A Deep Learning Approach for Urban Sound Classification Using Convolutional Neural Networks

Ç. İlayda Kurnaz
Department of Engineering
Atılım University

Baran Türkdoğan
Department of Engineering
Atılım University

Huseyn Kishiyev
Department of Engineering
Atılım University

***Abstract*—As cities become smarter and more interconnected, the classification of urban sound is becoming increasingly important. This study is devoted to investigating the application of Convolutional Neural Networks (CNNs) for categorization of urban sounds, a problem which has great relevance to environmental monitoring, public safety, and urban planning. By converting audio recordings into spectrograms we allow the CNN model to recognize without training the patterns and features of the sound data. We have chosen ten different sound classes that are relevant to the urban environment, including "dog_bark," "car_horn," "jackhammer," and "drilling." The CNN model was highly effective, with the overall accuracy of 92.3%. and accuracy of particular categories from 73% to 99%. Extensive experiments and analyses show how the deep learning approach, especially with CNNs, can be successfully applied to this problem of identification of subclasses of urban sounds. The result indicates that it may be quite useful for the real-world application in smart cities and environmental monitoring.**

***Index Terms*—Urban sound classification, Convolutional Neural Networks, Deep learning.**

I. Introduction

Sound classification is of great importance in many real-world applications, including environmental monitoring, smart cities, and urban planning. The ability to classify urban sounds, such as traffic noise, construction sounds, or natural sounds, enables actionable insights for automation, safety, and analytics.

Urban environments are challenging for sound classification due to overlapping sources and dynamic noise levels. Traditional methods are based on handcrafted features, such as Mel-frequency cepstral coefficients (MFCCs) or zero-crossing rates, which are generally not effective in capturing the complexity of patterns found in urban soundscapes. Deep learning, especially Convolutional Neural Networks (CNNs), has proven to be a promising alternative in automating feature extraction and learning from raw data.

This paper aims at categorizing ten urban sound categories: "dog_bark," "children_playing," "car_horn," "air_conditioner," "street_music," "gun_shot," "siren," "engine_idling," "jackhammer," and "drilling." These were selected based on their presence in urban soundscapes, and the various acoustic properties associated with them. Spectrograms are an appropriate input feature to use with deep learning models from images. This paper delves into the development, training, and evaluation of a CNN for classifying these urban sounds and presents results and challenges that came up during the process.

II. Objective

The goal of this project is to build an effective method for the classification of urban sounds using CNNs. This project will use audio spectrograms as input features to classify a variety of urban sound classes, which are "dog_bark," "car_horn," "jackhammer," and "drilling." Deep learning models are constructed to recognize these sound events in the urban environment, as they form the basis for many applications in smart cities, environmental monitoring, and automated systems.

During the project, I will try to:

- **Explore the Use of CNNs for Sound Classification:** Implement a CNN-based architecture tailored for audio classification, utilizing spectrograms to capture the frequency and time domain characteristics of urban sounds.
- **Achieve High Accuracy:** Aim to achieve an overall classification accuracy above 90%, with a particular focus on ensuring reliable performance across all sound categories.
- **Contribute to Smart City Applications:** Demonstrate the model's potential for real-time sound recognition, providing valuable insights into urban environments that could be used for automation, noise monitoring, and safety applications.
- **Enhance Understanding of Deep Learning in Audio Processing:** Develop a deeper understanding of how deep learning techniques can be applied to audio signals, specifically for urban soundscapes, and overcome challenges like overlapping sounds and background noise.

By the end of this project, I expect to have a fully functional and accurate urban sound classification model that can be applied to real-world scenarios, benefiting smart city technology and environmental monitoring efforts.

III. Methodology

A. Pipeline Steps

The workflow consists of the following steps:

1) **Data Loading**: Audio files were loaded from Google Drive using Librosa.
2) **Feature Extraction**: Audio signals were converted into 128×128 Mel spectrograms.
3) **Data Splitting**: The dataset was divided into training (80%) and testing (20%) sets.
4) **Model Training**: A CNN was trained on the extracted features using TensorFlow/Keras.

