratory prototyping the initial CNN prototype first used only files with exactly 4 seconds in length at 16Khz resulting in a fixed input size and a smaller subset of the data. After this was shown to work well with over 90 percent accuracy the same architecture is used for the subsequent experiments until model depth is explored. After adjusting the metadata to facilitate single label classification and avoid class imbalances across the 80/10/10 split it became more readily usable for this project. All models were trained on either A100 or L4 GPU's in a Google Colab virtual environment during the months of April and May of 2025 (28). A total of 34 experiments were conducted consisting of 11 rounds that are grouped into four categories: Configuration Exploration, Dataset Exploration , Model Depth and Patchout , Augmentations and Padding. Each of these four groups will begin with a table giving a concise view of the results followed by further details.

Configuration Exploration

Table 1: F1 scores by experiment round and model (highest per round in **bold**).

Round	ID	Model	F1	Notebook	Description
1	1.1	CNN	**0.9204**	GMD_CNN_prototype3.ipynb	Defaults with rock styles
1	1.2	PaSST	0.8544	PaSST_setup2.ipynb	Defaults with rock styles
2	2.1	PaSST	0.8660	PaSST_setup3.ipynb	PaSST config
2	2.2	PaSST	**0.8699**	PaSST_setup4.ipynb	PaSST config

1. **Experiments Round One - PaSSt and CNN default test with rock styles**

   The first round of experiments was aimed at utilizing a subset of the GMD that contained all 2 measure audio files with the style annotation rock in either the style-primary or style-secondary columns with the exception of the style-secondary entitled groove8 as this class had only a few files and because its style was given a generic name it would have little value for this initial experiment and was thus filtered out of the dataset. This experiment round utilizes 5082 of the original 18297 files. Following an 80 percent train, 10 percent validation, 10 percent test split,the train split contains 4062 samples,

validation 505 samples and test 515 samples all with roughly analogous proportional splits, for example all three splits contain 69.5 percent rock samples, 5.3 percent punk samples etc. on for each style-class in the dataset. This proportional data balancing is maintained throughout all experiments for consistency.

Hypothesis:

To properly establish baseline configurations for both model architectures we explore different data subsets and pretrained configurations with the expectation that the PaSST default pretrained model with the default linear layer head will result in higher classification accuracy of styles over the 3 layer CNN architecture because of the underlying AudioSet feature representation.

Experiment 1.1

In this experiment the model trained for the full 50 epochs achieving an F1 score of 0.9204 which is a good initial result for the CNN architecture. The confusion matrix shows the model mostly had issues with classifying secondary style tags within the primary style rock. This is not a particularly surprising result as the other styles can be seen as sub-styles of the rock class. Other than that a notable confusion is also between the labels rock-shuffle and rock-halftime.

Experiment 1.2

This experiment used the default pretrained model (arch="passt-s-swa-p16-128-ap476") which was noted by the authors to be the model that most efficiently converged, meaning it finishes training in fewer epochs (1). The options indicated in the architecture title refer to structured patchout (s) , Stochastic Weight Averaging (swa), Patch size = 16 (p16), Input frequency dimensions (128) and Average Precision .476 on Audioset (ap476). The model was set up to train for the full 50 epochs, but early stopping with patience of 10 was triggered, disabling further training beyond epoch 29. The model achieved an F1 score of 0.8544. The confusion matrix shows again the model mostly had issues with classifying rock secondary style tags with the main rock style class. Notably different from the first CNN experiment is that this version of PaSST confused classes punk, funk and rock shuffle with rock tracks substantially more.

Discussion:

This initial round of experiments indicates a potential to disprove the hypothesis but further experimentation will be implemented to expand upon these findings as the CNN baseline outperformed PaSST by 6.67 percent on the same dataset. From this stage it is useful to test with different pretrained configurations of PaSST provided and different dataset configurations.

2. **Experiments Round Two - PaSST pretrained configuration tests**

These experiments were set up to test which pretrained version of PaSST was most appropriate to use for additional experimentation. Two other configurations 'passt-s-kd-p16-128-ap486' and 'passt-l-kd-p16-128-ap47' are chosen amongst the available options because they either have the highest average precision on AudioSet (ap486) or have the highest average precision within a lighter transformer architecture (ap47). The aim of these experiments is to choose the optimal PaSST pretrained architecture to conduct future experiments with.

Hypothesis:

To determine which PaSST model to move forward with in subsequent experiment rounds, considerations of accuracy and training time will be tested. We expect 'passt-s-kd-p16-128-ap486' to outperform 'passt-s-swa-p16-128-ap476' due to higher average precision on Audioset and that 'passt-l-kd-p16-128-ap47' will outperform 'passt-s-swa-p16-128-ap476' in terms of training time because of its lighter architecture.

Experiment 2.1

In this experiment the PaSST configuration is changed to arch="passt-s-kd-p16-128-ap486" which has the highest Average Precision score on AudioSet. The options indicated in the architecture title refer to structured patchout (s) , knowledge distillation (kd), Patch size = 16 (p16), Input frequency dimensions (128) and Average Precision .486 on Audioset (ap486). The model was set up to train for the full 50 epochs, but early stopping triggered disabling further training beyond epoch 43. Upon inference the model achieved an F1 score of 0.8660 demonstrating a roughly 1.2 percent

improvement over the 'passt-s-swa-p16-128-ap476' architecture from the first PaSST experiment. The confusion matrix shows again the model mostly had issues with classifying rock secondary style tags with the main rock style class as is likely to occur due to the over-representation of the rock primary style annotation. When comparing the confusion matrix to the first PaSST experiment this architecture is notably better at disambiguating rock folk and rock rockabilly from rock. This leads us now to test the next pretrained configuration.

Experiment 2.2

In this experiment the pretrained configuration is set to arch="passt-l-kd-p16-128-ap47" with the 7 transformer attention blocks while still maintaining structured patchout with average precision of 0.4708 on AudioSet which is also referred to by the original authors as PaSST-L (light). The options indicated in the architecture title refer to light reduced depth=7 (l) , knowledge distillation (kd), Patch size = 16 (p16), Input frequency dimensions (128) and Average Precision .4708 on Audioset (ap47). The model trained faster than prior runs for the full 50 epochs with no triggering of early stopping. Upon inference, the model achieved an F1 score of 0.8699 demonstrating a 0.39 percent improvement over 'passt-s-kd-p16-128-ap486' from the second PaSST experiment and a 1.6 percent improvement over the initially used 'passt-s-swa-p16-128-ap476' configuration. The confusion matrix shows PaSST-L is marginally better at disambiguating punk rock, shuffle and rock-indie from rock while most other classes see little or no change.

Discussion:

These experiments demonstrate that PaSST-L provides not only the fastest training time but also the best F1 scores within this subset of the GMD. This means our hypotheses from this round of experiments were partially proven but exceeds expectations as PaSST-L not only trains faster but is also more accurate. From these results PaSST-L is chosen as the configuration for all future PaSST experiments.

Dataset Exploration

Table 2: F1 scores by experiment round and model (highest per round in **bold**).

Round	ID	Model	F1	Notebook	Description
3	3.1	CNN	0.3267	GMD_CNN_prototype4.ipynb	GMD-mini
3	3.2	PaSST	**0.3911**	PaSST_setup5.ipynb	GMD-mini
4	4.1	CNN	**0.7531**	GMD_CNN_prototype5.ipynb	GMD-full
4	4.2	PaSST	0.7367	PaSST_setup6.ipynb	GMD-full
5	5.1	PaSST	0.7269	PaSST_setup6_2.ipynb	GMD-full repeat
5	5.2	PaSST	0.7313	PaSST_setup6_3.ipynb	GMD-full repeat
5	5.3	CNN	0.7646	GMD_CNN_prototype5_2.ipynb	GMD-full repeat
5	5.4	CNN	**0.7827**	GMD_CNN_prototype5_3.ipynb	GMD-full repeat

3. **Experiments Round Three - 10 percent mini GMD tests**

With the PaSST-L architecture we further investigate how the baseline CNN and PaSST handle different subsets of the data. To do this we take the entire original dataset under 10 seconds with 18,264 files between all splits.Through analyzing the duration the files via the added length bin column, it is observed that there are no substantial length imbalances per split meaning that proportional balancing is not needed on duration in addition to class. To get a view of the entire dataset through a quicker training pass 10 percent (1870 files) proxy of the full dataset based is used. Experiments in this category involve 74 output classes as opposed to the 9 output classes in prior experiments.

Hypothesis:

To guide further experimentation with the full dataset, exploration within a smaller representative sample has the potential to provide useful insights. We expect PaSST to outperform the CNN now that more class variety will be present.

Experiment 3.1

In this experiment the model trained for the full 50 epochs with high initial train and validation losses showing steady decrease and a decreasing rate of validation loss. The model also finished training in under one hour due to the smaller dataset size. The model achieved an F1 score of 0.3267 indicating a substantial decrease in performance from experiments with other datasets. The confusion matrix shows the model

confuses funk with hiphop, afrobeat and soul while jazz fast is also confused with dance-breakbeat, latin-brazilian-bossa and jazz. This is notable as the model does appear even in this small subset to be clustering similar features together as these confused classes likely have musicological overlap. In addition similar confusion is seen with rock being predicted in many classes incorrectly, likely again due to the over-representation of that class within the dataset.

Experiment 3.2

In this experiment the model trained for the full 50 epochs in roughly one hour with no early stopping. Upon inference, the model achieved an F1 score of 0.3911 demonstrating decreased performance compared to the prior experiments but it outperforms the CNN by a noticeable 6.5 percent. This hints that PaSST is better at dealing with more diverse drum audio data in the GMD-mini. The confusion matrix shows similar confusion with funk and hip hop, jazz fast with dance breakbeat and interestingly latin-brazilian-sambareggae with neworleans-secondline.

