# Comparison of traditional machine learning models for environmental sound classification

Filippo Uslenghi
Dipartimento di Informatica, Università degli Studi di Milano, Milan, Italy

***Abstract*—This study evaluates several traditional machine learning models — K-NN, SVM, MLP and Random Forest — on the ESC-10 dataset. Features such as MFCCs, energy, energy entropy and ZCR were extracted from audio signals, and each model was optimized using cross-validation. The results show that all models outperformed a baseline dummy classifier, with Random Forest achieving the highest accuracy. This study highlights the capabilities of traditional machine learning methods for ESC and identifies the most challenging and easily classified sound categories.**

## I. Introduction

ENVIRONMENTAL sound classification (ESC) is an important task in audio pattern recognition. It involves analyzing audio recordings and categorizing them into predefined classes. This task has gained attention because of its many practical applications. Effective classification of environmental sounds can improve various systems, including security and surveillance, wildlife monitoring and smart home technologies. For instance, detecting the sound of a breaking window can alert security systems to potential intrusions, while identifying bird songs can help with ecological studies. Additionally, in healthcare, ESC can help monitor patients by detecting sounds that indicate distress or abnormal activities. These uses show the great potential of ESC.

Traditional machine learning models have been widely used for ESC. These models rely on extracting features from audio signals and using classifiers to categorize the sounds. Typical features include Mel Frequency Cepstral Coefficients (MFCCs), Zero-Crossing Rate (ZCR), and spectral features, which capture the essential characteristics of audio signals [1]. Various classifiers, such as Support Vector Machines (SVM) have been widely successful due to their high accuracy with different kernels and configurations, with notable success in studies by Chu et al. [2], Theodorou et al. [3], and Zhang et al. [4]. K-Nearest Neighbour (K-NN) classifiers have also been effective, especially when optimized for specific values of k, as seen in the work of Bountourakis et al. [5]. Additionally, Artificial Neural Networks (ANN) and Hidden Markov Models (HMM) have shown varying levels of success, with HMMs achieving high classification accuracy in studies such as those by Ntalampiras et al. [6] and Zhan and Kuroda [7].

This study focuses on evaluating the performance of traditional machine learning models, such as K-NN, SVM, multilayer perceptron (MLP) , and Random Forest, in classifying sounds from the ESC-10 dataset. The models are tuned using cross-validation techniques to optimize their performance while their effectiveness is assessed by comparing their results to a baseline classifier. This study aims to highlight the strengths and limitations of traditional approaches in this context.

## II. Methods overview

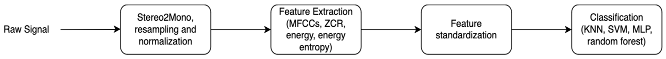
Raw Signal
Stereo2Mono, resampling and normalization
Feature Extraction (MFCCs, ZCR, energy, energy entropy)
Feature standardization
Classification (KNN, SVM, MLP, random forest)

Fig. 1. Block diagram of the pipeline used for audio classification.

The audio classification pipeline follows a standard procedure, shown in Fig. 1. First, all signals were converted to a single channel, resampled to a frequency of 44.1 kHz, and normalized using their Root Mean Squared (RMS) value. From the pre-processed signals time domain features were extracted, namely ZCR, Energy, and Energy Entropy, along with 39 MFCCs, from the 1st to the 40th (the 0th was discarded as it relates to the signal's energy). Time domain features were extracted using a short-term window. Mid-term statistics were then computed and averaged over the entire signal for a long-term perspective.

All the extracted features followed standardization and were used as input to four classifiers: K-NN, SVM, MLP, and Random Forest.

