Audio Classification on ESC-50

Submitted by

董更上 (12110615)
魏悦阳 (12111641)
杨至立 (12112711)
吴宛幸 (12112245)

Statistical Learning

January 2024

CONTENTS

1 Introduction 1

1.1 Summary 1

1.2 Dataset Esc50 Introduction 1

1.3 Feature Extraction 1

(i) MFCCs (Mel Frequency Cepstral Coefficients) 2

(ii) ZCR (Zero-Crossing Rate) 3

1.4 Human Recognition of the ESC10 and ESC50 Datasets . . . 4

2 Implementation Method 5

2.1 An Initial Exploration 5

2.2 Application of Traditional Statistical Learning Methods . . 6

2.3 Exploration of Deep Learning Strategies 7

2.4 Utilizing Pre-trained Large Models: CLAP 8

3 FUTURE WORK 10

3.1 Enhanced Feature Engineering 10

3.2 Multimodal Data Integration 10

3.3 Leveraging Transfer Learning 10

3.4 Ensemble Techniques 10

3.5 Interpretability 10

1 INTRODUCTION

1.1 Summary

The main task of our project is audio classification, where we use the ESC-50 dataset. It consists of 50 categories, each with 40 audio samples of 5 seconds in length. Our goal is to classify these audio samples. These audio samples cover a wide range of sounds, from natural environmental sounds (like rain, ocean waves) to man-made sounds (like helicopters, chainsaws). After figuring out how to extract audio features, we first tried to use a small neural network for training, and then tried to use PCA to reduce the dimensionality of the features. After that, we use K-fold Cross Validation to classify ESC-10 with KNN, Random Forest and SVM, and find that the accuracy of ESC-10 is more than 60%, but the accuracy of ESC-50 is only 30%. After that, we used deep learning to build a deeper neural network, and the classification accuracy was improved to more than 60%. Finally, by using CLAP (a structure similar to CLIP, but for audio processing), the classification accuracy reached 93.85%. All files used has been uploaded into our project Github repository.

1.2 Dataset Esc50 Introduction

The ESC-50 dataset is a labeled collection of 2000 environmental audio recordings suitable for benchmarking methods of environmental sound classification. The dataset consists of 5-second-long recordings organized into 50 semantical classes (with 40 examples per class) loosely arranged into 5 major categories:

Animals	Natural soundscapes & water sounds	Human, non-speech sounds	Interior/domestic sounds	Exterior/urban noises
Dog	Rain	Crying baby	Door knock	Helicopter
Rooster	Sea waves	Sneezing	Mouse click	Chainsaw
Pig	Crackling fire	Clapping	Keyboard typing	Siren
Cow	Crickets	Breathing	Door, wood creaks	Car horn
Frog	Chirping birds	Coughing	Can opening	Engine
Cat	Water drops	Footsteps	Washing machine	Train
Hen	Wind	Laughing	Vacuum cleaner	Church bells
Insects (flying)	Pouring water	Brushing teeth	Clock alarm	Airplane
Sheep	Toilet flush	Snoring	Clock tick	Fireworks
Crow	Thunderstorm	Drinking, sipping	Glass breaking	Hand saw

1.3 Feature Extraction

The first step is to extract features from the audio. We selected various features, including Mel Frequency Cepstral Coefficients (MFCCs) and zero-crossing

rate(ZCR), to comprehensively capture the characteristics of the audio. We used the librosa library, a Python package for audio and music analysis, which provides a variety of audio feature extraction functions.

(i) MFCCs (Mel Frequency Cepstral Coefficients)

- Mel Frequency
  The Mel frequency scale is a critical concept in audio processing, as it aligns frequency measurements with the human ear's perception of pitch. This scale more accurately reflects human auditory characteristics than a linear frequency scale. In the context of Mel Frequency Cepstral Coefficients (MFCCs), this scale is instrumental in converting the normal frequency spectrum of a sound signal into the Mel frequency spectrum. This conversion is a crucial step, as it lays the foundation for the subsequent cepstral analysis.

