Waveform based Audio Classification of Environmental Sounds Using Deep Learning

Bhanu Gupta(bvg9468),
Pranav Narayan(pn2229)
Purvav Punyani(psp8474)
New York University Tandon School of Engineering

Abstract

This research paper explores the effectiveness of waveform-based inputs for audio classification compared to spectrogram-based inputs. Using the UrbanSound8k dataset and deep learning techniques, we evaluate both approaches' accuracy, computational efficiency, and interpretability. Our findings show that waveform-based audio classification achieves an accuracy of 80%, slightly lower than the 84% accuracy achieved with spectrogram-based inputs from the literature survey. However, waveform-based inputs offer advantages such as preserving temporal information and reducing preprocessing steps, making them a promising alternative in scenarios where precise timing and temporal dynamics are crucial. This study contributes to understanding audio classification techniques and provides insights for researchers and practitioners interested in utilizing waveform-based inputs for audio analysis.

GitHub Repo: https://github.com/Purvav0511/Waveform-based-Audio-Classification-of-Environmental-Sounds-Using-Deep-Learning/tree/main

Introduction

Audio classification plays a crucial role in various fields, ranging from speech recognition and music genre classification to environmental sound analysis. With the advancements in deep learning techniques, particularly convolutional neural networks (CNNs), audio classification has significantly improved accuracy and performance. This study focuses on environmental sound classification, which involves identifying and categorizing sounds from the surrounding environment. To tackle this challenge, we leverage the power of deep learning by employing CNNs for audio classification. Specifically, we investigate the effectiveness of two approaches: classifying audio using raw waveforms and spectrograms. In this paper, we compare our findings with the work presented in the article "Audio Deep Learning Made Simple: Sound Classification Step-by-Step," published in Towards Data Science, where the authors have also employed spectrograms for sound classification. By exploring these two methods and comparing our results with existing approaches, we aim to contribute to the advancements in environmental sound classification using convolutional neural networks. The UrbanSound8k dataset serves as the foundation for our research, providing a comprehensive and relevant collection of environmental sound samples for training and evaluation.

Advantages of Waveform inputs over Spectrograms

- **Retaining temporal information:** Waveforms capture the raw audio data in its temporal domain, preserving the timing and sequence of sound events. This can be beneficial for tasks that rely on precise timing information, such as audio event detection or speech recognitionn.
- **Simplified preprocessing**: Waveform inputs require minimal preprocessing compared to spectrograms. Converting audio signals to spectrograms involves additional steps, such as applying Fourier transforms and selecting appropriate frequency bins. You can directly use the raw audio data with waveform inputs without additional transformations.
- **Efficient representation:** Waveform inputs have a lower dimensionality compared to spectrograms. Spectrograms typically have a 2D representation with frequency bins and time frames, resulting in higher memory requirements and computational complexity. Waveform inputs, on the other hand, are 1D signals, requiring less memory and computational resources.

Literature Survey

In recent years, there has been significant research interest in environmental sound classification using Convolutional Neural Networks (CNNs) and deep learning techniques. Zhang et al. provide a comprehensive guide in their paper titled "Audio Deep Learning Made Simple: Sound Classification Step-by-Step" [1]. They outline a step-by-step approach for sound classification, focusing on spectrograms as input for CNNs. In their study, Salamon et al. investigate the effectiveness of deep CNNs and data augmentation techniques for environmental sound classification [2]. They explore the combination of raw waveform inputs with data augmentation to improve model performance. Furthermore, Zhu et al. compare different architectural variations of CNNs for environmental sound classification and emphasize the importance of model selection and parameter tuning in their

work [3]. These studies highlight the significance of using spectrograms and raw waveforms as inputs for CNN-based audio classification models. Additionally, the availability of large-scale human-labeled datasets, such as the Audio Set dataset, introduced by Gemmeke et al. [4], provides valuable resources for training and evaluating these models.

Methodology

This research employs a deep learning model tailored for audio classification tasks, taking advantage of the extensive UrbanSound8K dataset. The UrbanSound8K dataset is a collection of 8732 labeled sound excerpts, each with a duration of up to 4 seconds, that represent a variety of urban sounds. The dataset is comprehensive, comprising ten distinct classes of sounds: air conditioner, car horn, children playing, dog bark, drilling, engine idling, gun shot, jackhammer, siren, and street music. Our methodology involves several stages, including data preprocessing, dataset creation and loading, model definition, training and validation, evaluation, and prediction. In the ensuing subsections, we elaborate on each of these stages, detailing the techniques and principles applied at every step of the way.