## III. Experimental setup

### A. Dataset

The dataset used in this work is the ESC-10 dataset, which is a smaller part of the larger ESC-50 dataset [8], designed for classifying environmental sounds. It includes recordings of 10 different sounds: dog bark, rain, sea waves, baby cry, clock tick, person sneeze, helicopter, chainsaw, rooster, and fire crackling (here denoted with integers from 0 to 9). There are 40 recordings for each sound, making the dataset completely balanced with a total of 400 audio files. These recordings come from free and public sources, capturing a variety of real-life sounds. All audio files in the ESC-10 dataset are in WAV format and each is 5 seconds long. They have a sampling rate of 44.1 kHz with a single channel and a bit depth of 16-bit. The dataset was then split into a training set of 300 samples files and a test set of 100 samples, preserving class balance.

### B. Feature extraction

As for the feature extraction, the MFCCs were extracted using the Torchaudio library [9], [10]. For the computation of the STFT, a Hanning window of 2048 samples and a stride of 512 samples was used. A total of 40 MFCCs were extracted, but the $0^{\text{th}}$ coefficient was discarded as it contains information related to the signal's intensity, which is not particularly useful for pattern recognition tasks.

For the time-domain features, external libraries were not used. The mean and standard deviation for each time-domain feature were computed over the entire signal as follows: the signal was first split into chunks of 1 second with an overlap of 0.5 seconds. For each chunk, a running window of 0.02 seconds with an overlap of 0.01 seconds was used to compute short-term features. These values were then aggregated by computing the mean and standard deviation in each chunk. Finally, the results were averaged over all chunks to obtain the long-term representation of the features. This process allowed to represent each audio file using 39 MFCCs along with the mean and standard deviation of ZCR, energy, and energy entropy, resulting in a total of 45 features per audio file.

### C. *Training and testing of the classifiers*

The training of the classifiers was carried out using cross-validation on the training set to determine optimal hyper-parameters. Once the hyper-parameters were identified, the classifiers were retrained on the entire training set and finally tested on the test set.

For the K-NN algorithm, different numbers of neighbors were tested, ranging from 1 to 25 and considering only odd numbers, as shown in Fig. 2. It was found that using only the first nearest neighbor was the most effective choice, resulting in an accuracy of 0.60 on the test set. The confusion matrix (Fig. 3) reveals that the third and sixth classes, corresponding to ”crackling fire” and ”rain”, were the most challenging to identify.

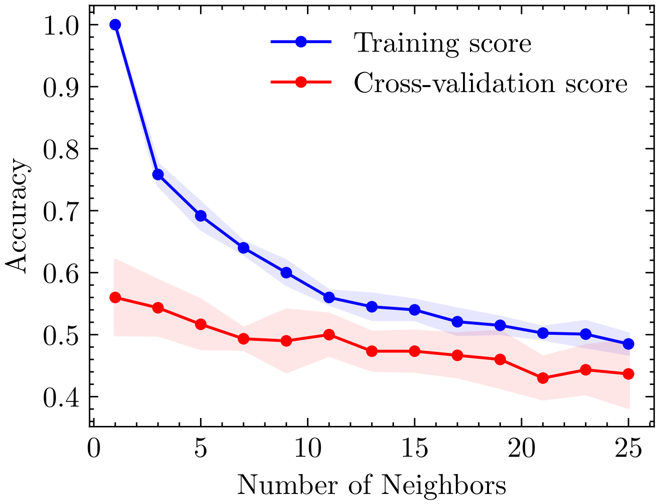

Fig. 2. Validation curve of K-NN considering different numbers of nearest neighbours.