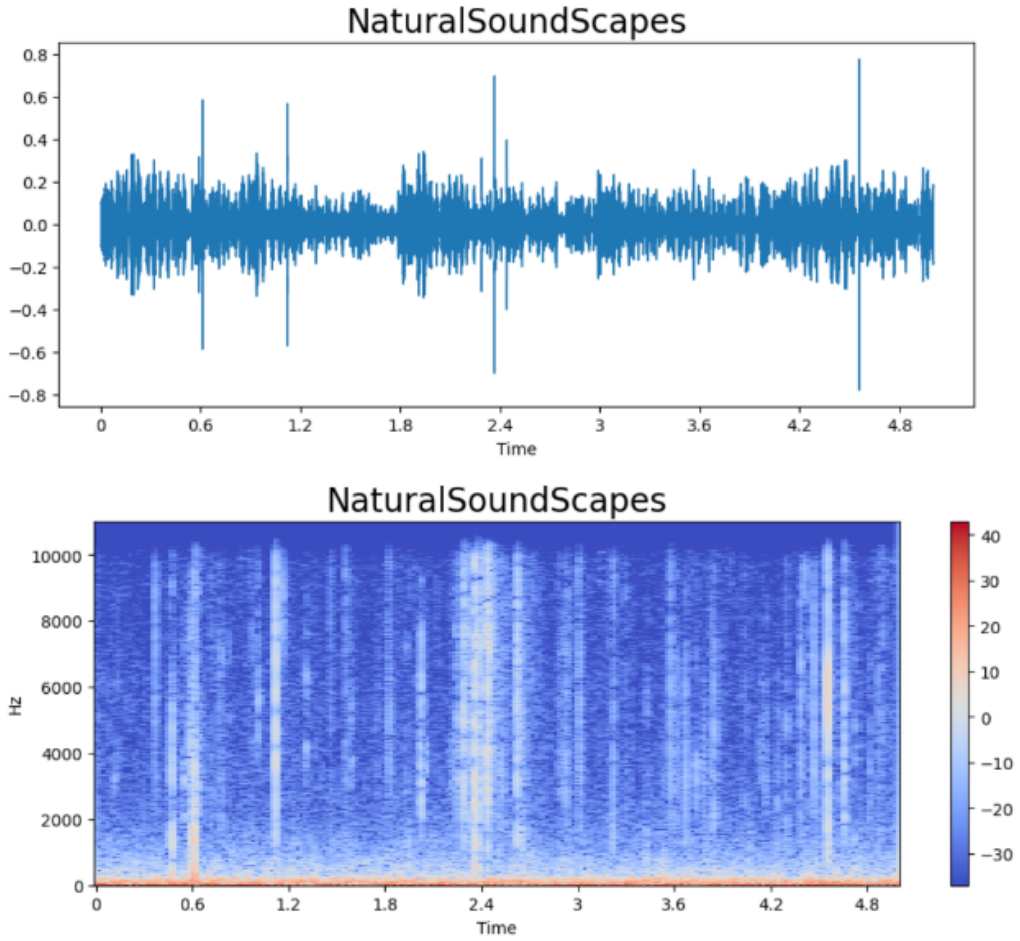

Figure 1: Waveshow and Spectrogram

- Cepstral Transform

The Cepstral Transform is an essential part of the MFCC extraction process. After mapping the sound signal onto the Mel frequency scale, the next step involves applying the cepstral transform to this Mel-scaled spectrum. This transform includes a logarithmic operation on the Mel spectrum, followed by a Discrete Cosine Transform (DCT). The purpose of this transformation is to convert the spectral information into a form that more effectively represents the characteristics of the sound as perceived by human listeners. The cepstral representation is particularly adept at isolating and emphasizing the key frequency characteristics of the signal, making it invaluable for speech and music analysis.