Here's a brief overview:

1. **Data Preprocessing**: It loads and preprocesses the audio data. The preprocessing includes downmixing the audio to mono, resampling it to a target sample rate, and padding or trimming it to a fixed length. The preprocessed audio data is then transformed into spectrogram representation or waveform.
2. **Dataset and DataLoader**: The preprocessed audio data is encapsulated into a PyTorch Dataset, and a DataLoader is created to efficiently feed the data into the model during training.
3. **Model Definition**: The AudioToVisualModel is defined, which is a Convolutional Recurrent Neural Network (CRNN). This model first uses a Convolutional Neural Network (CNN) to extract high-level features from the audio data, then feeds these features into a Recurrent Neural Network (RNN) to capture temporal dependencies, and finally classifies the audio using a fully connected layer.
4. **Training and Validation**: The model is trained for a certain number of epochs, using the Adam optimizer and Cross-Entropy loss function. After each epoch, the model is validated on the validation set.
5. **Testing and Evaluation**: The trained model is evaluated on the test set, and the test accuracy is reported. A confusion matrix is plotted to visualize the model's performance across different classes.
6. **Prediction**: The trained model is used to predict the class of a single audio file.

In terms of the architecture, the CRNN model is quite effective for audio classification tasks. The CNN layers are able to extract high-level features from the spectrogram or waveform of the audio, and the RNN layers (either LSTM or GRU as specified) can capture the temporal dependencies in these features. The final fully connected layer then maps these features to the class probabilities.

The number of layers, their sizes, the types of RNN used, and other hyperparameters are all configurable and would likely need to be tuned based on the specific task and dataset.

Model Architecture

The model architecture, defined by the AudioToVisualModel class, consists of several convolutional layers (conv1, conv2, conv3, conv4) for feature extraction, recurrent layers (rnn) for capturing temporal dependencies, and a fully connected layer (fc1) for classification. It takes audio waveforms as input and produces class predictions as output.

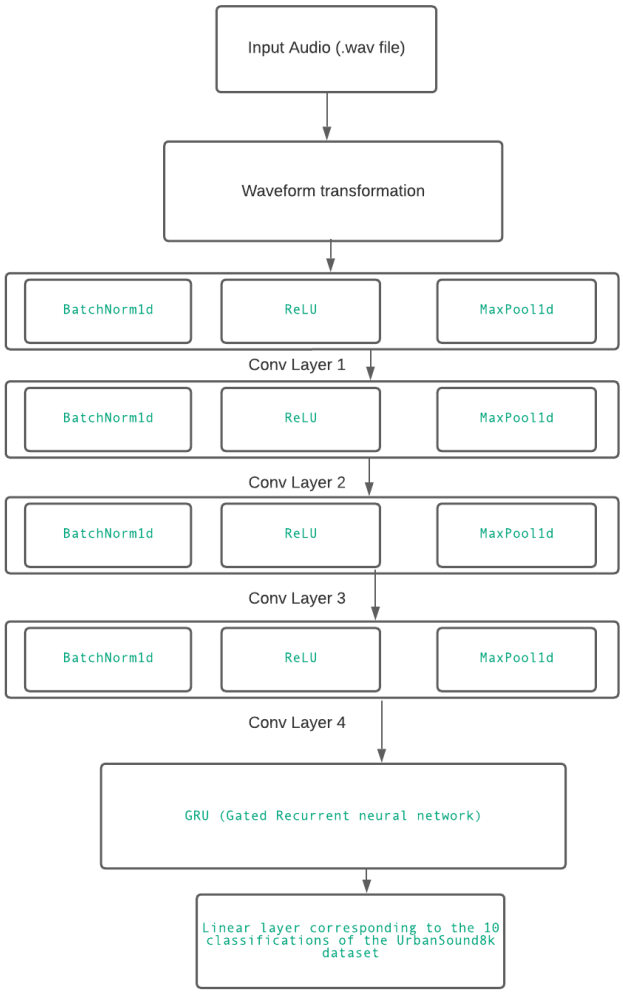

Figure 1: Model Architecture

Dataset and Training Parameters

- **Dataset:** The UrbanSound8k dataset is a widely used benchmark dataset for audio classification tasks. It consists of 8,732 labeled sound excerpts from urban environments, spanning ten different classes. These classes

represent various urban sound categories such as car horn, siren, street music, and drilling, among others. The dataset offers a diverse range of acoustic events captured in real-world urban settings, making it an ideal resource for training and evaluating audio classification models. Waveform representation shown in **Figure 2** and the Spectrogram representation shown in **Figure 3**