Discussion:

This round of experiments shows PaSST for the first time achieving higher accuracy than the baseline CNN, albeit with much lower overall F1 scores. This leads to expanding these tests to the entire GMD to observe if the similar results hold.

4. **Experiments Round Four - Expansion to the full GMD**

Now that the pipelines for both models have demonstrable functionality we expand our approach to the entire dataset.

Hypothesis:

Now that superior accuracy has been observed in PaSST over the CNN in the prior round of experiments with the GMD-mini, it is expected that PaSST-L will continue to outperform the CNN baseline when trained on the entire dataset.

Experiment 4.1

In this experiment the CNN trained for the full 50 epochs achieving an F1 score of 0.7531 indicating a substantial increase in performance from the 10 percent proxy

dataset in the previous experiment. This indicates the CNN performs better when dataset size increases even if proportional characteristics of the datasets are nearly identical. The confusion matrix shows the model confuses funk and hip hop with rock rather than other genres in the prior CNN experiment while jazz fast is also confused with dance-breakbeat, jazz-mediumfast and jazz. This is noteworthy because the model's confusion appears to be less focused as the incorrect predictions don't appear to be as musicologically related to each other and rather seem to be more related to the rock class imbalance in the dataset.

Experiment 4.2

In this experiment PaSST trained for the full 50 epochs in roughly 8.5 hours. Upon inference, the model achieved an F1 score of 0.7367 demonstrating increased performance compared to the 10 percent proxy experiment but marginally underperformed the CNN on the full dataset by roughly 1.25 percent. This gap in performance motivates further exploration for the sake of interpretability and hyperparameter tuning. For some class specifics, the confusion matrix shows jazz-fast being confused with jazz-mediumfast, dance-breakbeat, funk-purdieshuffle, afrocuban and rock halftime, additionally funk is again heavily confused with rock.

Discussion:

These results are compelling as the differences between PaSST and the CNN in terms of accuracy decreases when utilizing the full dataset. To further verify these findings repetition of these experimental conditions will be done.

5. **Experiments Round Five - Full GMD experiment repetition**

These experiments repeat those of the fourth round on the full dataset twice for each model architecture to provide a 3 repetition average.

Hypothesis:

Similar performance is expected as these experiments are a repetition of the previous round, slight variance in each training run is expected but on average the CNN should outperform PaSST.

Experiment 5.1

This experiment utilizes the A100 GPU achieving an F1 of 0.7269.

Experiment 5.2

This experiment utilizes an L4 GPU and achieves and F1 of 0.7313.

Experiment 5.3

This experiment uses A100 GPU and achieves an F1 of 0.7646.

Experiment 5.4

This experiment employs an A100 GPU and achieves and F1 of 0.7827.

These results provide a three experiment average of F1 0.7316 for PaSST and F1 0.7668 for the CNN baseline, thus the CNN beats PaSST in this configuration by 3.52 percent on average.

Discussion:

Repeating these experiments not only proves our hypothesis for this round of experiments but also widens the gap between the model types by around 2.25 percent. These repetitions assist in giving a clearer view into the performative differences between these models which will be further expanded by examining embedding visualizations for interpretability beyond accuracy.

Model Depth and Patchout

6. **Experiments Round Six - t-SNE and Time patchout**

This round of experiments begins the examination of embedding spaces for both models in different configurations for comparison. We use t-SNE as our visualization algorithm and examine the embeddings from PaSST on the full data set with and without time patchout augmentation to test the effect of this aspect of the architecture. Time patchout is chosen as a way to examine if the model is capturing time related features as it removes vertical sections from the spectrogram. This will also provide an indication if there is any difference between the representation in the embedding space for both

Table 3: F1 scores by experiment round and model (highest per round in **bold**).

Round	ID	Model	F1	Notebook	Description
6	6.1	PaSST	**0.7247**	PaSST_setup6_4.ipynb	t-SNE & Time patchout
6	6.2	PaSST	0.3861	PaSST_setup5_2.ipynb	t-SNE & Time patchout , GMD-mini
7	7.1	PaSST	**0.8582**	PaSST_setup6_5.ipynb	PaSST MLP 3 Layers
8	8.1	PaSST	0.8604	PaSST_setup6_6.ipynb	PaSST MLP 5 Layers
8	8.2	PaSST	0.7964	PaSST_setup6_7.ipynb	PaSST MLP 7 Layers
8	8.3	PaSST	**0.8659**	PaSST_setup6_8.ipynb	PaSST MLP 4 Layers
8	8.4	PaSST	0.8352	PaSST_setup6_9.ipynb	PaSST MLP 2 Layers
8	8.5	PaSST	0.8424	PaSST_setup6_10.ipynb	PaSST MLP 6 Layers
8	8.6	PaSST	0.8577	PaSST_setup6_11.ipynb	MLP 4 Layers, Symmetric Bottleneck
8	8.7	PaSST	0.8604	PaSST_setup6_12.ipynb	MLP 4 Layers, Progressive Bottleneck
9	9.1	CNN	0.8779	GMD_CNN_prototype6.ipynb	CNN 5 Conv Layers
9	9.2	CNN	**0.8944**	GMD_CNN_prototype6_2.ipynb	CNN 7 Conv Layers
9	9.3	CNN	0.8900	GMD_CNN_prototype6_3.ipynb	CNN 9 Conv Layers

the 10 percent proxy and the full dataset. All subsequent experiment rounds include t-SNE functionality in their respective notebooks.

Hypothesis:

Patchout removes sections of the spectrogram and we expect this form of augmentation to result in a slight shift in the embeddings space, indicating PaSST is sensitive to time patchout both on the GMD and GMD-mini.

The PaSST setup6-tSNE notebook shows t-SNE on the full GMD without time patchout to be compared with below. No retraining occurred here, only the initial t-SNE evaluation to visualize the embedding space. The t-SNE displays moderate cluster formation when tuning hyperparameter perplexity 30-50 which is consistent across all versions with the full dataset. Perplexity tuned to 50 with plotly[8] will be used for interactive analysis in later sections as it display the clearest presentation to compare and contrast both architectures.

Experiment 6.1

In this experiment time patchout is applied on the full dataset and retrained resulting in and F1 of 0.7247.

[8] `https://github.com/plotly/plotly.py`

Experiment 6.2

In this experiment t-SNE embedding visualization is performed on the GMD-mini with time patchout resulting in a slight decrease of -0.5 percent from no time patchout in experiment 3.2 yielding and F1 of 0.3861.

Time patchout appears to have a nearly unnoticeable effect on movement in the embedding space in both PaSST setups with the full dataset and the mini dataset. Both models show between a 0.5 - 0.8 percent percentage decrease in accuracy when performing time patchout. This suggests that our hypothesis is not proven in that time patchout augmentation does not substantially affect the embedding space and suggests the model is not capturing features that would be affected by time patchout.

Discussion:

Both embedding visualizations for GMD and GMD-mini show similar clustering but also anomalous clustering in the CNN appears to be different from PaSST ass seen in Figure 3. It is not exactly clear what this cluster is comprised of but when examining some of the classes in the plotly graph many overlap with confused classes. In addition t-SNE on the GMD-mini was not as interpretable due to the smaller number of examples in the test split. For this reason, from this point forward, only the full GMD will be utilized. The next experimental rounds will aim to further optimize the architectures by deepening the classification head on PaSST and expanding the convolutional layers of the CNN.

7. **Experiments Round Seven - PaSST MLP depth test**

In this section we attempt to improve the classifier head on top of the frozen embeddings in PaSST to examine it's effect on performance. The default classification head in PaSST is a single linear layer. This is a very shallow and simple classification head resulting in 115K trainable parameters which is likely hindering performance. To address this a deeper classification MLP is employed initially with three layers. This new MLP has LayerNORM, ReLU activation and dropout and 512 hidden-dim. In addition, since we use PaSSt-L that has knowledge distillation arch="passt-l-kd-p16-128-ap47", we also make sure the classification head for the distillation token and CLS

tokens are both updated to the MLP structure for consistency and to avoid averaging between the CLS and distillation tokens using different classifier heads, which would degrade the impact of the instantiated MLP.

Hypothesis:

The 3 layer MLP will improve performance of PaSST over the single linear layer due to increased quantity of trainable parameters.

Experiment 7.1

The 3 Layer MLP results in an F1 of 0.8582 showing marked improvement over the single linear layer.

Discussion:

It is now apparent that PaSST with the updated three layer MLP classification head surpasses the default implementation three experiment average by 12.66 percent indicating a substantial improvement. This iteration of PaSST also outperforms the 3 Conv2d layer CNN three experiment average by 9.14 percent. Because this optimization was successful additional hyperparameter tuning will be explored.

8. **Experiments Round Eight - PaSST MLP optimization and bottlenecks**

This round of experiments is aimed at determining the ideal depth of the PaSST MLP as the initial experiment used three layers and testing additional depths has the potential to reveal the optimal MLP depth. Through these experiments it was found that four layers resulted in the highest accuracy and below are the details for tests with two, four, five, six and seven layers trained on either L4 or A100 GPU's dependent on runtime availability. Experiments appear in the order they were conducted.

Hypothesis:

As the depth of the MLP increases so will accuracy up to a certain point upon which accuracy will plateau or decrease leading to discovery of the optimal depth.

Experiment 8.1

This experiment utilizes a five layer MLP with 2.4M trainable (up from 1.4M in the previous experiment) parameters and results in an F1 of 0.8604.

Experiment 8.2

This experiment employs a seven layer MLP with 3.5M trainable parameters and results in an over 6 percent decrease in accuracy from five Layers with an F1 score of 0.7964

Experiment 8.3

This experiment has a four layer MLP with 1.9M trainable parameters and surpasses the 5 layer MLP by 0.55 percent for a new high score. This is the best version of PaSST so far with an F1 of 0.8659.