- Mel Frequency Cepstral Coefficients
  After the application of the Cepstral Transform to the Mel-scaled spectrum, the resulting coefficients (MFCCs) encapsulate critical aspects of the sound signal. Each MFCC corresponds to a specific aspect of the sound's spectral characteristics. For example, $MFCC_0$ (the zeroth coefficient) is indicative of the overall energy of the speech signal, akin to volume or intensity. Conversely, $MFCC_1$ (the first coefficient) captures the spectrum's shape, which correlates with the perception of pitch. Typically, the first few coefficients, such as the first 13, contain the majority of the pertinent information for speech and audio analysis. The higher-order coefficients, though less critical, can provide insights into more nuanced features of the signal.

(ii) ZCR (Zero-Crossing Rate)

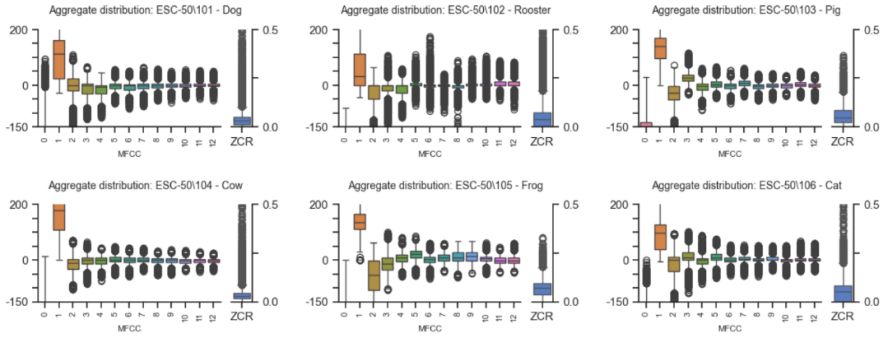

Figure 2: MFCCs and ZCR of audios

Zero-Crossing Rate is the rate at which an audio signal changes from positive to negative or back. It counts how many times the signal crosses the zero amplitude axis within a specific time frame. In audio processing, ZCR is a simple yet effective way to analyze certain characteristics of a sound wave.

It's especially useful for distinguishing different types of sounds. Generally, a

higher zero-crossing rate is associated with complex sounds like certain types of music, while a lower rate often indicates simpler sounds or less complex noise.

1.4 Human Recognition of the ESC10 and ESC50 Datasets

Prior to the application of diverse statistical learning techniques for audio classification, an initial analysis of human recognition and categorization of these sounds, derived from the ESC10/50 datasets, is conducted. On the ESC10 dataset, featuring merely 10 sound categories, human performance is commendably high. As

Sounds	Baby cry	Chainsaw	Clock tick	Dog bark	Fire crackling	Helico-pter	Person sneeze	Rain	Rooster	Sea waves
Baby cry	100.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Chainsaw	0.00%	98.30%	0.00%	0.00%	0.00%	1.50%	0.00%	0.00%	0.20%	0.00%
Clock tick	0.00%	0.00%	99.70%	0.00%	0.00%	0.00%	0.00%	0.30%	0.00%	0.00%
Dog bark	0.00%	0.00%	0.00%	99.80%	0.00%	0.00%	0.20%	0.00%	0.00%	0.00%
Fire crackling	0.00%	0.20%	0.70%	0.20%	87.40%	0.20%	0.00%	11.10%	0.00%	0.20%
Helicopter	0.00%	4.80%	0.00%	0.20%	0.40%	91.90%	0.00%	0.80%	0.00%	1.90%
Person sneeze	0.40%	0.00%	0.00%	0.00%	0.00%	0.00%	99.60%	0.00%	0.00%	0.00%
Rain	0.00%	0.60%	0.00%	0.00%	6.70%	0.60%	0.00%	89.70%	0.00%	2.40%
Rooster	0.00%	0.00%	0.00%	0.20%	0.00%	0.00%	0.00%	0.00%	99.80%	0.00%
Sea waves	0.00%	1.80%	0.00%	0.40%	0.00%	0.40%	0.00%	6.20%	0.00%	91.10%