- **Loss Function:** The model utilizes the nn.CrossEntropyLoss() function, which computes the cross-entropy loss between the predicted class probabilities and the true labels. This loss function is well-suited for multi-class classification tasks.
- **Validation Procedure:** The validate_epoch function performs validation after each training epoch. It sets the model in evaluation mode, iterates over the validation data batches, performs a forward pass to obtain predictions, calculates the loss, and computes the accuracy by comparing the predicted labels with the true labels. The average loss and accuracy for the validation dataset are computed by dividing the accumulated loss and correct predictions by the total number of samples in the validation dataset. The chosen hyperparameters for the model include a learning rate (lr) of 0.001 for the Adam optimizer (optim.Adam), several hidden units (rnn_hidden_size) set to 512 in the recurrent layers, and several recurrent layers (rnn_num_layers) set to 2. These hyperparameters can be adjusted based on the specific requirements and characteristics of the dataset.
- **Training Procedure:** The train_epoch function performs one epoch of training. It sets the model in training mode, iterates over the batches of the training data, performs a forward pass to obtain predictions, and calculates the cross-entropy loss using nn.CrossEntropyLoss(), performs backpropagation and updates the model parameters using the Adam optimizer. The average loss for the epoch is computed by dividing the accumulated loss by the total number of samples in the training dataset.
- **Validation Procedure:** The validate_epoch function performs validation after each training epoch. It sets the model in evaluation mode, iterates over the validation data batches, performs a forward pass to obtain predictions, calculates the loss, and computes the accuracy by comparing the predicted labels with the true labels. The average loss and accuracy for the validation dataset are computed by dividing the accumulated loss and correct predictions by the total number of samples in the validation dataset.

Results

The results of our Deep Learning model are presented in this section. We evaluated the performance of our model using a variety of metrics, including a confusion matrix, training validation curve, and accuracy scores for training, validation, and testing data.

Confusion Matrix

The training validation curve provides insight into the model's learning progression over the training period. Our model demonstrates an ideal trend, with the training error decreasing and the validation error reaching a minimum point, beyond which it starts to increase, suggesting the optimal stopping point to prevent overfitting. Figure 4 depicts this curve and the point of divergence, which indicates the onset of overfitting.

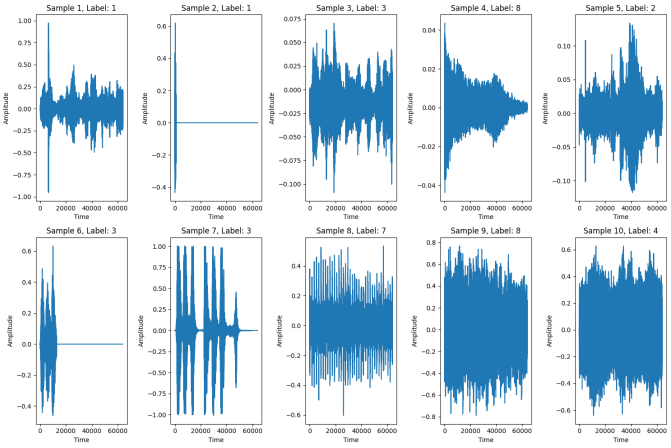

Figure 2: Waveform.

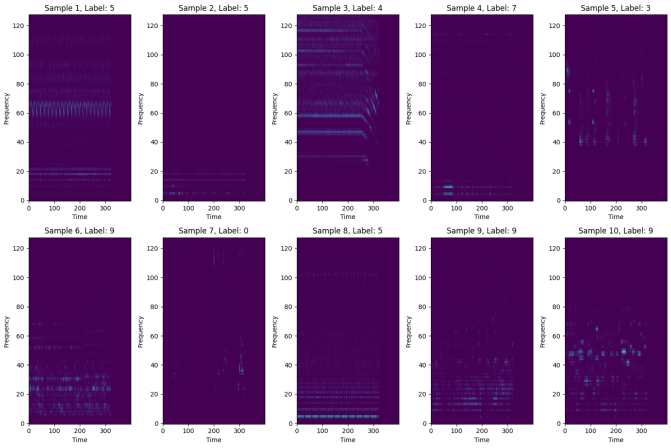

Figure 3: Spectrogram

Training Validation Curve

The training validation curve provides insight into the model's learning progression over the training period. Our model demonstrates an ideal trend, with the training error decreasing and the validation error reaching a minimum point, beyond which it starts to increase, suggesting the optimal stopping point to prevent overfitting. Figure 5 depicts this curve and the point of divergence, which indicates the onset of overfitting.