Experiment 8.4

This experiment uses a two layer MLP with 866K trainable parameters and demonstrates a performance decrease of 3.07 percent compared to four layers with an F1 score of 0.8352.

Experiment 8.5

This experiment utilizes a six Layer MLP with 3M trainable parameters resulting in an F1 of 0.8424.

To seek out further optimizations of the MLP symmetric and progressive bottlenecks are used within the layers as this technique can compress the feature representations and has the potential to increase accuracy even though the MLP is implemented with 0.3 dropout. The bottleneck concept in neural networks refers to intentionally creating a layer with fewer neurons than the surrounding layers. This architectural choice serves several purposes, the primary one being feature compression which forces the network to learn more compact, essential representations by squeezing information through a narrower layer.

Hypothesis:

Bottlenecked MLP's will increase accuracy within the 4 Layer MLP due to feature compression creating a modest optimization.

Experiment 8.6

This experiment utilizes a 0.8 hidden size middle layer (symmetric) bottleneck 4 layer MLP with 200k reduction to 1.7M Trainable parameters. This results in a decrease in performance by 0.82 percent over PaSST with four layer MLP and no bottleneck with an F1 of 0.8577.

Experiment 8.7

This experiment employs a progressive bottleneck within the four layer MLP consisting of a 100k reduction to 1.8M trainable parameters. This configuration also under performs the no bottleneck four Layer MLP version of PaSST by 0.55 percent with F1 of 0.8604. In light of these results further testing with bottlenecked MLP's is ceased as both iterations appear to create performance degradation in terms of accuracy disproving the hypothesis for this experimental round.

Discussion:

These experiments show the no bottleneck, no augmentation four Layer MLP equipped PaSST is the best performing version in terms of accuracy with 86.59 percent. It also achieves these results with consistently more efficient training time duration. Now, analogous improvements to the CNN should be made to ensure fair comparison as to not 'straw-man' our contrast between the two models.

9. **Experiments Round Nine - CNN optimizations**

We utilize this round to determine the ideal depth of the convolutional layers for the CNN as the initial experiment used 3 Conv2d layers. Through the experiments detailed below it becomes evident that seven layers is the optimal CNN configuration in this context. Training setups remain consistent and experiments appear in the order they were conducted.

Hypothesis:

Adding convolutional layers will increase accuracy up to a point at which performance will either stagnate or degrade as deeper networks may not always result in improved accuracy.

Experiment 9.1

This experiment uses five Conv2d layers and surpasses the best PaSST version in experiment 8.3 by 1.2 percent with F1 of 0.8779.

Experiment 9.2

This experiment utilizes seven Conv2d layers instead of five resulting in an increased F1 to 0.8944.

Experiment 9.3

This experiment implements nine Conv2d layers resulting in a 0.44 percent decrease compared to seven layers resulting in an F1 of 0.8900.

Discussion:

These results indicate seven layers is the ideal depth for the CNN but due to compute budget and time constraints further experimentation with this aspect is not pursued. The next group of experiments will test some additional augmentation types and padding modes to observe their effects.

Augmentations and Padding

10. **Experiments Round Ten - Additional Augmentations**

    In this round we apply augmentation to test both models' robustness to perturbed audio conditions and see if they produce any drift in the embeddings. To perform this he python library audiomentations[9] is utilized. To perform this we employ 'AddGaussianNoise' and 'RoomSimulator' augmentations as they can be efficiently applied during training directly on the waveforms and do not rely on creating new files or referencing separate directories of audio files. Gaussian noise adds random noise while RoomSimulator creates the effect of a physical space upon the waveform. Both of these augmentations are not perfect but create perturbations that can make the the models more robust to variety in the audio. In addition for PaSST we will apply

[9] `https://github.com/iver56/audiomentations`

Table 4: F1 scores by experiment round and model (highest per round in **bold**).

Round	ID	Model	F1	Notebook	Description
10	10.1	CNN	**0.9080**	GMD_CNN_prototype6_4.ipynb	CNN 7 Conv Layers + GaussianNoise, RoomSim
10	10.2	PaSST	0.7953	PaSST_setup6_13.ipynb	MLP 4 Layers + GaussianNoise, RoomSim
10	10.3	PaSST	0.8446	PaSST_setup6_14.ipynb	MLP 4 Layers + TimeStretch
10	10.4	CNN	0.8632	GMD_CNN_prototype6_5.ipynb	CNN 7 Conv Layers + TimeStretch
11	11.1	PaSST	0.8429	PaSST_setup6_15.ipynb	PaSST MLP 4 Layers, Circular padding
11	11.2	PaSST	**0.8752**	PaSST_setup6_16.ipynb	PaSST MLP 4 Layers, Reflection padding
11	11.3	CNN	0.8747	GMD_CNN_prototype6_6.ipynb	CNN 7 Conv Layers, GaussianNoise, RoomSimulator, Reflection padding
11	11.4	CNN	0.8736	GMD_CNN_prototype6_7.ipynb	CNN 7 Conv Layers, Reflection padding
11	11.5	CNN	0.8747	GMD_CNN_prototype6_8.ipynb	CNN 7 Conv Layers, Circular padding

'TimeStretch' in a separate experiment to investigate if drift occurs in the embedding space as changing the duration but not the pitch of the audio can further elucidate how PaSST is affected by time. These tests with augmentation are limited but serve as tools to observe how each model handles noise and time disturbances beyond the original provided audio. These waveform based augmentations added significant compute resource intensity and increased train time.

Hypothesis:

PaSST will be more robust to time and noise perturbations due to the AudioSet embeddings, disentangled time and frequency positional encodings and will exhibit a slight decrease in accuracy but not as much as the CNN. An effect in embedding drift is also expected due to waveform based augmentation.

Experiment 10.1

This experiment results in a surprising 1.36 percent increase in accuracy over no augmentations. Lower validation loss during training over the previous best CNN is also observed, indicating these augmentations not only helped reduce the chance of overfitting but also increased accuracy. This experiment produces the highest score for the

CNN with seven Conv2d Layers for this project and thus achieves the highest accuracy on drum style classification within the GMD with an F1 of 0.9080.

Hypothesis:

A modest increase in accuracy similar to what occurred with the CNN is expected due to increased robustness from audio variety and perturbations.

Experiment 10.2

In this experiment surprisingly, PaSST loses 7.06 percent of accuracy from the best PaSST with 4 layer MLP and no augmentation in experiment 8.3 (0.8659 F1). Full results show an F1 of 0.7953.

Hypothesis:

A small decrease in accuracy along with embedding drift is expected due to TimeStretch augmentation as these are changing the original durations of the variable length drum audio.

Experiment 10.3

In this experiment embedding drift occurs that at a glance appears quite different from the patchout augmentation especially around 50 perplexity on the t-SNE and will explore this further in the analysis section.

Discussion:

These experiments bring about some compelling results and insights. The CNN performs at it's best when employing RoomSimulator and Gaussian Noise in experiment 10.1 but when applied to PaSST in 10.2 the effect is actually detrimental in terms of accuracy. Both models also lose accuracy when TimeStretch is implemented. This shows that PaSST is generally lest robust to augmentations when compared to the CNN. From this we move to the last experimental round exploring different padding modes.

11. **Experiments Round Eleven - Circular and Reflection Padding**

In this round different modes of padding are applied to observe if there are effects

accuracy, latent representations and training. Before this padding with zeros was the only mode utilized. The torch functionality for padding is used with mode switching to 'circular' and 'reflection' in different tests. These padding modes are invoked in such a way that the padding occurs in chunks smaller than the current length up until the max of 10 seconds and is trimmed to that exact length in samples if necessary. Two experiments are conducted on PaSST then three on the CNN and these experiments continue to appear in the order conducted.

Hypothesis:

PaSST and the baseline CNN will exhibit modest increases in accuracy due to circular and reflection padding as these give the model more information to learn from as opposed to padding with zeros.

Experiment 11.1

Circular padding surprisingly results in a 2.3 percent decrease over the best PaSST model. Even though this is the case train and validation loss are very stable throughout and the validation accuracy curve appears to not converge as fast as previous rounds. This may suggest that PaSST could benefit from a max training epochs greater than 50 as implemented. Full results show F1 of 0.8429.

Experiment 11.2

Reflection padding results in a 0.93 percent increase in F1 over the best PaSST model, meaning this setup achieves the best out of all PaSST experiments with the full GMD. Train and validation loss are less stable than with circular padding and similar to the last round the validation accuracy curve is not entirely flat at the end of training further corroborating that PaSST may benefit from higher max epoch count. Full results show F1 of 0.8752.

Experiment 11.3

Switching back to the CNN, GaussianNoise, RoomSimulator and reflection padding results in a 3.33 percent decrease in F1 over the best CNN model. These augmentations are kept in for this experiment as they previously improved the CNN's accuracy

however this shows the model performs worse when padding mode is changed from zeros to reflection. For the train and validation loss curves it is mostly stable with the exception of a spike around the 25th epoch. In addition when examining the validation accuracy curve it appears the CNN converged at around epoch 30 with the curve mostly flat from thereon out as the early stopping was triggered at epoch 40. This is an interesting change in the training because it shows that the CNN converges faster than PaSST while achieving nearly the same F1 of 0.8747.

Experiment 11.4

The only change in this experiment from the last is the removal of augmentations to isolate the effect of the change from padding with zeros to padding with reflection. This test with just reflection padding results in a 3.44 percent decrease in F1 over the best CNN model. Interestingly, the train and validation loss curves mostly plateau after epoch 12 of 50 which was not observed in previous rounds. In addition, when examining the validation accuracy curve it appears the CNN improves very little after around epoch 15 of 50 showing faster convergence than PaSST. From this it is also noted that augmentations help in terms of training stability but do not in terms of F1 with the result of 0.8736.