5) **Evaluation and Visualization**: Metrics such as accuracy and confusion matrices were computed and visualized.

B. Model Design

The CNN architecture comprises:

- **Input Layer:** Accepts $128 \times 128 \times 1$ spectrogram inputs.
- **Convolutional Layers:** Feature extraction using 3×3 filters.
- **Batch Normalization:** Stabilizes learning and improves convergence.
- **Pooling Layers:** Dimensionality reduction via max pooling.
- **Dropout Layers:** Prevents overfitting by randomly deactivating neurons during training.
- **Fully Connected Layers:** Final classification into sound categories.

IV. Modules and Key Functions

This study utilizes several Python libraries to streamline the data processing and classification pipeline:

A. Imported Modules

- **Librosa**: Used for audio signal processing, including loading audio files and generating spectrograms.
- **Numpy**: Handles numerical operations for data preprocessing and feature normalization.
- **Matplotlib**: Visualizes spectrograms, accuracy plots, and confusion matrices.
- **Torch**: Manages device allocation (e.g., CPU or GPU) for computational efficiency.
- **TensorFlow/Keras**: Provides the framework for designing, training, and evaluating the CNN model.

B. Key Functions

*a) **create_spectrogram**:* This function generates a Mel spectrogram from an audio file:

```
def create_spectrogram(file_path, n_mels=128):
    y, sr = librosa.load(file_path)
    spectrogram = librosa.feature.melspectrogram
    (y, sr, n_mels=n_mels)
    log_spectrogram = librosa.power_to_db
    (spectrogram, ref=np.max)
    return log_spectrogram
```

*b) **augment_audio**:* This function applies data augmentation techniques like noise addition and pitch shifting:

```
def augment_audio(y, sr):
    noise = np.random.normal(0, 0.05, y.shape)
    y_noisy = y + noise
    y_shifted = librosa.effects.pitch_shift
    (y, sr, n_steps=2)
    return y_noisy, y_shifted
```

V. Convolutional Neural Network (CNN) Architecture

CNNs are a type of deep learning models, very effective in dealing with grid-like data such as images or spectrograms. The CNN is a type of feedforward neural network that automatically and adaptively learns spatial hierarchies of features, which are very important in image classification, object detection, and audio classification for this study. Below, we describe the CNN architecture used in our model in greater detail.

A. CNN Architecture Design

This research uses a Convolutional Neural Network architecture that has layers, each dedicated to feature extraction and classification. The detailed architecture employed is as described below:

- **Input Layer:** The input to this network is the spectrogram, 128 × 128 × 1 in greyscale. That is, there is a 128 × 128 resolution on the timefrequency representation of the audio signal. Each of the pixels contained in this spectrogram will encode the strength of the signal at a particular frequency and time point.
- **Convolutional Layers:** These layers perform feature extraction by applying a set of filters (or kernels), typically of size 3×3. Their main objective is to capture key patterns such as edges, textures, and more intricate structures that are discovered as the network deepens.
- **Activation Function:** After each convolution operation, a non-linear activation function, such as the Rectified Linear Unit (ReLU), is employed. This activation introduces the non-linearity needed for the model to capture more complex patterns in the data.
- **Batch Normalization:** This layer helps in stabilizing and speeding up the training process by normalizing the input to each layer. The input is standardized to have a mean of zero and a variance of one, which makes training more stable and improves the model's ability to converge efficiently.
- **Pooling Layers:** Max-pooling layers serve the purpose of reducing the spatial dimensions of the feature maps. This not only helps retain the most critical features but also reduces computational costs and overfitting. A typical pooling operation halves the size of the feature map.
- **Dropout Layers:** Dropout is used as a regularization method to prevent the model from overfitting. During training, a fraction of neurons is randomly set to zero, forcing the network to learn more generalized and robust features.
- **Fully Connected Layers:** After feature extraction, the fully connected layers are responsible for making the final classification decisions. The output layer consists of neurons corresponding to the different target classes (in this case, different sound categories). The softmax function is typically applied at the output to convert the raw scores into probabilities, which represent the likelihood of each class.