Figure 3: Humman accuracy on ESC10

human listeners, we exhibit strong recognition performance on this dataset, achieving an impressive average accuracy rate of 95.7%. However, a closer examination of the confusion matrix reveals that our ability to recognize certain types of natural environmental sounds, specifically Fire crackling, Rain, Sea waves, and Helicopter, is somewhat less accurate. This observation suggests that while overall human recognition is highly effective, there are specific areas where discernment can be further refined and improved. However, facing the task of differentiating between 50 distinct categories presents a much greater challenge, as the complexity and the likelihood of misidentification increases substantially with the expanded variety of sound types. This scenario significantly tests the limits of human auditory classification capabilities.

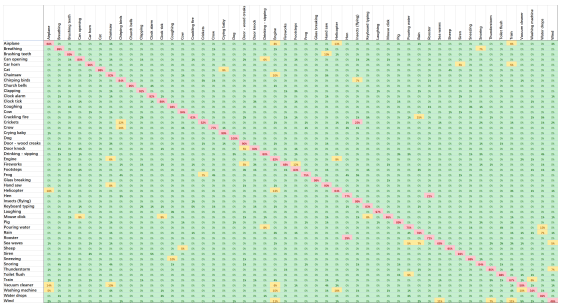

Figure 4: Human accuracy on ESC50

Brushing teeth	Can opening	Car horn	Cat	Chainsaw	Chirping birds	Church bells	Clapping
89.30%	80.50%	89.90%	87.50%	82.70%	84.00%	95.20%	91.90%
Coughing	Cow	Crackling fire	Crickets	Crow	Crying baby	Dog	Door - wood
93.50%	94.10%	63.40%	51.80%	76.60%	98.70%	100.00%	90.00%
Engine	Fireworks	Footsteps	Frog	Glass breaking	Hand saw	Helicopter	Hen
81.70%	68.00%	83.10%	75.30%	98.70%	90.00%	63.90%	76.90%
Laughing	Mouse click	Pig	Pouring water	Rain	Rooster	Sea waves	Sheep
97.30%	65.00%	88.60%	74.70%	77.60%	71.20%	68.00%	94.90%
Snoring	Thunderst orm	Toilet flush	Train	Vacuum cleaner	Washing machine	Water drops	Wind
84.20%	84.90%	87.70%	66.70%	57.70%	34.20%	92.00%	45.80%

Figure 5: Human recall on ESC50

The recall matrix for human recognition of ESC50 dataset shows varying levels of accuracy across different sound categories. High recall rates, like in "Dog"

and "Glass breaking," suggest these sounds are easily and consistently identified by listeners. Categories with lower recall rates, such as "Vacuum cleaner" and "Wind," indicate these sounds are more challenging to recognize correctly, possibly due to their less distinct nature or greater similarity to other sounds.

2 IMPLEMENTATION METHOD

2.1 An Initial Exploration

Layer (type)	Output Shape	Param #
lstm_1 (LSTM)	(None, 256)	264192
dropout_3 (Dropout)	(None, 256)	0
dense_3 (Dense)	(None, 64)	16448
dropout_4 (Dropout)	(None, 64)	0
dense_4 (Dense)	(None, 32)	2080
dropout_5 (Dropout)	(None, 32)	0
dense_5 (Dense)	(None, 5)	165

Total params: 282885 (1.08 MB)
Trainable params: 282885 (1.08 MB)
Non-trainable params: 0 (0.00 Byte)

Figure 6: The little model with 7 layers

Considering that there are some invalid sounds at the beginning and end of the audio, we cut the audio from 0.5 to 4.5 seconds. First, extracted 40 MFCC coefficients and the zero-crossing rate of the audio using the librosa library in Python. Then each audio file corresponds to 41 features. Next, for the first trail, we built a simple neural network model consists of 1 LSTM layer, 3 dropout layers and 3 dense layer. By allocating 80% of the dataset for training purposes, we attained a classification accuracy rate of 53%.