Experiment 11.5

The last experiment conducted uses circular padding only and results in a 3.33 percent decrease in F1 over the best CNN model. Like before, augmentations are removed to isolate the effect of the padding mode change. Similar to the last experiment with just reflection padding, train and validation loss curves appear to be more unstable than padding with reflection. Validation loss generally plateaus after epoch 13 of 50 and when examining the validation accuracy curve, the CNN improves very little after around epoch 20 of 50 adding evidence that it converges faster than PaSST but with lower accuracy when padding modes are changed from zeros. Full results show F1 0.8747.

Discussion:

These experiments elucidate a few important attributes of the models. Firstly it is

apparent that the CNN works best with zero padding and PaSST works best with reflection padding. This may indicate that with longer drum audio samples PaSST would be superior as the CNN is more effective with dealing with zeros. Additionally, circular padding against our intuition is not as effective as reflection padding in both models. We can also see that based on the validation accuracy curves PaSST may benefit from a higher training epoch limit as the CNN converges faster but PaSST training curves hint that it could benefit from additional epochs to fully converge and enact early stopping. In the CNN we generally observe somewhat more unstable training when padding modes are changed in addition to a reduction in F1 over the pad with zeros method. In regards to training time we did not notice anything divergent from previous rounds so padding mode changes do not seem to affect this aspect. This final experiment section concludes with a new best score for PaSST and some insights that hint will be explored in the analysis section.

8 Analysis

This section will aim to derive insights from the conducted experiments. Because of the 'black-box' nature of these models, inspecting the embedding space computed from the t-SNE along with results from the confusion matrices will help increase model interpretability. We group these analysis rounds by each architecture type and conclude with shared insights between both that are documented in the results sheet [10] .

PaSST Insights

1. **Embeddings: Top 10 style classes by centroid shift**

   Analyzing by centroid shift elucidates how the center of clusters move based on different experimental choices. The largest mean centroid shift in the most moved classes comes from pad with reflection and zero padding comparison. This experiment comparison is also accompanied by a 3.06 percent increase in accuracy meaning reflection padding

[10] https://docs.google.com/spreadsheets/d/1l6qoCJU3BJl25YsMWcbxRspjxEavY37Ds0NKYRluSn0/edit?usp=sharing

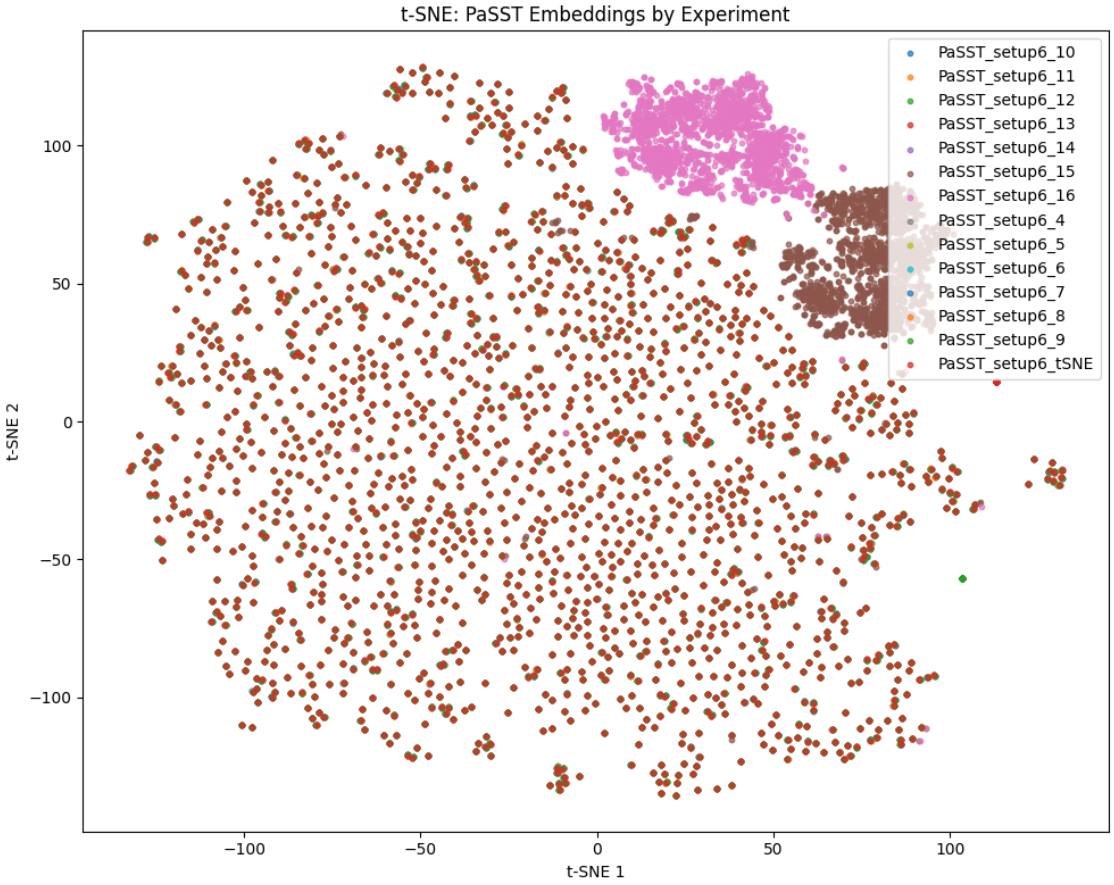

Figure 1: t-SNE across PaSST experiments.

is more accurate than padding with reflection and time stretch, while no substantial movement occurs when comparing zero padding with and without time stretch.

When comparing the reflection padding to circular padding experiments the mean centroid shift drops from 8.32 to 6.82. The classes in common with the prior comparison are 'latin-reggaeton' , 'soul-groove10' , 'soul-groove4' , 'pop-soft' and 'pop-groove7'. This shows that at least there is some overlap in the classes that shift the most due to changes in padding modes. This comparison also is accompanied by 3.33 percent lower accuracy for circular padding compared to reflection padding.

Then when comparing circular padding with progressive bottlenecking the mean centroid shift drops from 6.82 to 6.30 when contrasted with the previous comparison meaning circular padding produces less shift in the embedding space than reflection padding. Interestingly the top 10 shifted style classes in this comparison have no over-

lap with the previous comparison but share some overlap with the first, that being with style classes 'jazz-klezmer' , 'latin-merengue' , 'rock-rockabilly' and 'latin-ando'. It is difficult to determine an exact reason for this phenomenon but it does illustrate that different classes move differently in the embedding space based on padding and bottle-necking. Here it is good to recall that circular padding proves less effective in terms of accuracy when compared to zero padding and progressive bottleneck as shown in experiments 8.6, 8.7 and 11.1 in Table 5.

When examining these comparisons it becomes apparent that reflection padding produces substantially more embedding shift than time stretch on zero padding with certain classes moving the most (jazz, latin, rock, pop and soul). This also continues but with less mean centroid shift when circular padding is used further indicating that padding mode changes produce the most embedding movement. There also does not seem to be a clear connection between change in accuracy and embedding movements. This is also evident in the t-SNE as there are clear separate clusters when these padding modes are applied as seen in Figure 1. Now that the most shifted classes are analyzed, a similar analysis by primary styles will be explored in the next subsection.

2. **Embeddings: Primary Style and Centroid Shift**

When comparing all of the embeddings grouped by primary style the most shift across all 17 primary styles comes from reflection padding. Through conducting the same three comparisons from the last subsection, there is a different rank order of the primary styles with the most shift. We also note the same trend that reflection padding when compared to zero padding with time stretch produces the most mean centroid shift followed by the comparison with reflection padding vs. circular padding and thirdly by circular padding vs. zero padding with progressive bottleneck. One commonality is the primary style 'dance' shifts the least in all three comparisons. There are also similar rank orderings for most shifted primary styles across the three comparisons with 'rock', 'afrobeat' and 'funk' primary styles. These results show that when substantive embedding cluster centroid shift occurs it effects broadly amongst the primary styles with the only commonalities being in a few of the primary styles which are in themselves

inconsistent across experiment comparisons.

3. **Confusion Matrices comparison**

   When analyzing the confusion matrices, focus is placed on experiments with the most notable change from the best PaSST without augmentation achieved in experiment 8.3 which results in examining experiments 10.2 , 11.2 and 11.2 in Table 5.

   Comparing 8.3 with 10.2 examines the effect of the augmentation GaussianNoise and RoomSimualtor. As noted in the experiments section F1 drops from 0.8659 to 0.7953 in this experiment.

   These augmentations make the model more confused with certain style classes. For example 'funk' , 'latin', 'hip hop' and 'rock'. Jazz styles are also confused and notably jazz fast exhibits the exact same confusion between models with being most misclassified as 'dance-breakbeat' or 'jazz-medium fast'. Additionally 'latin-brazillian-bossa' is also more confused with 'jazz-funk' and 'jazz' , 'Neworleans-funk' with 'dance-disco' and a slight amount more confusion in rock classes. This demonstrates that when these augmentations are introduced you see more confusion with classes that are likely musically similar. For example it can be speculated that 'dance-breakbeat' or 'jazz-medium fast' are more confused because they both likely contain rapid snare drum hits and ghost notes who's transients are smeared slightly by the augmentations causing a drop in accuracy. A similar speculation could be made for 'latin-brazillian-bossa' being confused with 'jazz-funk' and 'jazz' as bossa nova as a genre is historically influenced by jazz which could be apparent even in just the drum style. These speculations would need further research to be confirmed but this analysis shows that when drum audio is augmented with GaussianNoise and RoomSimulator, additional details emerge.

   Comparing experiemnts 8.3 with 11.1 examines the effect of circular padding. As noted in Table 5 F1 drops slightly from 0.8659 to 0.8429. 'Funk' is classified more accurately and the nearly identical misclassification of 'jazz-fast' is apparent. Jazz and latin styles generally see slightly more confusion with apparently less musical coherence ; for example 'latin-brazillian-bossa' is now more confused with funk and rock. This round of comparison leads us to speculate that as accuracy increases the confusion

becomes more scattered among classes where this less of a clear musical connection or a less clear underlying feature overlap. What this could mean is that with circular padding the model correctly classifies some of those examples we examined before but is still left with outliers.

