A comparison experiment of convolution and recurrent neural network models on audio classification problems, by using big data technologies

Zhe Luo  
CCT College DublinDublin  
2023234@student.cct.ie

***Abstract***—This research explores the integration of big data technologies with deep learning models for audio classification, employing Apache Spark, Apache Hadoop, and HDFS for data management and Exploratory Data Analysis (EDA). Through visualizing feature extraction and data augmentation, it lays the foundation for further experimentation. The study conducts preliminary experiments, merging Spark's parallel processing with convolutional and recurrent neural network models, to assess their cohesive potential for improving audio classification methodologies. A comparasion analysis of these models, based on distinct audio data feature extraction techniques, provides a thorough evaluation. The research addresses critical questions regarding the feasibility of integrating big data with deep learning for audio data classification and compares the performance of convolutional versus recurrent neural network models in this context. Findings highlight the synergistic potential of these technologies in managing audio classification, discuss implications, limitations, and suggest directions for future research, culminating in a comprehensive synthesis of the conducted research.

Keywords—HDFS,Spark,Deep-learning ,Audio Classification

# Introduction (*Heading 1*)

The integration of big data and deep learning technologies represents a fast growing in the realm of audio classification, marking a significant leap forward from traditional methods. Inspired by seminal works, this research delves into the synergistic potential of these technologies to enhance the classification and recognition of audio datasets.

Thornton, Zhang, Leitner, and Boyang (2019) laid the groundwork for this exploration by effectively utilizing Mel Spectrograms in conjunction with Convolutional Neural Networks (CNNs) for audio recognition tasks. Their pioneering work underscored the efficacy of Mel frequency cepstral coefficients (MFCCs) in audio data processing, spotlighted the pivotal role of Mel spectrogram feature extraction techniques in bolstering audio recognition capabilities. This foundational research illustrates the critical importance of sophisticated feature extraction methodologies in the successful classification of audio data.

Building on these insights, Olusola O. Abayomi-Alli et al. (2022) contributed a comprehensive review on the use of data augmentation and deep learning methods in sound classification. Their analysis, emphasizing practical data enhancement techniques, particularly the concept of data augmentation, provides a crucial strategy for addressing data overfitting issues in audio classification research. This methodological advancement, as highlighted in their review, serves as a key technique in the development of robust models capable of generalizing well on unseen dataset.

The pursuit of integrating big data and deep learning technologies for audio classification is further inspired by the innovative approach of Sharan Duggirala and Moh Teng-Sheng (2020) in music genre classification. By employing Natural Language Processing (NLP) and Spark, their research showcases the seamless integration of deep learning frameworks with big data technologies, thereby paving the way for novel approaches in music genre classification. This integration demonstrating the capacity for advanced computational frameworks to effectively handle large-scale, complex audio datasets, providing a blueprint for future research in the domain.

Additionally, the choice of Spark as the computational framework is informed by the performance comparison conducted by Sandeep Bhargava, Goyal, Bright Keswani, and Dinesh (2019). Their findings explain Spark's exceptional performance among mainstream big data distributed computing frameworks, underscoring its aptitude for managing large-scale audio data with sophisticated machine learning algorithms. This comparison not only highlights Spark's computational efficiency but also its scalability and flexibility in processing vast datasets.

This study seek to contribute valuable insights to the field of audio classification, emphasizing the instrumental role of integrating big data and deep learning technologies. This study sets the stage for a detailed investigation into the comparative efficacy of convolutional and recurrent neural network models in audio classification, supported by the robust capabilities of big data technologies. Through a series of objectives that include utilizing Apache Spark, Apache Hadoop, and HDFS for storage management and exploratory data analysis, this research aims to provide a comprehensive evaluation of these models based on different audio data feature extraction techniques. By addressing key research questions, discussing findings, and outlining potential future research directions, this study seeks to contribute valuable insights to the field of audio classification, emphasizing the instrumental role of integrating big data and deep learning technologies.

## Topic

A comparative experiment on audio classification problems through the combination of big data and deep learning technologies..