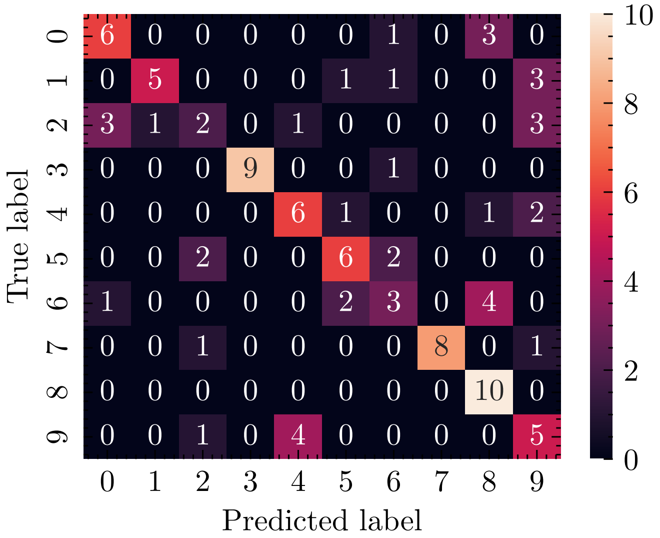

Fig. 3. Confusion matrix of K-NN considering one nearest neighbour.

For the SVM, a grid search was conducted to test different values for the hyper-parameters C and gamma, each varying between 0.01, 0.1, 1, 10, and 100. Additionally, different kernel functions were tested, including radial basis function (RBF), polynomial, and sigmoidal. A one-vs-one strategy was employed to handle the multiclass classification task. The cross-validation results showed that using a constraint parameter C with value 1, gamma with value 0.01 and an RBF kernel was the optimal choice, obtaining a 0.68 accuracy on the test set. The confusion matrix (Fig. 4) reveals that the class corresponding to ”rain” was the most challenging to identify.

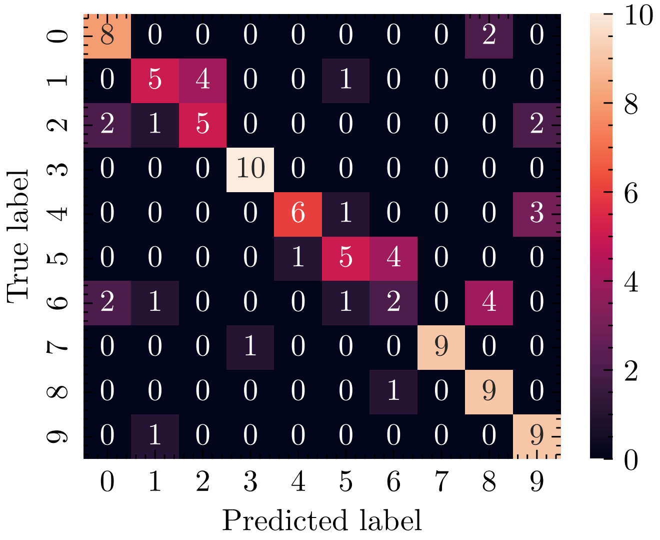

Fig. 4. Confusion matrix of SVM with C=1, gamma=0.01 and an RBF kernel.

A similar approach was conducted with the MLP. The hyper-parameters tested included: the number of hidden layers (one with 80 neurons, two with 80 and 50 neurons, three with 80, 50, and 50 neurons), learning rate (0.01 and 0.001), optimization algorithm (Adam and Stochastic Gradient Descent), activation function (ReLU and hyperbolic tangent (tanh)), weight decay (0.0001, 0.001, and 0.01), and batch size (32 and 64). The best hyper-parameter combination was found to be: two hidden layer (80 and 50 neurons), a learning rate of