Comparing 8.3 with 11.2 examines the effect of reflection padding. As noted in Table 5 F1 increases slightly from 0.8659 to 0.8752 as it achieves the best score for PaSST in the entire project and beats the CNN within experiment round 11. The difference in accuracy between these is less than one percent so distinguishing them is a difficult task. Like the last comparison there is noted improved accuracy in the funk classes with 'jazz-fast' misclassified in the same way, demonstrating that the features related to those are consistently confused regardless of the experiments we ran. There are mixed results in the latin classes by comparison with 'latin' and being less confused but 'latin-brazillian-bossa' being confused with 'rock' and 'reggae'. Generally, slight improvements exist across most classes due to the marginally higher accuracy.

In summary by comparing these confusion matrices there appears to be a few key takeaways. When GaussianNoise and RoomSimulator augmentations are introduced, accuracy drops but potentially musically related style classes are revealed. Circular padding decreases accuracy over no augmentations and struggles with some outlier classes. Lastly, reflection padding increases accuracy but has similar outliers. The key insight gained is that slightly lowering accuracy through waveform augmentations revealed some similarly confused classes across experiments with the most changes in accuracy.

4. **t-SNE comparison**

In the PaSST-setup6-tSNE-eval2 notebook[11] the embedding space between experiments is examined. The most notable insight is that there is very little movement in the embeddings across all the experiments until we change the padding modes to reflection or circular as shown in Figure 1. With perplexity tuned to 50 a clear cluster is evident for both padding modes. This is notable because the augmentations provided

[11] `https://colab.research.google.com/drive/1IUAfOao5JlcgFeFcd_T397L-qOGEIulS?usp=sharing`

the most change in accuracy (2-7 percent) but did not produce a noticeable embedding movement while concurrently substantial movement of the embeddings from pad mode changes only produced moderate changes in accuracy of roughly 1-2.5 percent. This continues to show that embedding space movement does not change in conjunction with accuracy which leads into conducting the same analysis for the CNN.

CNN Insights

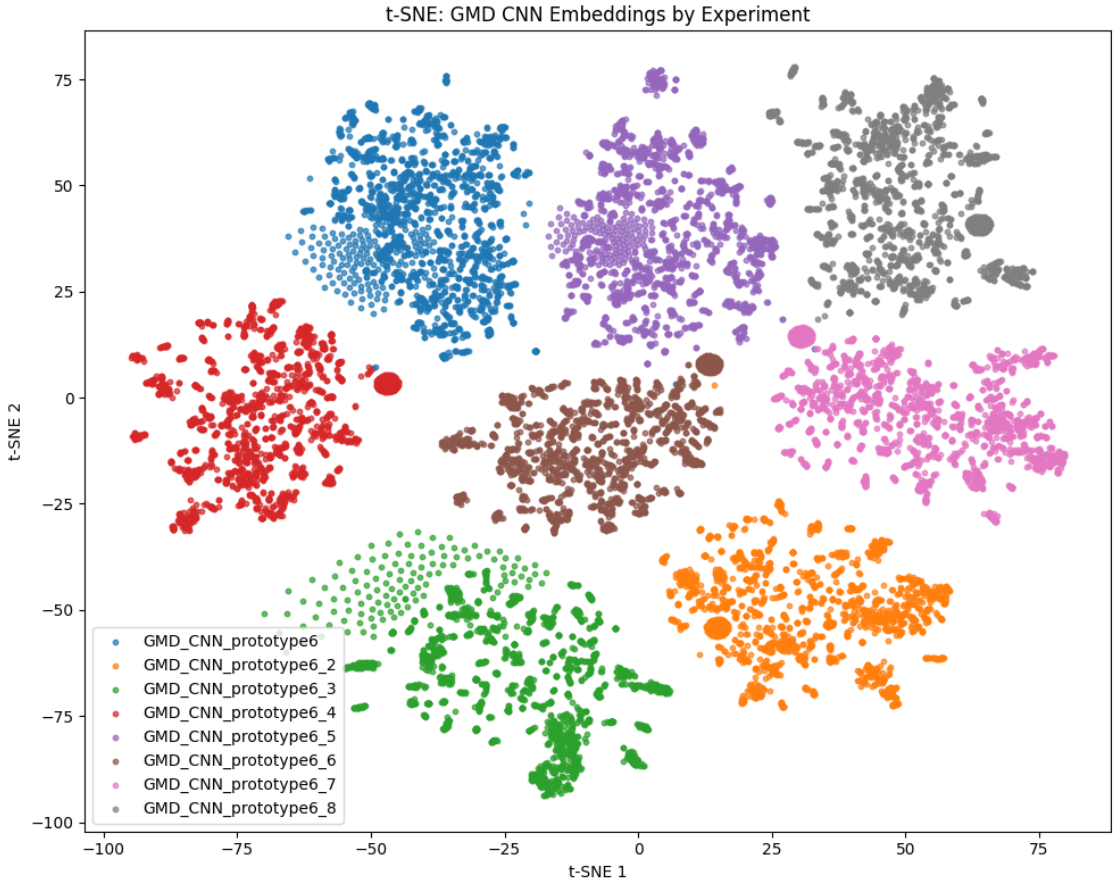

Figure 2: t-SNE across CNN experiments.

1. **Embeddings: Top 10 style classes by centroid shift**

   To compare embeddings concisely focus is placed on experiments with the most centroid shift after convolutional layer depth is optimized at 7 Conv2d layers narrowing the comparison to experiments 9.2 with 10.1 , 10.4 , 11.4 and 11.5 shown in Table 5. The

CNN overall produced roughly 10 times more shift compared to PaSST as evidenced by the t-SNE visualizations with metrics noted in the results sheet just as they were for PaSST[12].

Similar to PaSST, The greatest mean centroid shift in the CNN for most moved classes comes from reflection padding and zero padding comparison. However unlike PaSST, this experiment comparison results in a 2.08 percent decrease in accuracy meaning reflection padding is detrimental to the CNN's accuracy performance. When comparing zero padding with reflection padding (Experiments 9.2 and 11.4) 6 of the top 10 most shifted classes are the same these being in order of shift 'rock-rockabilly' , 'dance-disco' , 'funk-latin' , 'afrobeat' , 'rock-groove8' , and 'funk'. It should be noted here these 6 style classes are in the top 10 for all experiments examined here underlying significant commonalities between them.

Then when comparing zero padding with circular padding (Experiments 9.2 and 11.5) the mean centroid shift drops marginally lower than with the previous comparison. Within the top 10 shifted classes the main notable change is the addition of style class 'latin-brazilian-sambareggae'. The rank order of shift for these classes is also changed but the differences are small as the overall delta in accuracy is 1.93 percent. From this it is clear only that circular padding created more drift for certain latin styles.

When contrasting Zero padding vs. Zero Padding with Time Stretch augmentation (Experiments 9.2 and 10.4) mean centroid shift is again marginally lower than our last comparison. Notably here another latin class added to the top 10 that being 'latin-dominican-merengue' in addition to 'reggae-slow'. The confusion matrices show that reggae and latin are often confused across this experimental comparison showing that a drop in overall accuracy (0.8944 to 0.8632) also coincides with embedding drift in certain classes.

Lastly, Zero padding vs. Zero Padding with GaussianNoise and RoomSimulator (Experiments 9.2 and 10.1) show mean centroid shift lower of the examined comparisons, which is notable because this round had the highest accuracy demonstrating that higher

[12] `https://docs.google.com/spreadsheets/d/1l6qoCJU3BJl25YsMWcbxRspjxEavY37Ds0NKYRluSn0/edit?usp=sharing`

accuracy coincides with lower mean centroid shift among the top 10 most shifted style classes.

In summary, changes in padding mode produce the most mean centroid shift among the top 10 shifted style classes, which is consistent with PaSST. Timestretch augmentation also does not have a major effect on embedding shift but it does produce lower accuracy. Lastly Gaussian Noise and Room Simulator Augmentations produce less mean centroid shift while also increasing accuracy. The co-occurrence of lower top 10 style class mean centroid shift and higher accuracy appears to be a key distinction from performance in PaSST. Additionally the top 10 style classes by mean centroid shift have very little overlap with the experiments examined for PaSST which leads into investigating primary styles to see if these phenomena continue.

2. **Embeddings: Primary Style and Centroid Shift**

   When comparing the CNN embeddings grouped by primary style it is further shown that the most shift comes from reflection padding, just as it did in PaSST. When noting the same experimental comparisons inspected in the last subsection, there is a slight difference in rank order of the primary styles with the most shift. When grouped by primary style the order of most shift by experiment comparison is the same. This shows that analyzing the top 10 most shifted style classes and by primary style provides a similar representation generally across experimental comparisons. Interesting, primary style 'dance' shifts near the top in all comparisons where in PaSST it shifted the least. Similar primary styles are present near the top of most shift like 'hiphop', 'dance' ,'afrobeat' and 'funk'. When compared with PaSST some of the classes appear to have musical overlap (for example funk and hiphop are likely to be musically similar) but this does still seem to indicate a difference between how the models deal with features of these classes. Overall, these results show that when substantive centroid shift occurs it affects similar classes by rank order slightly more so than in PaSST.