## Objectives:

* Utilize Apache Spark, Apache Hadoop, and HDFS for storage management and Exploratory Data Analysis (EDA) of audio datasets, and proof how to visualize feature extraction and data augmentation, laying the groundwork for subsequent experiments.
* Conduct preliminary experiments by integrating Spark's parallel data processing with convolutional and recurrent neural network models to demonstrate the seamless integration of big data technologies and deep learning frameworks, deriving theoretical grounds for further model improvements from the results..
* Perform a comparative experiment of two sets of convolutional and recurrent neural network models based on different audio data feature extraction techniques and evaluate the models..

Identify applicable funding agency here. If none, delete this text box.

* Discuss key findings, specifically their implications and limitations, and potential future research directions
* Conclusion based on all the research conducted.

## Research Questions

* Can big data technologies be effectively integrated with deep learning technologies for large-scale audio data classification problems?
* How do convolutional and recurrent neural network models perform in audio data classification problems?

# Literature Review

In the age of big data and advanced computing, the intersection of Apache Spark's robust data processing capabilities with deep learning techniques has ushered in a new era of analytical possibilities, especially in the realm of audio classification. The evolution from traditional data processing frameworks like Hadoop's MapReduce to more dynamic and efficient systems such as Spark has marked a significant leap in handling large-scale data analysis. This literature review draw upon a comprehensive array of research to explore the synergy between Spark and deep learning models for audio classification, emphasizing the integration of Mel Frequency Cepstral Coefficients (MFCCs) and Mel Spectrograms for feature extraction, and the pivotal role of data augmentation and overfitting prevention strategies.

Bansod (2015) and Hazarika et al. (2017) discussed the efficiency of big data analysis with Spark, highlighting its superior performance over Hadoop's MapReduce. Spark's in-memory data processing capability significantly reduces the time required for processing large datasets, making it an attractive choice for data analytics. Moreover, the integration of Spark with HDFS allows for seamless access to data stored in Hadoop's distributed file system, enhancing the scalability and reliability of data processing operations.

The synergy between Spark and deep learning is exemplified in the work of Pumperla and Cahall (2022), who discuss a framework for distributed deep learning with Keras and Spark. This integration leverages Spark's distributed computing capabilities to accelerate deep learning algorithms, demonstrating the potential of combination Spark with advanced machine learning techniques to address complex data analysis tasks.

In the domain of audio classification, Spark's application has shown promising results. Duggirala and Moh Teng-Sheng (2020) present a novel approach to music genre classification using natural language processing techniques and Spark, highlighting Spark's versatility in handling diverse data types and complex analytical tasks. Similarly, Chaudhury et al. (2022) explain the efficacy of using machine learning with Apache Spark for large-scale music genre analysis and classification, further illustrating Spark's capability in processing and analyzing audio data efficiently.

The programming style of PySpark, as discussed by Drabas and Lee (2017), emphasizes simplicity, readability, and efficiency. PySpark allows developers to utilize the power of Spark through Python, offering a rich set of APIs for data manipulation and analysis. Effective programming in PySpark involves leveraging Spark's data abstraction models, such as RDDs (Resilient Distributed Datasets) and Dataframe, to optimize data processing tasks and improve the performance of distributed systems.

In the rapidly advancing field of audio classification, deep learning has taken center stage, offering groundbreaking approaches to understanding complex audio data. The critical choice of feature extraction method, specifically Mel Frequency Cepstral Coefficients (MFCCs) and Mel Spectrograms, paired with the use of advanced neural network architectures like Convolutional Neural Networks (CNNs), Long Short-Term Memory networks (LSTMs), Gated Recurrent Units (GRUs), and Convolutional Recurrent Neural Networks (CRNNs), forms the foundation of modern audio processing techniques.

AbdulKh.Zrar & Al-TalabaniK.Abdulbasit (2022) provide an insightful examination of MFCCs, detailing their vital role in capturing the spectral properties of audio signal. This method's efficacy in emphasizing the phonetic characteristics of sound makes it indispensable for audio classification tasks. Concurrently, Thornton et al. (2019) highlight the effectiveness of Mel Spectrograms in conjunction with CNNs for audio recognition. The visual representation of sound offered by Mel Spectrograms enables CNNs to detect patterns within complex audio landscapes, making it a powerful tool for sound analysis.