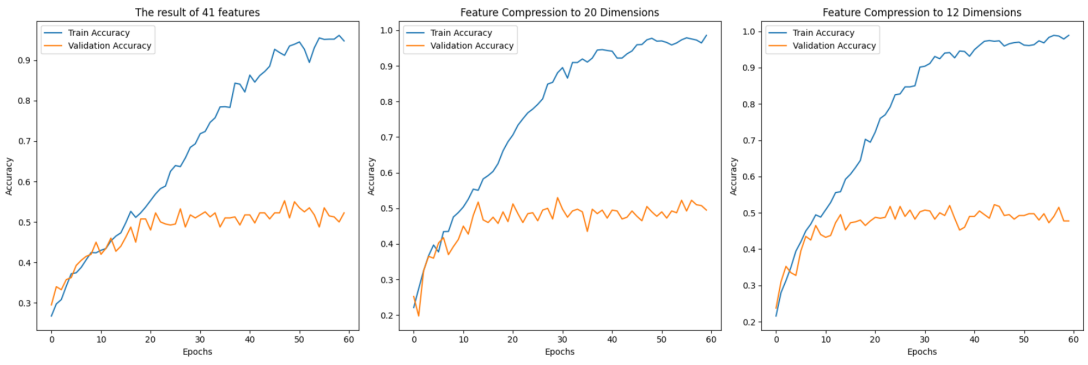

Figure 7: The results obtained by full features, compressing the feature to 20 dimensions and compressing the feature to 12 dimensions

This is a decent result, but since there are 41 features, it takes a relatively long time to compute. In order to shorten the training time, we decided to use PCA (Principal Component Analysis) to reduce the dimensionality of the features, thus reducing computation time. First, we use PCA to reduce the features to 20 dimensions, and then use the same method and model to train the data. Although the training time decreased, the accuracy dropped(50%). To verify the results, we reduced the feature dimensionality to 12 again and employed the same model and training approach for consistency in evaluation. The accuracy obtained on the test set was 49%. This result once again proves that when using PCA to extract features, information will be lost. Considering the time cost and accuracy requirements, we finally gave up using PCA to reduce the dimensionality of features.

2.2 Application of Traditional Statistical Learning Methods

We applied statistical learning methods such as k-NN, Random Forest, and SVM to classify the ESC50 dataset, utilizing a five-fold cross-validation method to ensure the robustness of our model evaluations. This cross-validation technique helps mitigate the risk of model overfitting due to limited data and allows for a more reliable estimate of model accuracy by systematically rotating the validation set across the entire dataset. Each fold consists of a unique 20% segment of the data, ensuring each part is used for validation once. Through this approach, we sought to achieve a balance between model reliability and training comprehensiveness.

Classifier	Fold1	Fold2	Fold3	Fold4	Fold5	Average Accuracy
k-NN	29.20%	31.20%	32.50%	33.20%	27.70%	30.80%
Random Forest	40.70%	44.00%	42.50%	42.20%	40.00%	41.80%
SVM	36.20%	37.00%	36.00%	39.00%	38.50%	37.30%