3. **Confusion Matrices comparison**

   Similar to how there were compared with PaSST, close examination is only performed on experiments that result in the most notable change from the best CNN without

augmentation achieved in experiment 9.2 with 7 Conv2d layers (0.8944). The following will then examine experiments 10.1 , 10.4 and 11.4 seen in Table 5

Comparing experiment 9.2 with 10.1 tests the effect of GaussianNoise and RoomSimualtor augmentations which resulted in an F1 increase from 0.8944 to 0.9080. This result was unexpected as these augmentations were expected to result in a drop in accuracy but instead lead to the highest F1 of any experiment in the project. Upon examining the confusion matrices the augmentations appear to reduce the confusion between jazz and funk classes with the exception of 'jazz-fast' which is again confused with 'dace-breakbeat', 'funk', 'afrocuban' and 'jazz-mediumfast'. This shows that the CNN and PaSST struggle in nearly the same way with this particular style likely hinting that they have underlying similarities in the feature representation for these two classes. Additionally, the augmentations make the model confuse 'latin-brazilian-bossa' less with 'jazz' and more with 'reggae' and 'rock' without any increase in accuracy, hinting that the model is more influenced at least in part by the over-representation of the 'rock' class. Outside of this generally there are small increases in accuracy for latin related classes and marginal decreases for classes like 'rock-halftime' although it is difficult to make definitive statements from these findings as the difference in accuracy between these models is less than 1.5 percent.

Comparing experiments 9.2 and 10.4 seeks to examine the effect of TimeStrectch augmentation which results in an F1 decrease from 0.8944 to 0.8632.

It is immediately noticable that the model is slightly more confused with funk related classes but the confusion is with classes like 'hiphop' and 'jazz-funk' hinting at an underlying similarity in feature representation broadly for that style. Jazz again is more confused with other jazz styles with 'jazz-fast' exhibiting nearly identical confusion compared to the last evaluation corroborating that time stretch is not influencing at least this particular style class. Time stretch also reduces accuracy generally across latin styles with for example 'latin-brazilian-bossa' accuracy dropping for more confusion with rock related classes while still maintaining some jazz confusion. This appears to indicate that time stretch makes the model more sensitive to class imbalances. The

style class ‘rock-halftime’ again noticeably drops in accuracy being confused with ‘jazz’ ,’and funk-purdieshuffle’ which is compelling because especially the famous prude shuffle beat invented by drummer Bernard Purdie could be seen as quite musically similar to rock-halftime. This is yet another hint that some form of underlying feature representation may be present between certain confused classes. Lastly, other classes see marginal decreases in accuracy as expected due to the overall lower accuracy.

Comparing experiments 9.2 and 11.4 probes the effect of padding with reflection which results in a F1 decrease from 0.8944 to 0.8736.

The first notable difference compared to the last analysis is that reflection padding does not increase confusion with funk classes as much. Both models have nearly identical performance with those classes. The model again struggles significantly with ‘jazz-fast’ confusing it with the same classes as in prior comparisons. Accuracy drops for most latin classes and the same phenomenon with ‘latin-brazilian-bossa’ occurs where it’s confusion becomes confused centered around the rock class. This is another instance of observing that as augmentations or padding modes are changed the model becomes more sensitive to class imbalance. Other notable confusions are ‘New Orleans-funk’ with many latin styles and some rock styles. This confusion appears less musically cohered to previous rounds but the confusion with certain latin classes hints at the potential for shared underlying feature representation due to musical similarity.

4. **t-SNE comparison**

These comparisons begin on experiments where the convolutional layers have been increased from the 3 layer original prototype as it provides a more precise comparison between optimizations within the hyper parameters. All t-SNE’s are evaluated at perplexity tuned to 50 to ensure consistency with other comparisons shown in the overview in Figure 2 .

Firstly when experiments 9.1 and 9.2 compare five and seven convolutional layers, clearer clustering is evident with seven layers as specifically style classes often form miniature clusters rather than being more broadly clustered by primary style. Classes like funk and hip hop are more separated in seven layers while jazz styles generally also

seem more tightly clustered. Here it is key to note that a structure in the embedding space emerges that is evenly spread then composes a nearly symmetrical shape. It appears anomalous as it appears distinct from all other clusters but contains a variety of seemingly unrelated classes within it. It is not entirely clear what this is but it could be related to the confused classes or the beat / fill designations present in the GMD. This structure of 'misfits' is also much more tightly clustered in the seven rather than five layers. This phenomenon indicates the model is somehow clustering these instances from disparate classes. In experiment 9.3 with 9 convolutional layers this anomalous structure is more spread out over the embedding space while much of the other clusters stay largely intact indicating this group of embeddings is more sensitive to model depth. Tighter clustering appears to coincide with higher accuracy as seven conv2d layers had the best F1 in its experiment round of 0.8944 as opposed to the 0.8900 with 9 layers which is one aspect where the CNN and PaSST diverge.

In experiment 10.1 we achieve the highest F1 on the CNN of 0.9080 and note movement in the embedding space but similar clustering. In addition the anomalous cluster appears to be heavily focused around confused classes relating to 'jazz-fast' and 'jazz-mediumfast' that we noted both in the CNN and PaSST further showing that this cluster may indeed be constituted of misclassified examples.

Furthermore when GaussianNoise and RoomSimulator is swapped for TimeStretch a clear reduction in cluster density in some primary styles is observed but more noticeably in the anomalous cluster mentioned previously. This could mean that TimeStretch is smearing transients as there is a drop to F1 of 0.8632.

The remaining experiments of round eleven that focus primarily on different padding modes do create movement in the latent space but similar clustering occurs across all three. Potentially similar musical styles are clustered near each other similar to previous observations hinting further that the CNN also has some underlying feature representation on musically related classes however the emergence of the anomalous cluster elucidates some form of commonality between it's constituent classes that we suspect has to do with drum fills rather than beats that may have more varied transients

and lack of prominent downbeats although this would need further experimentation to be confirmed.

Embedding Comparison Summary

The baseline CNN and PaSST have noticeably different representations in the embedding space. The CNN appears to be more capable at clustering specific style classes together while PaSST's clustering represents the potential for more cohesion to primary style. The CNN may more clearly separate these classes because it is trained solely on the GMD and has no pretraining from general audio embeddings.

However both architectures appear to exhibit an underlying representation of features linking similar styles as evidenced by similar primary styles clustering near each other although to different degrees. The CNN embeddings exhibit an anomalous cluster that is not present in PaSST which is suspected to pertain to misclassifications or drum fills based on examining the original GMD metadata and confusion matrices in conjunction with the t-SNE. If this were the case it would indicate PaSST is better at dealing with these anomalies although it achieves overall lower accuracy. It is also noted that embedding shifts coincide more with accuracy changes in the CNN rather than PaSST although this is not confirmed to be causal or correlative. The CNN's increased embedding shift as shown Figure 2 when compared to PaSST's in Figure 1, indicates differing feature representation sensitivity as the CNN embeddings move with each experiment while PaSST embedding movement is only notable when padding modes are changed.

9 Discussion

The prior sections of this project demonstrate the baseline CNN is able to perform exceptionally well in GMD drum audio style classification and that the potential for transfer learning with PaSST from general audio embeddings does not have a positive impact in terms of accuracy. This could derive from a variety of reasons, one of which being the 'world model' that these embeddings exemplify does not contain within it a representation of the features that would aid in the classification of drum audio style. AudioSet features lead to lower

accuracy in PaSST but a more nuanced examination shows the potential for more robust underlying feature representation than the CNN. Additionally the limited compute budget of this project potentially contributed to PaSST's performance not meeting the initial expectations of exceeding the CNN.

The clustering behavior of the CNN demonstrates notable differences from PaSST, namely that individual style classes are more distinct and outliers are clustered together rather than by their primary styles. This suggests the CNN is learning more local features for drum audio style classification and struggles with a certain subset of the GMD in a way that PaSST does not. This could indicate the baseline CNN has a lesser degree of broader primary style based feature understanding compared to PaSST.

PaSST interestingly outperforms the CNN on the GMD-mini showing that as data size increases to the full GMD the influence of the Audioset embeddings begins to degrade accuracy in comparison to the CNN. It is also noteworthy that before optimizing model architectures, the accuracy differences between them are often less than 1.25 percent. This hints that with more detailed hyperparameter tuning and compute budget, PaSST could plausibly achieve accuracy within a smaller margin of the baseline CNN.

Another point of interest is training time, while both models appear to train in roughly similar durations, once any form of waveform augmentation is introduced the CNN exhibits much more need for compute. Augmentation experiments were deliberately limited with no more than two happening concurrently to avoid excessive affect ambiguities. In either case both models perform exceptionally well and likely exceed most human experts, although this would need experimental verification. As one of the distinguishing features of PaSST, structured time patchout surprisingly degrades performance where it was designed to create an advantage for classification. This result should have been anticipated because drum audio styles are a result of onset patterns over time and if sections of this are removed from the spectrogram the model, this could create confusion. It is suspected that many of the confused styles classes deal with short and low amplitude transients that patchout and other augmentations smear which further confuses the model becoming less disambiguated from the frozen embeddings.

For PaSST, bottlenecking the MLP both in a symmetrical and progressive fashion hin-

ders accuracy, although the progressive bottleneck was slightly more performative. This technique was minimally explored and could be expanded to utilizing different dropout rates or designing a more sophisticated MLP. The architectural changes made in this project were exploratory in nature to examine effects in terms of accuracy, embedding drift and train time. In a context where compute resources and time are less constrained, more abundant hyperparameter tuning could continue until a more optimal state is reached.

Another result that was not expected in Experiment 10.1 in Table 5 where the CNN achieved nearly 91 percent accuracy surpassing all other experiments. We can speculate that the gaussian noise helps the model more clearly disambiguate transients or downbeats although this would need to be experimentally confirmed. As for the effect of RoomSimulator, since it was not employed on its own, it's not clear what the results would be in the absence of gaussian noise. This would also need experimentally verified as the inclusion of acoustic reflections could produce a variety of effects on the overtones in audio files that could alter the signal such that classification accuracy is affected.