The intersection of MFCCs and Mel Spectrograms with neural networks has sparked significant advancements in audio classification. For instance, Ahmed A. Khamees et al. (2021) demonstrate the potential of combining CNNs and RNNs to classify music genre, revealing the complementary strengths of these models in capturing both spectral and temporal audio features. Yu-Huei Cheng et al. (2020) further the discourse by showcasing the CRNN model's proficiency in music genre classification, illustrating how the integration of convolutional and recurrent layers can enhance model performance by effectively processing both spatial and sequential data.

Moreover, Alessandro Maccagno et al. (2021) apply a CNN approach to audio classification in construction sites, demonstrating the adaptability of CNNs in diverse audio environments. This versatility is critical for developing models that can function in specific contexts, from urban soundscape analysis to industrial noise monitoring.

The role of data augmentation in mitigating overfitting and enhancing model robustness is underscored by Olusola O. Abayomi-Alli et al. (2022). Through techniques like pitch shifting, time stretching, and adding white noise, models are exposed to a wider array of acoustic variations, which is crucial for developing generalizable and resilient classification systems. This systematic review highlights the importance of data augmentation in training deep learning models to perform consistently across varied audio datasets.

In conclusion, the convergence of Apache Spark's data processing capabilities with advanced deep learning models represents a frontier in audio classification. The integration of MFCCs and Mel Spectrograms for feature extraction, combined with strategic data augmentation techniques, offer a comprehensive approach to tackling the challenges of audio signal processing. As this review demonstrates, leveraging the strengths of Spark alongside deep learning innovations presents a promising pathway for future research and application in audio analysis, promising not only to enhance the accuracy and efficiency of classification tasks but also to explore new dimensions in understanding audio data in the era of big data.

## Ⅲ Methodology

3.1 Dataset Selection

The dataset we have utilized, ESC-50, is a publicly available audio dataset meticulously curated to span 50 distinct categories, ranging from animal sounds and natural noise to human-made noises and musical instruments. Each category in the ESC-50 dataset is represented by 40 distinct audio recordings, summing up to a total of 2000 clips, all uniformly sampled to ensure consistency. This dataset is widely recognized in the sound classification research community for its comprehensive variety, the uniformity of audio sample conditions, and the challenges it presents due to the subtle nuances and similarities across categories. Its structured format and diverse content make it a reliable benchmark for evaluating the efficacy of various sound classification models. ESC-50's widespread use in academic studies and competitions has further established its status as a definitive resource for those seeking to innovate and improve upon existing audio analysis technologies.

3.2 License

The dataset is available under the terms of the [Creative Commons Attribution Non-Commercial license](https://creativecommons.org/licenses/by-nc/3.0/). A smaller subset (clips tagged as ESC-10) is distributed under CC BY (Attribution). Attributions for each clip are available in the [LICENSE file](https://github.com/karolpiczak/ESC-50/blob/master/LICENSE).

3.3 Configuration Phase

In this phase, the primary focus was on integrating Spark and HDFS. To introduce the hardware environment for this experiment, it was conducted on a Linux virtual machine environment powered by an Intel(R) Core(TM) i5-9300H CPU @ 2.40GHz with 14GB of RAM. The setup began with the installation of Hadoop, followed by the installation of Spark and PySpark on top of Hadoop. This configuration enabled the use of Spark's distributed cluster under the Python programming environment. The experiment was set to local mode utilizing all 8 CPU cores available, though it could be adjusted to distributed cluster mode depending on the availability of distributed cluster resources and specific requirements.

The dataset was imported into HDFS using the following method. The choice of HDFS was due to its scalability, high fault tolerance, high throughput, cost-effectiveness, ease of use, data locality optimization, and support for large files, making it the preferred solution for processing and storing large-scale datasets. Most importantly, its compatibility ensures seamless integration with Spark.