Figure 8: Accuracy on ESC50

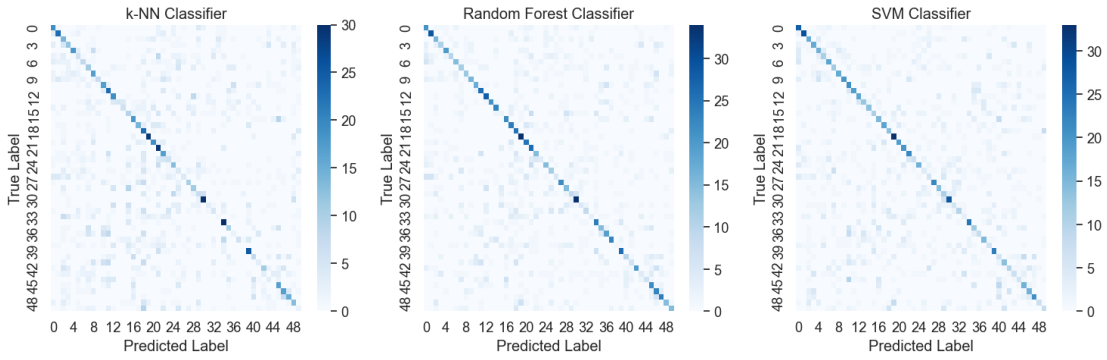

Figure 9: Confusion Matrices for k-NN, Random Forest, and SVM Classifiers on ESC50

In conclusion, our experimentation with the ESC50 dataset using k-NN, Random Forest, and SVM classifiers, as substantiated by five-fold cross-validation, indicates a varying degree of success. The Random Forest classifier outperforms the other two with a notable average accuracy, suggesting its better suitability for this task. However, the confusion matrices reveal room for improvement across all classifiers, particularly in correctly identifying specific sound categories. These insights point towards a need for further model refinement and exploration of alternative classification techniques to enhance performance.

2.3 Exploration of Deep Learning Strategies

```python
class ResidualBlock(nn.Module):
    def __init__(self, in_channels, out_channels, stride=1):
        super(ResidualBlock, self).__init__()
        self.conv1 = nn.Conv2d(in_channels, out_channels, kernel_size=3, stride=stride, padding=1)
        self.bn1 = nn.BatchNorm2d(out_channels)
        self.relu = nn.ReLU(inplace=True)
        self.conv2 = nn.Conv2d(out_channels, out_channels, kernel_size=3, stride=1, padding=1)
        self.bn2 = nn.BatchNorm2d(out_channels)
        self.downsample = nn.Sequential(
            nn.Conv2d(in_channels, out_channels, kernel_size=1, stride=stride),
            nn.BatchNorm2d(out_channels)
        ) if stride != 1 or in_channels != out_channels else None

    def forward(self, x):
        residual = x
        out = self.conv1(x)
        out = self.bn1(out)
        out = self.relu(out)
        out = self.conv2(out)
        out = self.bn2(out)
        if self.downsample is not None:
            residual = self.downsample(x)
        out += residual
        out = self.relu(out)
        return out
```

```python
class AudioCNN(nn.Module):
    def __init__(self, num_classes, conv1_channels=128, conv2_channels=256, fc1_out_features=256, num_residual_blocks=3):
        super(AudioCNN, self).__init__()
        self.conv1 = ResidualBlock(1, conv1_channels)
        self.conv2 = ResidualBlock(conv1_channels, conv2_channels)
        # Add more residual blocks
        self.res_blocks = nn.ModuleList([ResidualBlock(conv2_channels, conv2_channels) for _ in range(num_residual_blocks)])
        self.pool = nn.AdaptiveAvgPool2d((1, 1))
        self.fc1 = nn.Linear(conv2_channels, fc1_out_features)
        self.fc2 = nn.Linear(fc1_out_features, num_classes)

    def forward(self, x):
        x = self.conv1(x)
        x = self.conv2(x)
        for block in self.res_blocks:
            x = block(x)
        x = self.pool(x)
        x = x.view(x.size(0), -1)
        x = F.relu(self.fc1(x))
        x = self.fc2(x)
        return x
```

Figure 10: Residual block and AudioCNN

This model performs well on the ESC-50 dataset, primarily benefiting from the introduction of residual blocks and adaptive average pooling in its structure. Residual blocks effectively mitigate the vanishing gradient problem, aiding in deeper feature learning, while adaptive average pooling provides robustness to audio of different lengths[1]. The model's parameters are also flexible, allowing adaptation to various audio classification scenarios. Additionally, its clear feedforward process, involving the processing of Conv1 and Conv2, the introduction of multiple residual blocks, and the operations of fully connected layers, enables effective learning of complex audio features. In future work, further optimization of the model structure, adjustment of hyperparameters, and exploration of more advanced training techniques could be considered to enhance performance and achieve better audio classification results.