B. CNN Architecture Illustration

The following image illustrates the typical architecture of a Convolutional Neural Network (CNN) as applied to audio classification:

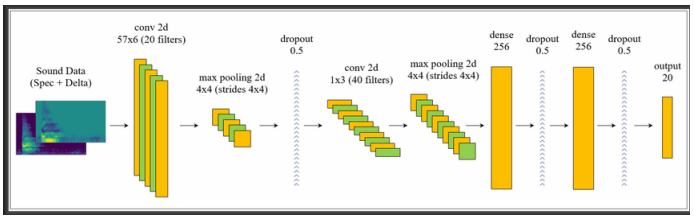

Fig. 1. Illustration of a typical CNN architecture used for audio classification.

C. Details of Layers

- **Convolutional Layer:** Each convolutional layer applies multiple filters to the input, resulting in a set of feature maps that highlight various aspects of the input image (such as edges or textures).
- **Pooling Layer:** After the convolution, a pooling operation (often max-pooling) reduces the spatial dimensions of the feature map. This helps the model focus on the most important features and reduces the computational burden.
- **Fully Connected Layer:** After the convolutional and pooling layers, the high-level features are passed into one or more fully connected layers. These layers map the learned features to the final output classes.

D. Deep Learning and CNNs

In deep learning, CNNs have proven to be extremely effective due to their ability to automatically extract features from data, minimizing the need for manual feature engineering. By learning hierarchical features at different levels of abstraction, CNNs are able to achieve state-of-the-art performance on tasks such as image classification, object detection, and in this case, audio classification using spectrograms.

E. Summary of the CNN Model

The CNN used in this study can be summarized as follows:

- Input: $128 \times 128 \times 1$ spectrogram.
- Layers: Multiple convolutional layers, activation functions, batch normalization, max-pooling layers, dropout, and fully connected layers.
- Output: A set of predicted probabilities for different sound categories.

The combination of these layers enables the CNN to effectively learn features from spectrograms, classify the audio signals into their respective categories, and generalize well to unseen data.

VI. Results and Key Outputs

A. Performance Metrics

Figures 2 and 3 depict the model's accuracy and loss during training and validation:

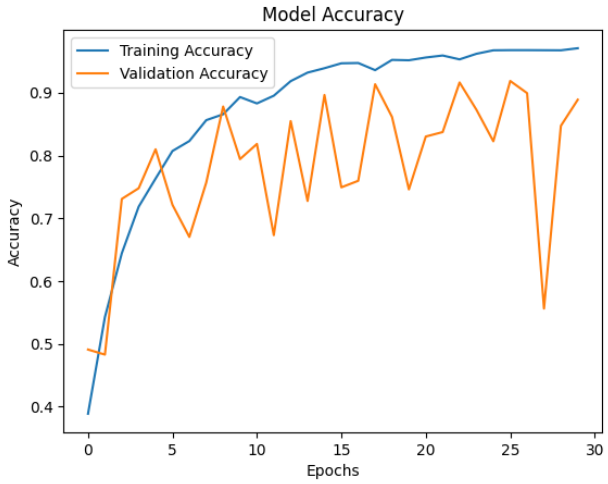

Fig. 2. Training and validation accuracy across epochs.