3.4 EDA Phase

In this stage, the main tool used is Spark's distributed cluster technology, primarily because Spark employs an advanced Directed Acyclic Graph (DAG) execution engine and the core structure of Resilient Distributed Datasets (RDDs). Furthermore, Spark optimizes the computation process through lazy evaluation, meaning transformation operations on data are not executed immediately but are triggered by action operations (like count, collect) which then perform the actual computations.

Therefore, in this experiment, the objective was to minimize triggering actions as much as possible. Each time an action is executed, Spark initiates a new job to carry out all necessary transformations. The execution of jobs involves task scheduling and the allocation of execution resources (such as CPU and memory), where frequent triggering of action operations leads to repetitive scheduling and resource allocation overheads, reducing overall processing efficiency. Appropriate use of spark.stop() was also made to stop the SparkContext process, allowing for effective management of cluster resources.

Using Spark, the dataset was examined to ensure consistency in the data volume of each category, as well as uniformity in duration, sampling rates, and more. The dataset's audio was accessed and displayed individually using spark.sparkContext.binaryFiles combined with Python's librosa library. This setup facilitated the extraction of audio features through Mel spectrograms and Mel Frequency Cepstral Coefficients (MFCC), along with the visualization of 6 different data augmentation techniques. Finally, after extracting audio features using Mel spectrograms and applying two data augmentation techniques, the GRU model was experimented with audio from 5 categories of the dataset due to memory constraints. Based on the results, experiences were summarized to prepare and provide a benchmark for subsequent formal deep learning experiments.

3.5 Preliminary Experiment

After extracting audio features using Mel spectrograms and applying two types of data augmentation, the GRU model was used to experiment with audio from 5 categories of the dataset due to memory constraints. This step aims to identify any significant flaws in the previous experimental work and summarize experiences based on the results. The insights gained will prepare for and benchmark subsequent formal deep learning experiments.

3.6 Preliminary Experiment Result

Preliminary experiment demonstrated that while it is feasible to combine Spark with local Python libraries for data processing, the efficiency in a single-machine setup is low, and there's a high risk of memory overflow. This issue likely arises because Spark, even in single-machine mode, engages in task scheduling and simulates network communication as it would in a distributed environment. These operations do not cease in a single-machine context, thus consuming extra CPU time and causing delays. However, this experiment has successfully executed distributed clustering from the EDA phase to data processing and integrating with deep learning workflows. In real-life scenarios with multiple nodes, the combination of Spark and Hadoop can fully leverage its speed, versatile data processing capabilities, excellent fault tolerance, and efficient resource management.

From the perspective of deep learning, despite using the bidirectional GRU model anticipated to perform well, alongside dropout, L2 regularization, and two data augmentation methods (adding noise and changing speed), the model still faced severe overfitting issues. This could be attributed to the ESC-50 dataset having only 40 samples per category, which is insufficient for deep learning. Despite 100 iterations, the model's generalization ability remained weak. Thus, future research will focus on breaking through using data augmentation and expansion to improve the model.

3.7 Experimental Phase

In this formal experimental phase, a comparison of four models—LSTM, GRU, CNN, and CRNN—is conducted. It was observed that using Spark for feature extraction and data augmentation under local conditions was not optimal due to performance limitations. Therefore, only the PySpark compiler and TensorFlow framework were used, without employing Spark for data processing. From the ESC50 dataset, the first five categories were extracted for the experiments to increase efficiency and reduce the occurrence of anomalies. During this phase, comparative experiments were conducted using two different audio feature extraction methods: Mel spectrograms and Mel Frequency Cepstral Coefficients (MFCC). Mel spectrograms visually represent the sound signal spectrum based on human perception of sound frequencies. Human hearing is not linear across frequencies; it is more sensitive to changes at lower frequencies and less so at higher frequencies. The Mel scale is based on this non-linear perception and has two dimensions, the first representing Mel frequency and the second time. MFCC is a feature further extracted from Mel spectrograms. Through the application of Discrete Cosine Transform (DCT) to Mel spectrograms, cepstral features of the sound are captured. The most intuitive difference is that after MFCC feature extraction, the data dimensions are smaller and more abstract. In this experiment, the data dimensionality after Mel spectrogram feature extraction was (128, 270), while after MFCC extraction, it was reduced to (20, 270). This means a significant reduction in frequency features while preserving the characteristics of temporal evolution.