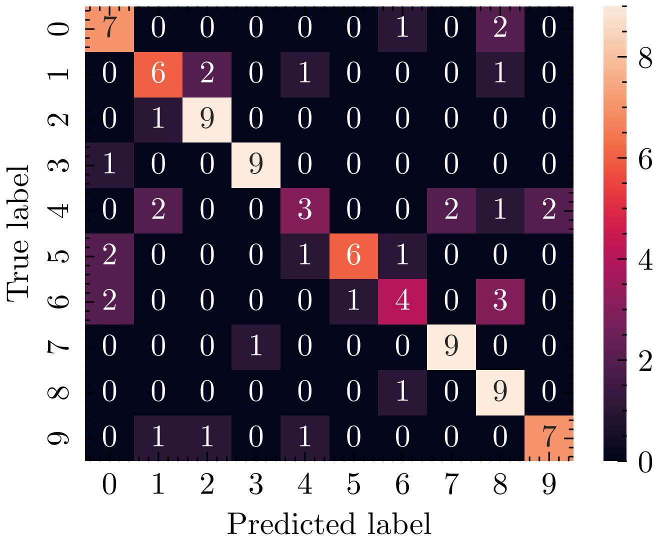

Fig. 5. Confusion matrix of MLP with two hidden layer of 80 and 50 neurons, learning rate of 0.01, adam solver, tanh activation function, weigth decay of 0.01 and batch size of 64 samples.

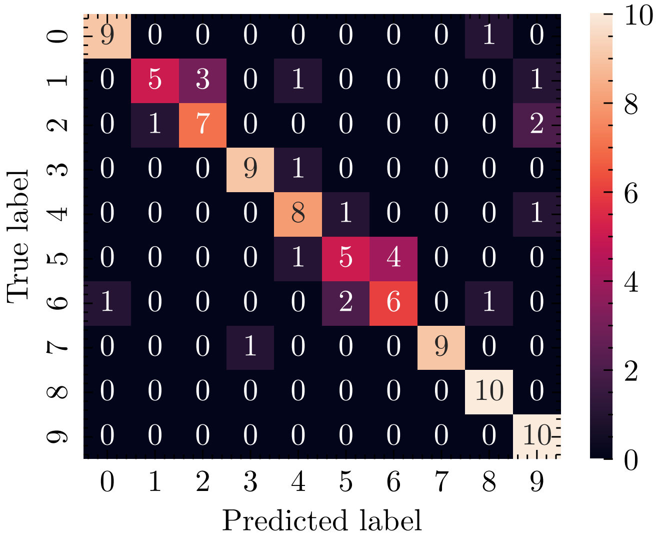

Fig. 6. Confusion matrix of Random Forest with Gini index, 300 trees and a maximum depth of 20.

0.01, the Adam solver, the tanh activation function, a weight decay of 0.01 and a batch size of 64 samples. The resulting MLP achieved 0.69 accuracy on the test set. The confusion matrix (Fig. 5) shows that the hardest classes to identify were the one corresponding to ”dog” and ”rain”.

Finally, the Random Forest classifier was trained using the same procedure. The grid search evaluated two different measures of leaf quality: the Gini index and entropy. It also tested various numbers of trees (100, 200, 300, 400, and 500) and different values for the maximum depth allowed for each tree (10, 20, 30, 40, and 50). The optimal hyper-parameters identified were: the Gini index for leaf quality, 300 trees, and a maximum depth of 20. The resulting Random Forest classifier achieved an accuracy of 0.78 on the test set. Its confusion matrix (Fig. 6) indicates that it successfully identified at least half of the samples in each class.

### *D. Results*

Aside from accuracy also precision, recall, and F1-score were computed to evaluate the models comprehensively. Additionally, for comparison purposes, a baseline dummy classifier was tested, which classified the test data by randomly choosing from a uniform distribution over the classes. This choice is motivated by the fact that the train set was completely balanced with respect to class distribution. The performance of all classifiers is reported in Table I and in Fig. 7.

TABLE I
ACCURACY, PRECISION, RECALL AND F1 SCORE OF ALL THE MODELS.

| Model | Accuracy | Precision | Recall | F1-score |
|---|---|---|---|---|
| Dummy | 0.10 | 0.10 | 0.10 | 0.10 |
| KNN | 0.60 | 0.62 | 0.60 | 0.59 |
| SVM | 0.68 | 0.68 | 0.68 | 0.67 |
| MLP | 0.69 | 0.69 | 0.69 | 0.68 |
| Random Forest | 0.78 | 0.78 | 0.78 | 0.77 |

The results clearly show that all classifiers performed significantly better than the dummy classifier, suggesting that they were able to capture meaningful patterns within the data. The K-NN classifier was the least performing, likely due to its simplicity. Indeed, more complex models, such as the non-linear SVM and MLP, improved the classification accuracy by 13% and 15%, respectively. However, the most effective algorithm was the Random Forest, which showed a 30% improvement over K-NN, approximately 15% over SVM, and 13% over MLP. This suggests that an ensemble of many small classifiers can better model the underlying distribution of the features.