The model demonstrates excellent performance during training, reaching a final training loss of 0.1838. In the evaluation on the ESC-50 test set, the model achieves an accuracy of 63.5%, indicating its ability to classify audio samples to a certain extent. Such results are quite satisfactory for complex audio classification tasks. However, for further performance improvement, experimenting with adjustments to the model structure, hyperparameters, and employing more sophisticated training techniques could be explored.

The model underwent three training cycles, with the first cycle training for 35 epochs. The second and third cycles continued training based on the results of the first model. The graph below illustrates the training loss and accuracy for the first and third cycles.

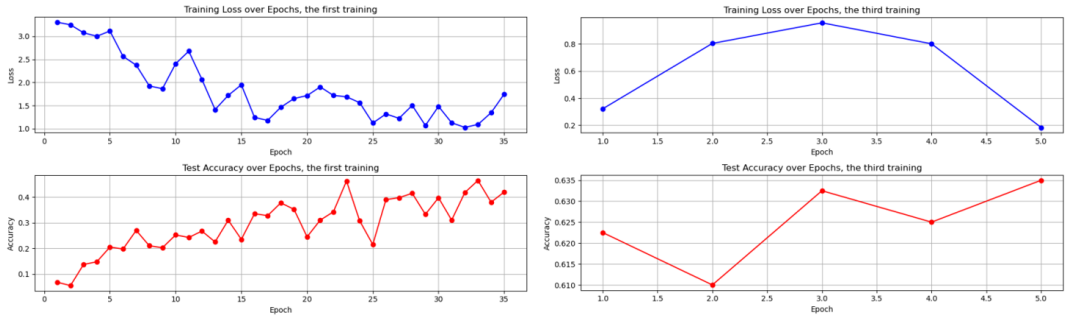

Figure 11: The first and third test result of loss and accuracy

2.4 Utilizing Pre-trained Large Models: CLAP

In addition to the aforementioned methods, we sought further improvements in accuracy. To achieve this, we explore the concepts of a pre-trainedCLAP(Contrastive Language-Audio Pretraining) model as described by Elizalde et al.[2] By conducting zero-shot inference, and ultimately achieving an accuracy of 93.85% on the ESC50 dataset.

The CLAP model, developed by Microsoft, is a multimodal large model that combines audio and text. Its core principle involves learning acoustic concepts from natural language supervision and achieving ”Zero-Shot” inference. The model shares similarities with the well-known CLIP model. During the training phase, Microsoft utilized 128,010 (120,000) audio-text pairs from four datasets for training. For text information processing, the BERT model was employed as an encoder to obtain text embeddings. In audio information processing, the CNN14 was used as an encoder to obtain audio embeddings.

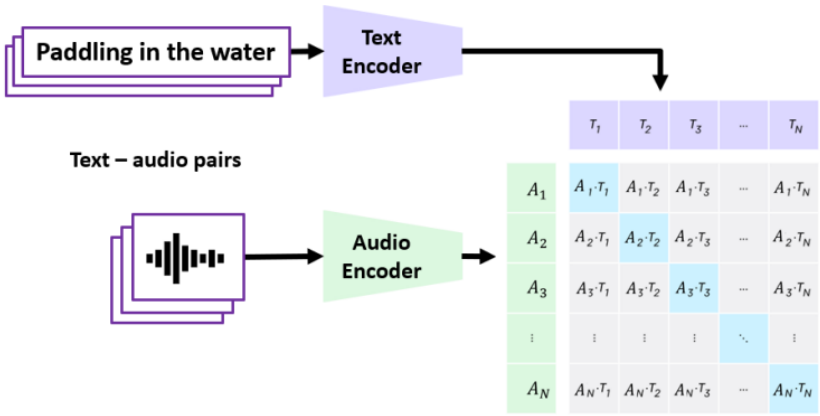

Figure 12: CLAP model

By multiplying the two embeddings, a feature matrix of size n×n was obtained, where diagonal elements represent positive samples, and the rest are negative samples. Contrastive learning was applied to compute cosine similarity as the training loss. This process yielded the final results.