Lastly we note that upon close examination of the embeddings and confusion matrices that both models struggle with certain classes consistently, those being often in jazz, latin or funk style classes. However even though both of these models become confused in these classes, in the CNN the confusion is more related to class imbalance of the rock style while in PaSST confusion is more related to primary style, hinting at a more robust feature representation. It is difficult to point to a specific reason for these misclassifications with one interesting note that 291 of the original files were labeled 'fill' rather than all others as 'beat' so it is possible this is playing a role but would need further investigation.

Significance

To our knowledge this is the first study available that not only explores strictly audio based style classification from the GMD but also that investigates the relationship between general audio embeddings in a pretrained AST based model for the task of drum audio style classification. The methods undertaken in this project have the potential to be improved upon such that they can be applied to for example applied to other drum audio contexts. The pipelines built for this project can still be utilized for drum audio style in other contexts

given proper regularization. The analysis of the embeddings over experimental variations also uncovers differing feature representation for the GMD across architecture types creating ample context for future research.

Limitations

One significant limitation is that the GMD is highly imbalanced towards one annotation with the primary style 'rock' accounting for around two thirds of the dataset. A more balanced dataset or more sophisticated methods to mitigate class imbalance may have led to different results. Although the dataset is for the most part very well organized by its original authors, this is certainly a flaw. In addition as we have alluded to earlier, the GMD style annotations include some level of subjectivity and could lack nuance in some cases that may have led to more confusion in the ultimate results of the classification.

This project also utilized only the audio files while the MIDI was also available. The possibility to build separate pipelines for MIDI data then fuse them into an ensemble model has the potential for elucidating if inclusion of MIDI data along with audio would improve performance. However, solely utilizing audio mitigates scope creep while also making the model potentially more applicable to real world environments where MIDI data would not be present.

Lastly, this project was limited by access to high throughput compute, even though L4 and A100's were used in the Google Colab environment, at times unpredictable fluctuations in training speed occurred which created numerous difficulties.

Future Research

There are essentially two main directions in which this research could be broadened; those being the expansion of datasets utilized and combining these pipelines with others to facilitate the creation of ensemble models for other contexts.

One direction of future research would be to incorporate the E-GMD[13], an expanded version of GMD that is much larger and varied. Leveraging the E-GMD would explore

[13] `https://magenta.tensorflow.org/datasets/e-gmd/`

whether models trained on a broader set of timbral and stylistic variations learn more robust and widely applicable features. Due to the more varied nature of the E-GMD it may also perform better in supervised or unsupervised transfer learning contexts which would bring new research insights for drum audio style classification. Additionally becuase the E-GMD does not appear to have presegmented audio with appropriate metadata, it would open up the potential for the application of few shot learning. Adapting the E-GMD to this pipeline would however involve the need to create segmented audio files based on available MIDI and a precise indexing protocol akin to what was done the GMD along with other potential architectural modifications and hyperparameter tuning. This appears feasible due to MIDI data inclusion with the audio and has the potential for setting a state of the art benchmark for drum audio style classification on the largest known dataset of it's kind. We can speculate these models would perform slightly differently in terms of accuracy compared to the results of this project but would still be performative. This future direction would also be more generalizable to different types of drum timbres making it more usable in our next type of future direction; application to polyphonic full mix audio.

For the second direction of future research an exploration of application to non MIDI aligned / based audio would be worth exploring. This would likely involve using real word field recordings from acoustic drum recordings of various types. Since this project has shown the classification performs above 85 percent when regularized properly it would be compelling to attempt this with recordings that are less regularized or structured. This would have the potential to uncover more nuanced styles that may exist in the at times ambiguous area of primary styles. Additionally some of these recordings would likely contain additional environmental noise and since AudioSet contains general audio embeddings there could be the potential to detect and segment these environmental sounds from drum styles. This would make the models more generalizable to real world contexts and thus could eventually lead to more real time application of the classification pipelines. This would also reduced the need for human created annotations which can lead to faster data set creation that can be utilized in subsequent studies.

10 Conclusion

In conclusion, this thesis conducted to our knowledge the first known comparison of supervised transfer learning for PaSST and a baseline CNN on the GMD for drum audio style classification. The results indicate that AudioSet embeddings when frozen in PaSST did not provide an advantage in terms of accuracy over the baseline CNN as originally hypothesized. Examining the embeddings space reveals that even though PaSST achieves lower accuracy it appears to exhibit more robust primary style based clustering while CNN clustering exhibits clearer mini-clusters around individual style classes hinting at more adherence to local rather than global features.

Waveform augmentations helped elucidate similarities in confused classes and demonstrated the baseline CNN was more sensitive to class imbalance. Additionally, these augmentations had varied effects when applied to both model architectures with GaussianNoise and RoomSimulator improving the CNN while degrading PaSST. Timestretch also decreased performance in both the CNN and PaSST. PaSST unexpectedly improves to its highest score with reflection padding whereas the CNN exhibits a small performance decrease from both reflection and circular padding. Meanwhile these padding mode changes produce the only substantive movement when compared to other PaSST experiments. This also led to the observation that in PaSST, mean centroid shift did not clearly correspond to changes in accuracy while in the CNN higher accuracy coincides with lower mean centroid shift.

Model depth and patchout experiments demonstrated anomalous clustering behavior in the CNN and patchout augmentation, a key feature of PaSST, did not improve the models accuracy or produce embedding drift. PaSST-L with less transformer blocks was shown to be ideal for the task which increased accuracy and training efficiency over other pretrained configurations. PaSST outperformed the baseline CNN on GMD-mini but not on GMD although the gap in performance between the two shrunk to 3.27 percent when utilizing all style classes present in the dataset. For PaSST a four layer MLP was found to be the ideal depth and that when bottlenecked, performance did not improve. For the CNN it was shown that seven layers was the optimal depth.

Confusion matrix comparison also showed both models struggled often with similar classes

but the CNN was at times more confused with the over-representation of the 'rock' annotation.

These results indicate the 'world model' from AudioSet embeddings exhibits a nuanced relationship to the task of drum audio style classification. Although accuracy with these embeddings in PaSST is lower than the baseline CNN, when the CNN is challenged with padding mode changes and certain waveform augmentations, its embeddings are more sensitive to centroid shift and class imbalance. Furthermore, PaSST maintains more stable primary style clustering and lower mean centroid shift indicating that general audio embeddings may aid in creating a more robust underlying feature representation for drum audio styles.

References

[1] K. Koutini, J. Schlüter, H. Eghbal-zadeh and G. Widmer, "Efficient Training of Audio Transformers with Patchout," *arXiv preprint arXiv:2110.05069v3*, 2022.

[2] S. Chen, Y. Wu, C. Wang, S. Liu, D. Tompkins, Z. Chen and F. Wei, "BEATS: Audio Pre-Training with Acoustic Tokenizers," *arXiv preprint arXiv:2212.09058*, 2022.

[3] K. Koutini, H. Eghbal-zadeh and G. Widmer, "Receptive Field Regularization Techniques for Audio Classification and Tagging with Deep Convolutional Neural Networks," *arXiv preprint arXiv:2105.12395v1*, 2021.

[4] A. Quelennec, M. Olvera, G. Peeters and S. Essid, "On the Choice of the Optimal Temporal Support for Audio Classification with Pre-Trained Embeddings," *arXiv preprint arXiv:2312.14005v1*, 2023.

[5] Y. Ding and A. Lerch, "Audio Embeddings as Teachers for Music Classification," in *Proc. of the 24th Int. Society for Music Information Retrieval Conf. (ISMIR)*, Milan, Italy, 2023. *arXiv preprint arXiv:2306.17424v1*, 2023.

[6] K. Choi, G. Fazekas, M. Sandler, and K. Cho, "Convolutional Recurrent Neural Networks for Music Classification," *arXiv preprint arXiv:1609.04243v3*, 2016.

[7] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. J. Weiss, and K. Wilson, "CNN architectures for large-scale audio classification," *arXiv preprint arXiv:1609.09430v2*, 2017.

[8] J. Cramer, H.-H. Wu, J. Salamon, and J. P. Bello, "Look, Listen, and Learn More: Design Choices for Deep Audio Embeddings," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

[9] Y. Gong, Y.-A. Chung, and J. Glass, "AST: Audio Spectrogram Transformer," *arXiv preprint arXiv:2104.01778v3*, 2021.

[10] M. Won, K. Choi, and X. Serra, "Semi-Supervised Music Tagging Transformer," *arXiv preprint arXiv:2111.13457v1*, 2021.

[11] K. Hiner, "Drum Classification," *https://github.com/khiner/DrumClassification*, 2023. [Online; accessed May 26, 2025.]

[12] M. Yun and J. Bi, "Deep Learning for Musical Instrument Recognition," Tech. Report, Department of Electrical and Computer Engineering, University of Rochester, 2018.

[13] R. Vogl, "Deep Learning Methods for Drum Transcription and Drum Pattern Generation," Doctoral thesis, Institute of Computational Perception, Johannes Kepler University Linz, Austria, November 2018.

[14] L. Géré, N. Audebert and P. Rigaux, "Improved Symbolic Drum Style Classification with Grammar-Based Hierarchical Representations," in *Proc. 25th Int. Society for Music Information Retrieval Conf. (ISMIR)*, San Francisco, CA, USA, 2024. *arXiv preprint arXiv:2407.17536v1*, 2024.

[15] J. Gillick, A. Roberts, J. Engel, D. Eck and D. Bamman, "Learning to Groove with Inverse Sequence Transformations," in *Proc. 36th Int. Conf. on Machine Learning (ICML)*, Long Beach, CA, USA, 2019. *arXiv preprint arXiv:1905.06118v2*, 2019.

[16] K. Choi, G. Fazekas, M. Sandler and K. Cho, "Transfer Learning for Music Classification and Regression Tasks," in *Proc. 18th Int. Society for Music Information Retrieval Conf. (ISMIR)*, Suzhou, China, 2017. *arXiv preprint arXiv:1703.09179v4*, 2017.