Comparing the results with those reported in [8] reveals some differences. The K-NN classifier in this study performed worse, with an accuracy of 60% compared to 66.7% in their work. The SVM classifier showed similar performance, achieving 68% accuracy compared to 67.5% in their study. However, the Random Forest classifier performed better, with

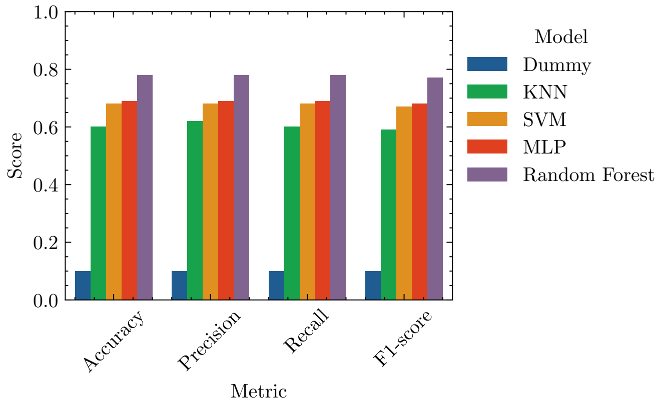

Fig. 7. Accuracy, precision, recall and F1 score of all the models tested and the baseline classifier.

TABLE II
F1-SCORE OF ALL CLASSIFIERS OVER ALL CLASSES.

| Model<br>Class | KNN | MLP | Random Forest | SVM |
|---|---|---|---|---|
| 0 | 0.60 | 0.64 | 0.90 | 0.73 |
| 1 | 0.62 | 0.60 | 0.62 | 0.56 |
| 2 | 0.25 | 0.82 | 0.70 | 0.53 |
| 3 | 0.95 | 0.90 | 0.90 | 0.95 |
| 4 | 0.57 | 0.38 | 0.76 | 0.71 |
| 5 | 0.60 | 0.71 | 0.56 | 0.56 |
| 6 | 0.33 | 0.47 | 0.60 | 0.24 |
| 7 | 0.89 | 0.86 | 0.95 | 0.95 |
| 8 | 0.71 | 0.69 | 0.91 | 0.72 |
| 9 | 0.42 | 0.74 | 0.83 | 0.75 |

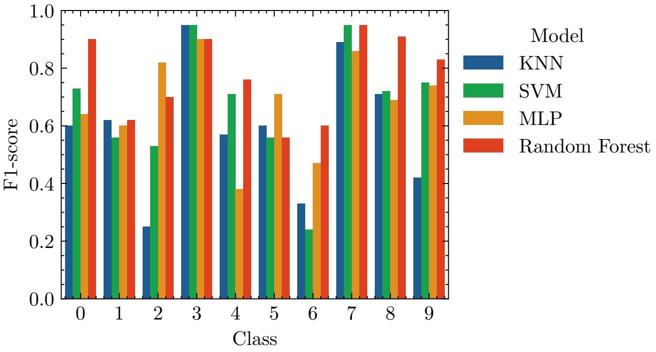

Fig. 8. F1-score obtained by all classifiers over each class.

an accuracy of 78% compared to 72.7% in their work. This variation in results is likely due to differences in the features used. Piczak et al. used only Zero Crossing Rate (ZCR) and the first 12 MFCCs (excluding the $0^{\text{th}}$ MFCC), whereas this study incorporated more features including energy, energy entropy, and 27 additional MFCCs.