3.7 The criterion for the comparison

The same data preprocessing approach was applied across all experiments, employing padding to adjust all data to the appropriate dimensions and utilizing the MinMaxScaler feature scaling method to ensure uniformity in feature scales. The same training-to-testing dataset split ratio was maintained, and a similar network architecture was deployed across models, with the core network structure consisting of three layers and neuron counts of (16, 32, 16). Identical data augmentation techniques were applied, including adding white noise, changing speed, pitch shifting, and time shifting to enrich the data. The augmented data was then stacked with the original dataset, achieving an expansion effect from the initial 200 samples to 1000. The same activation function, ReLU, the same dropout rate (0.3), and the same L2 regularization coefficient (0.01) were consistently used to prevent overfitting. Based on these standards, the four models underwent 100 iterations of training, followed by a comparison of the final testing accuracy and runtime.

These specific parameter choices were meticulously considered and are mostly grounded in empirically validated effective practices. For example, the use of MinMaxScaler for feature scaling not only stabilizes and enhances the efficiency of models but also accelerates the convergence rate of the models, effectively avoiding gradient vanishing or exploding. The choice of the ReLU activation function, compared to other functions like sigmoid or tanh, offers simplicity in computation—requiring only a determination of whether the input is greater than zero—thus ensuring greater efficiency during training and inference. Similarly, the selection of a 0.2 dropout rate and a 0.01 L1 regularization coefficient represents a conservative approach based on preliminary experimentation. All these choices aim to ensure that each model trains in a comparable environment.

3.8 Limitations

Due to hardware constraints, Spark was not utilized for data preprocessing in the formal experiment phase, despite its proven feasibility. This decision was made because using Spark for data augmentation in a single-machine environment would significantly increase run times and could easily lead to data overflow and memory exhaustion. Even in local mode, Spark attempts to manage and schedule tasks through its task scheduler. While this scheduling is essential in a distributed environment to coordinate work across different nodes, it introduces unnecessary overhead in a single-machine setting, consuming resources even when task scheduling and network communication simulation are not required.

Additionally, skill limitations necessitated the selection of four out of six possible data augmentation techniques. There is reason to believe that employing a broader range of data enhancement methods could achieve higher accuracy in model training. Moreover, the scale of the deep learning network had to be kept relatively small due to these limitations. Although this does not affect comparative experiments, increasing the number of network layers and the count of neurons could potentially lead to improved accuracy.

3.9 Experimental Result

Model Comparison Based on Mel Spectrogram Feature Extraction Data

Model Comparison Based on Mel-Frequency Cepstral Coefficients (MFCCs) Feature Extraction Data

From the comparison results, the impact of different feature processing methods on the models is significant. Firstly, regarding run time, after feature extraction with MFCC, due to the significant reduction in data dimensionality, all models saw a substantial decrease in run time, indicating a considerable reduction in the computational power required for the models.

Regarding accuracy, for recurrent neural networks represented by LSTM, bidirectional LSTM, and bidirectional GRU, there was a huge performance disparity under different feature extraction techniques like Mel spectrograms and MFCC. This could be because MFCC, being a further extraction from Mel spectrograms, might lose more information crucial for recurrent neural networks. Additionally, recurrent neural networks are typically sensitive to the length of the input sequence. After feature extraction with MFCC, the length of the feature sequence is significantly reduced, making the associations between features more abstract and a critical factor affecting model performance.

In contrast, CNN and CRNN, which include convolutional layers, are less sensitive to changes in feature extraction techniques because they can capture local features and spatial hierarchies. Therefore, even when the representation becomes more abstract, their performance is not significantly affected. CRNN, despite performing well in two sets of comparative experiments, did not achieve the expected outcome of capturing both spatial and temporal information for superior training effects. Instead, training took a long time, possibly because CRNN integrates the structures of CNN and RNN, making it more complex than either alone. Additionally, CRNN has to perform gradient calculations and manage gradient backpropagation through time series, likely increasing computational burden. Also, CRNN needs to store both convolutional features and RNN states in memory, which could lead to frequent memory swapping operations under limited memory resources, further slowing down training speed.