[17] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang and M. Plumbley, "PANNs: Large-Scale Pretrained Audio Neural Networks for Audio Pattern Recognition," *arXiv preprint arXiv:1912.10211v5*, 2020.

[18] B. Elizalde, S. Deshmukh, M. Al Ismail and H. Wang, "CLAP: Learning Audio Concepts from Natural Language Supervision," *arXiv preprint arXiv:2206.04769v1*, 2022.

[19] H. Park, Y. Chung and J.-H. Kim, "Deep Neural Networks-based Classification Methodologies of Speech, Audio and Music, and its Integration for Audio Metadata Tagging," *Journal of Web Engineering*, vol. 22, no. 1, pp. 1–26, April 2023.

[20] T. Morocutti, F. Schmid, K. Koutini and G. Widmer, "Device-Robust Acoustic Scene Classification via Impulse Response Augmentation," *arXiv preprint arXiv:2305.07499v2*, 2023.

[21] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal and M. Ritter, "Audio Set: An ontology and human-labeled dataset for audio events," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 776–780.

[22] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles and H. Jegou, "Training data-efficient image transformers & distillation through attention," *arXiv preprint arXiv:2012.12877v2*, 2021.

[23] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An Image Is Worth 16X16 Words: Transformers For Image Recognition At Scale," *arXiv preprint arXiv:2010.11929v2*, 2021.

[24] K. Koutini, "PaSST," https://github.com/kkoutini/PaSST, 2023. [Online; accessed May 28, 2025.]

[25] K. Koutini, "PaSSThear21," https://github.com/kkoutini/passt$_h$$ear$21, 2023.[$Online$; $accessedMay$28

[26] A. Gazneli, G. Zimmerman, T. Ridnik, G. Sharir, and A. Noy, "End-to-End Audio Strikes Back: Boosting Augmentations Towards An Efficient Audio Classification Network," *arXiv preprint arXiv:2204.11479v5*, 2022.

[27] I. Jordal, "audiomentations," https://github.com/iver56/audiomentations, 2025. [Online; accessed June 1, 2025.]

[28] T. Eriksen, "LUMT-Thesis-Resources," https://drive.google.com/drive/folders/127Oq1qeRvnjP1QwH
cCmi8A6PvQg?usp=drive$_l$$ink$, 2025.[$Online$; $accessedJune$1, 2025.]

11 Appendices

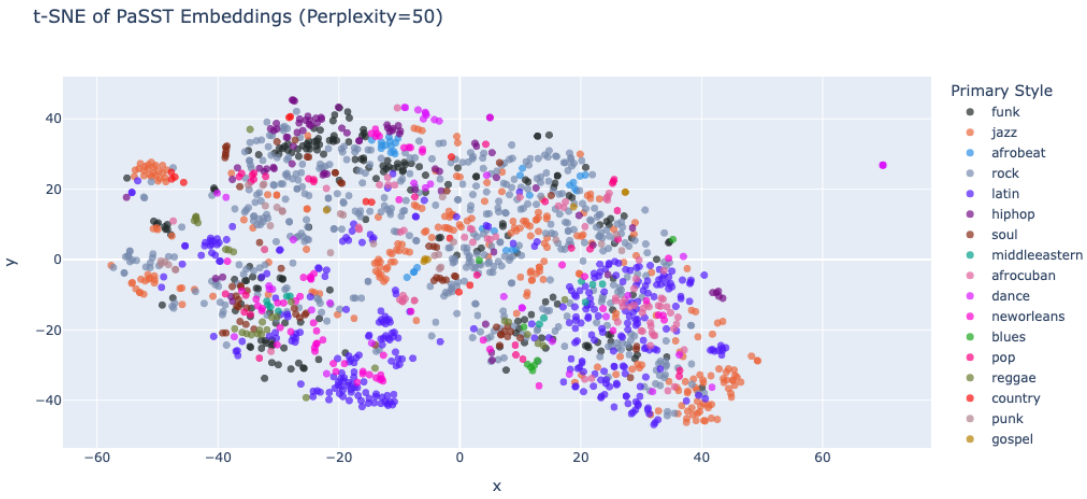

Figure 3: t-SNE for the best PaSST model by primary style.

Figure 4: t-SNE for the best GMD CNN model by primary style.

Table 5: F1 scores for all experiments (highest per round in **bold**).

Round	ID	Model	F1	Notebook	Description
1	1.1	CNN	**0.9204**	GMD_CNN_prototype3.ipynb	Defaults with rock styles
1	1.2	PaSST	0.8544	PaSST_setup2.ipynb	Defaults with rock styles
2	2.1	PaSST	0.8660	PaSST_setup3.ipynb	PaSST config
2	2.2	PaSST	**0.8699**	PaSST_setup4.ipynb	PaSST config
3	3.1	CNN	0.3267	GMD_CNN_prototype4.ipynb	GMD-mini
3	3.2	PaSST	**0.3911**	PaSST_setup5.ipynb	GMD-mini
4	4.1	CNN	**0.7531**	GMD_CNN_prototype5.ipynb	GMD-full
4	4.2	PaSST	0.7367	PaSST_setup6.ipynb	GMD-full
5	5.1	PaSST	0.7269	PaSST_setup6_2.ipynb	GMD-full repeat
5	5.2	PaSST	0.7313	PaSST_setup6_3.ipynb	GMD-full repeat
5	5.3	CNN	0.7646	GMD_CNN_prototype5_2.ipynb	GMD-full repeat
5	5.4	CNN	**0.7827**	GMD_CNN_prototype5_3.ipynb	GMD-full repeat
6	6.1	PaSST	**0.7247**	PaSST_setup6_4.ipynb	t-SNE & Time patchout
6	6.2	PaSST	0.3861	PaSST_setup5_2.ipynb	t-SNE & Time patchout, GMD-mini
7	7.1	PaSST	**0.8582**	PaSST_setup6_5.ipynb	PaSST MLP 3 Layers
8	8.1	PaSST	0.8604	PaSST_setup6_6.ipynb	PaSST MLP 5 Layers
8	8.2	PaSST	0.7964	PaSST_setup6_7.ipynb	PaSST MLP 7 Layers
8	8.3	PaSST	**0.8659**	PaSST_setup6_8.ipynb	PaSST MLP 4 Layers
8	8.4	PaSST	0.8352	PaSST_setup6_9.ipynb	PaSST MLP 2 Layers
8	8.5	PaSST	0.8424	PaSST_setup6_10.ipynb	PaSST MLP 6 Layers
8	8.6	PaSST	0.8577	PaSST_setup6_11.ipynb	MLP 4 Layers, Symmetric Bottleneck
8	8.7	PaSST	0.8604	PaSST_setup6_12.ipynb	MLP 4 Layers, Progressive Bottleneck
9	9.1	CNN	0.8779	GMD_CNN_prototype6.ipynb	CNN 5 Conv Layers
9	9.2	CNN	**0.8944**	GMD_CNN_prototype6_2.ipynb	CNN 7 Conv Layers
9	9.3	CNN	0.8900	GMD_CNN_prototype6_3.ipynb	CNN 9 Conv Layers
10	10.1	CNN	**0.9080**	GMD_CNN_prototype6_4.ipynb	CNN 7 Conv Layers + GaussianNoise, RoomSim
10	10.2	PaSST	0.7953	PaSST_setup6_13.ipynb	MLP 4 Layers + GaussianNoise, RoomSim
10	10.3	PaSST	0.8446	PaSST_setup6_14.ipynb	MLP 4 Layers + TimeStretch
10	10.4	CNN	0.8632	GMD_CNN_prototype6_5.ipynb	CNN 7 Conv Layers + TimeStretch
11	11.1	PaSST	0.8429	PaSST_setup6_15.ipynb	PaSST MLP 4 Layers, Circular padding
11	11.2	PaSST	**0.8752**	PaSST_setup6_16.ipynb	PaSST MLP 4 Layers, Reflection padding
11	11.3	CNN	0.8747	GMD_CNN_prototype6_6.ipynb	CNN 7 Conv Layers, GaussianNoise, RoomSimulator, Reflection padding
11	11.4	CNN	0.8736	GMD_CNN_prototype6_7.ipynb	CNN 7 Conv Layers, Reflection padding
11	11.5	CNN	0.8747	GMD_CNN_prototype6_8.ipynb	CNN 7 Conv Layers, Circular padding

Table 6: Glossary of Terms

Acronym	**Definition**
MIR	Music Information Retrieval
GMD	Groove MIDI Dataset
E-GMD	Expanded Groove MIDI Dataset
PaSST-L	Patchout Audio Spectrogram Transformer – Light
CNN	Convolutional Neural Network
MLP	Multilayer Perceptron
AST	Audio Spectrogram Transformer
ViT	Vision Transformer
DeiT	Data Efficient Image Transformer
LSTM	Long Short-Term Memory
ADT	Automatic Drum Transcription
MIDI	Musical Instrument Digital Interface
PANN	Pretrained Audio Neural Network
CLAP	Contrastive Language Audio Pretraining
STFT	Short-Time Fourier Transform
NFFT	Non-Uniform Fast Fourier Transform
ReLU	Rectified Linear Unit
LayerNORM	Layer Normalization
GPU	Graphics Processing Unit
t-SNE	t-distributed stochastic neighbor embedding
VGG	Visual Geometry Group
OpenL3	Open-source deep audio and image embeddings
OpenMIC	Instrument recognition dataset
CLS Embedding	Classification embedding
F1	Accuracy metric
mAP	Mean average precision
AUC	Area under the curve
ESC-50	Dataset for Environmental Sound Classification
AudioSet	A large-scale dataset of manually annotated audio events
LLM	Large Language Model
SOTA	State of the art
DEMUCS	Hybrid Spectrogram and Waveform Source Separation
BPM	Beats per minute
SpecAugment	Spectrogram augmentation technique