Accuracy Scores

Our model achieved a validation accuracy of and test accuracy of 79.62%. These results are competitive, indicating that our model has learned well from the training dataset,

generalized reasonably well to the unseen validation dataset, and performed robustly on the test dataset.

The test accuracy is particularly noteworthy as it provides a reliable measure of how the model will likely perform on unseen, real-world data. With an accuracy of 79.62%, our model demonstrates a robust predictive capability, showcasing its potential utility in practical applications.

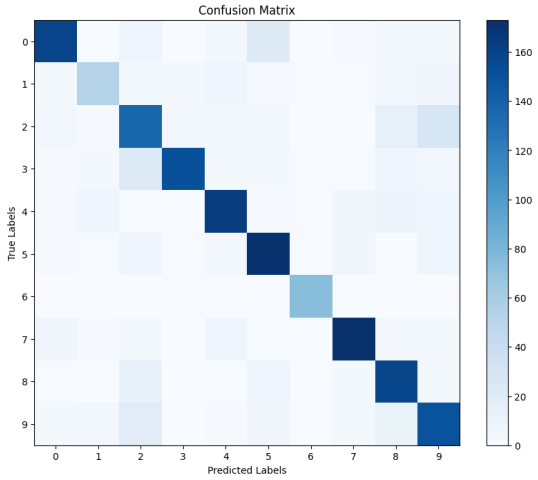

Figure 4: Confusion Matrix

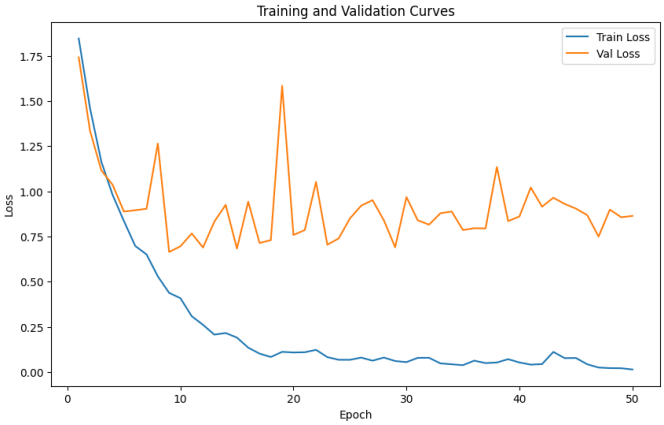

Figure 5: Training and Validation Curve

Future Scope

The findings of this research suggest several possible directions for future work. First, the techniques used in this study could be applied to larger and more diverse audio datasets. While the UrbanSound8k dataset provided a solid foundation for this research, it would be interesting to see how waveform-based audio classification performs on larger and more varied audio data.

Secondly, further research could focus on improving the accuracy of waveform-based audio classification. Although the accuracy achieved in this study was slightly lower than that of spectrogram-based inputs, there may be opportunities to enhance this with more advanced deep learning models or refined preprocessing techniques.

Finally, exploring the combination of waveform and spectrogram-based inputs could be an interesting avenue for future research. Such a hybrid approach could potentially leverage the strengths of both techniques, enhancing overall classification performance.

Conclusion

We initially aimed to create a model that classifies audio and generates Waveform and spectrogram for two data sets ESC 50 and Urbansound8k. Our final work mainly focuses on using Waveform-based inputs over the spectrogram-based inputs used in our reference papers. We have implemented a novel classifier using Convolutional Neural Networks and GRU. Our final goal was to classify based on Waveforms over Spectrograms and detail the advantages of doing so, as we have detailed in earlier sections.

Acknowledgments

I would like to express my sincere gratitude to Prof. Chinmay Hegde, the teaching assistants and the peers who helped review and critique the various components and provided valuable feedback and insightful suggestions during the review process of this paper. Their expertise and thoughtful critique greatly improved the overall quality of this research.

References

1. Z. Zhang et al., ”Audio Deep Learning Made Simple: Sound Classification Step-by-Step,” 2018.
2. J. Salamon et al., ”Deep Convolutional Neural Networks and Data Augmentation for Environmental Sound Classification,” 2014.
3. L. Zhu et al., ”Environmental Sound Classification with Convolutional Neural Networks,” 2019.
4. J. F. Gemmeke et al., ”Audio Set: An Ontology and Human-Labeled Dataset for Audio Events,” 2017.
5. https://towardsdatascience.com/audio-deep-learning-made-simple-sound-classification-step-by-step-cebc936bbe5
6. ChatGPT