CNN demonstrated good stability, achieving 89% accuracy in both sets of experiments with very consistent results, indicating strong robustness. In practical applications, with Mel spectrogram feature extraction, the bidirectional LSTM model also achieved 84% accuracy, with much shorter run times compared to CNN, offering an option depending on specific needs.

## Ⅳ Conclusion

In this comparative experiment, the integration of Hadoop, HDFS, Spark, PySpark with deep learning achieved significant success, holding profound practical implications. In today's era of data explosion, utilizing distributed systems for storage and computation is indispensable in real-world applications such as voice recognition and voice-controlled locks. Among the two groups of deep learning models compared, the CNN model exhibited the highest accuracy and robustness, while the bidirectional LSTM also performed commendably with Mel spectrogram feature extraction, offering a more lightweight solution. Although CRNN can combine the advantages of convolutional and recurrent neural networks, it may require more effort in data feature preprocessing and network structure design.

Future research directions, in the realm of big data, could tailor to the actual application fields and data scenarios. With Spark's adaptability in various environments, more methods for data storage, such as HBase, MongoDB, etc., can be explored. Using Spark's MLlib for experimenting with more machine learning models or even employing distributed deep learning frameworks for model training, such as distributed TensorFlow, Apache MXNet, etc., is also promising. In deep learning, continual optimization and selection of suitable feature extraction methods, data augmentation to expand datasets as much as possible, and choosing or designing the right model structures can achieve higher accuracy. Ideal research might also compare the performance of various models using more feature extraction techniques, data augmentation methods, and suitable network structures.

##### References

Abdul, Z. K., & Al-Talabani, A. K. (2022). Mel Frequency Cepstral Coefficient and its Applications: A Review. *IEEE*.

Ahmed A. Khamees, H. D. (2021). Classifying Audio Music Genres Using CNN and RNN.

Alessandro Maccagno, A. M.-C. (2021). *A CNN Approach for Audio Classification in Construction Sites*.

Bahmei, B., Birmingham, E., & Arzanpour, S. (2022). CNN-RNN and Data Augmentation Using Deep Convolutional Generative Adversarial Network for Environmental Sound Classification. *IEEE*.

Bansod, A. (August 2015). Efficient big data analysis with apache spark in HDFS. *International Journal of Engineering and Advanced Technology (IJEAT)*.

Hazarika, A. V., Ram, G. J., & Jain, E. (2017). Performance comparision of Hadoop and spark engine. *IEEE*.

Jothilakshmi, E. S. (2018). Large scale data based audio scene classification. *International Journal of Speech Technology*.

Max Pumperla1, 2. D. (2022). Elephas: Distributed Deep Learning with Keras &Spark.

Mousumi Chaudhury ORCID, A. K. (2022). Large-Scale Music Genre Analysis and Classification Using Machine Learning with Apache Spark. *Electronics*.

Olusola O. Abayomi-Alli 1ORCID, R. D.-O. (2022). Data Augmentation and Deep Learning Methods in Sound Classification: A Systematic Review. *Electronics*.

Purwins, H., Li, B., Virtanen, T., Schlüter, J., Chang, S.-Y., & Sainath, T. (2019). Deep Learning for Audio Signal Processing.

Sandeep Bhargava1, D. G. (2019). Performance Comparison of Big Data Analytics Platforms.

Sharan Duggirala, T.-S. M. (2020). A Novel Approach to Music Genre Classification using Natural Language Processing and Spark. *IEEE*.

Thornton, B. Z. (2019). Audio Recognition using Mel Spectrograms and Convolution Neural Networks.

Tomasz Drabas, D. L. (2017). *Learning PySpark.*

Yu-Huei Cheng, M. I.-C. (2020). Automatic Music Genre Classification Based on CRNN.

Word Count: 4050 words

.