We also conducted zero-shot classification using CLAP on the ESC50 dataset. The process involves inputting the dataset categories into the text encoder and the audio into the audio encoder. Cosine similarity is then calculated, and the result with the highest similarity is considered the predicted outcome.

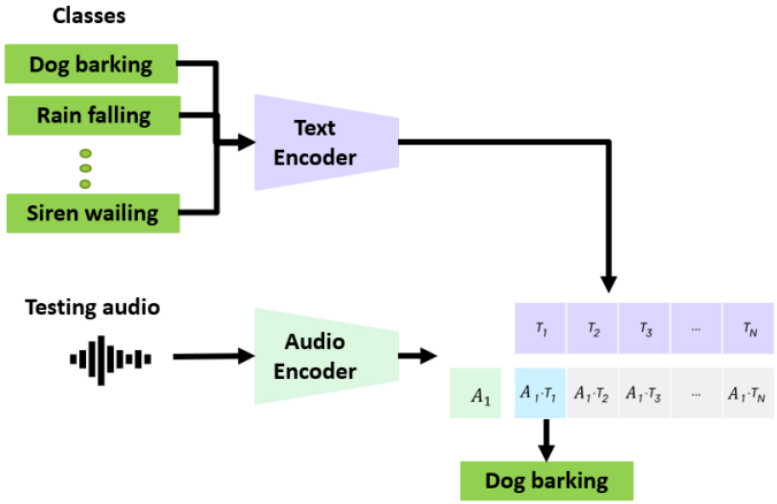

Figure 13: Zero-shot classification with the CLAP model

In terms of code implementation, we loaded the ESC50 dataset and the CLAP2023 model. We utilized the prompt "this is the sound of" as a standardized cue, encoding the 50 categories. The audio was processed using the audio encoder, and

after calculating the similarity, a softmax layer was applied to obtain probabilities and determine the final result.

```
Loading audio files
2000it [00:00, 16530.64it/s]
100%|██████████| 2000/2000 [07:23<00:00,  4.50it/s]
ESC50 Accuracy 0.9385
```

Figure 14: Accuracy for CLAP on ESC50

In the end, we achieved an accuracy of 0.9385. The reason CLAP can achieve, and even surpass, human-level recognition in the ESC50 environmental sound dataset is also explained in the paper. The training data for CLAP consists of audio-caption datasets, primarily encompassing descriptions of sound events, sound scenes, actions, and objects. The audio is uniformly truncated to a length of 5 seconds, similar to the content of the audio in ESC50.

3 FUTURE WORK

3.1 Enhanced Feature Engineering

Further investigation into audio feature extraction is warranted to identify features that could potentially improve classification accuracy. Exploration of deep learning-based automatic feature extraction could offer significant advancements.

3.2 Multimodal Data Integration

Integrating additional modalities such as contextual information or visual cues could provide a more holistic approach to audio classification.

3.3 Leveraging Transfer Learning

Expanding the use of transfer learning with different pre-trained models has the potential to significantly improve performance on the ESC50 dataset.

3.4 Ensemble Techniques

Combining predictions from multiple models through ensemble methods could improve classification accuracy and is a promising direction for research.

3.5 Interpretability

Finally, there is a need to focus on the interpretability of machine learning models used in audio classification to understand their decision-making process better.

These areas represent the next steps in the journey to refine and enhance the classification of environmental sounds within the field of machine learning.

REFERENCES

[1] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[2] Elizalde, B., Deshmukh, S., Al Ismail, M., & Wang, H. (2022). CLAP: Learning audio concepts from natural language supervision. arXiv preprint arXiv:2206.04769. Retrieved from https://arxiv.org/abs/2206.04769