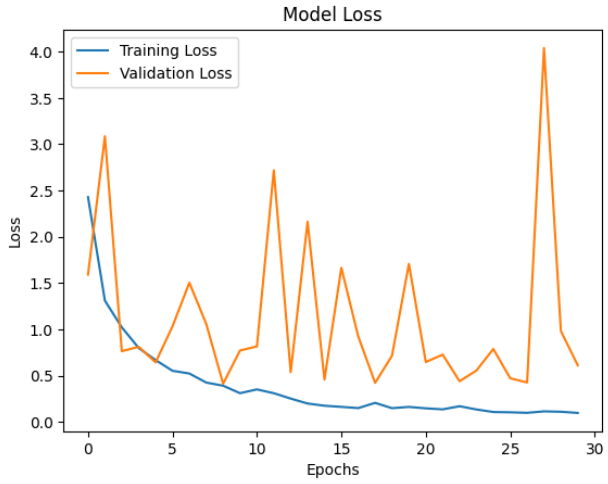

Fig. 3. Training and validation loss across epochs.

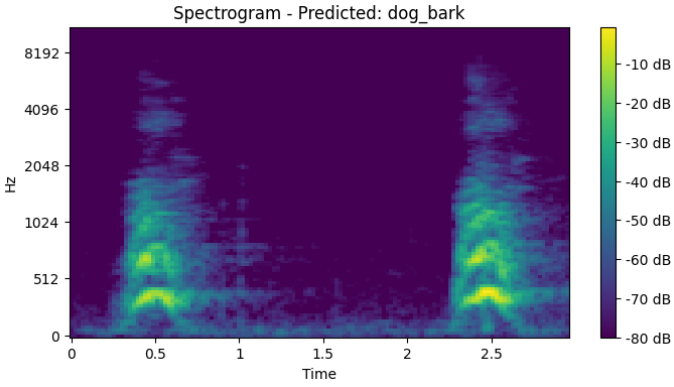

Fig. 4. Predicted spectogram for the input audio.

B. Predicted Spectrogram

The Figure 4 shows the predicted Mel spectrogram for a given audio input. This spectrogram represents the time-

frequency representation of the audio signal, which is used for classification tasks.

C. Confusion Matrix

Class-wise performance was analyzed using a confusion matrix (Figure 5), which shows the model's ability to classify each category accurately.

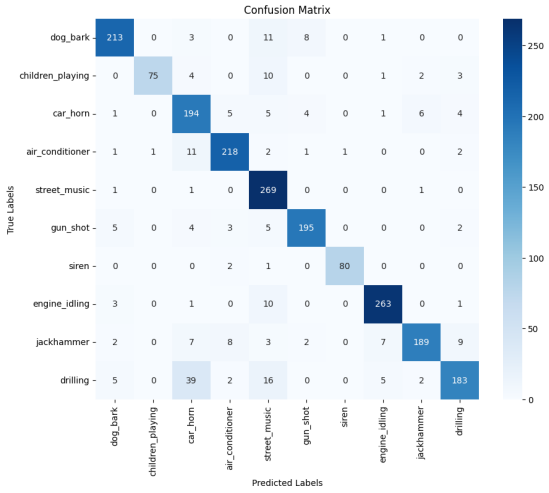

Fig. 5. Confusion matrix showing class-wise performance on the test dataset.

VII. Discussion

The results validate the efficacy of CNNs for urban sound classification. Key takeaways include:

- **Strengths:** The model achieved excellent performance for classes such as "dog_bark," "drilling," and "gun_shot," with accuracies exceeding 90%.
- **Challenges:** Lower accuracies in "street_music" and "car_horn" suggest challenges with overlapping sound patterns.
- **Future Work:** Explore advanced architectures like RCNNs and ResNets to improve classification for overlapping sound categories.

VIII. Conclusion

This study successfully demonstrates the application of CNNs for urban sound classification. By automating feature extraction and leveraging spectrograms, the model achieved robust performance. The high accuracy for most classes indicates its potential for practical applications in smart cities and environmental monitoring. Future research will focus on expanding the dataset and incorporating real-time classification systems.

References

[1] IEEE: A Deep Learning Approach for Urban Sound Classification
[2] ArXiv: An Overview of Deep Learning in Sound Processing
[3] ResearchGate: A Deep Learning Approach for Urban Sound Classification