Finally, an inspection of the F1-scores obtained by each model across each classes can help identify which classes were the most difficult to classify and which were easier to classify. This information is represented in Fig. 8 revealing that the class related to rain sounds was the hardest to identify overall, while the sounds of a crying baby and a crowing rooster were easily detected. This results are also displayed in Table II.

## IV. Conclusion

This study focused on classifying environmental sounds from the ESC-10 dataset using traditional machine learning models. The audio signals were processed, relevant features were extracted, and several classifiers were evaluated. Each classifier was optimized through cross-validation to identify the best hyper-parameters, and their performance was compared against a dummy classifier to assess effectiveness.

The results demonstrated that all classifiers significantly outperformed the dummy classifier, indicating their ability to capture meaningful patterns in the audio data. Among the classifiers, the Random Forest algorithm achieved the highest accuracy of 0.78.

F1-scores for each class were examined to determine which sounds were most challenging to identify. It was observed that rain sounds were the most difficult to classify, while the sounds of a crying baby and a crowing rooster were more easily detected.

In summary, while simpler models such as K-NN offer a baseline level of performance, more advanced models, such as SVM, MLP, and particularly Random Forest, provide higher accuracy.

## References

[1] A. Bansal and N. K. Garg, "Environmental sound classification: A descriptive review of the literature," *Intelligent Systems with Applications*, vol. 16, p. 200115, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2667305322000539

[2] S. Chu, S. Narayanan, C.-c. J. Kuo, and M. J. Mataric, "Where am i? scene recognition for mobile robots using audio features," in *2006 IEEE International Conference on Multimedia and Expo*, 2006, pp. 885–888.

[3] T. Theodorou, I. Mporas, and N. Fakotakis, "Automatic sound recognition of urban environment events," in *Speech and Computer*, A. Ronzhin, R. Potapova, and N. Fakotakis, Eds. Cham: Springer International Publishing, 2015, pp. 129–136.

[4] X. Zhang, Y. Zou, and W. Shi, "Dilated convolution neural network with leakyrelu for environmental sound classification," in *2017 22nd International Conference on Digital Signal Processing (DSP)*, 2017, pp. 1–5.

[5] V. Bountourakis, L. Vrysis, and G. Papanikolaou, "Machine learning algorithms for environmental sound recognition: Towards soundscape semantics," in *Proceedings of the Audio Mostly 2015 on Interaction With Sound*, ser. AM '15. New York, NY, USA: Association for Computing Machinery, 2015. [Online]. Available: https://doi.org/10.1145/2814895.2814905

[6] S. Ntalampiras, I. Potamitis, and N. Fakotakis, "Automatic recognition of urban environmental sound events," 2008. [Online]. Available: https://api.semanticscholar.org/CorpusID:11792727

[7] Y. Zhan and T. Kuroda, "Wearable sensor-based human activity recognition from environmental background sounds," *Journal of Ambient Intelligence and Humanized Computing*, vol. 5, 02 2012.

[8] K. J. Piczak, "ESC: Dataset for Environmental Sound Classification," in *Proceedings of the 23rd Annual ACM Conference on Multimedia*. ACM Press, pp. 1015–1018. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2733373.2806390

[9] J. Hwang, M. Hira, C. Chen, X. Zhang, Z. Ni, G. Sun, P. Ma, R. Huang, V. Pratap, Y. Zhang, A. Kumar, C.-Y. Yu, C. Zhu, C. Liu, J. Kahn, M. Ravanelli, P. Sun, S. Watanabe, Y. Shi, Y. Tao, R. Scheibler, S. Cornell, S. Kim, and S. Petridis, "Torchaudio 2.1: Advancing speech recognition, self-supervised learning, and audio processing components for pytorch," 2023.

[10] Y.-Y. Yang, M. Hira, Z. Ni, A. Chourdia, A. Astafurov, C. Chen, C.-F. Yeh, C. Puhrsch, D. Pollack, D. Genzel, D. Greenberg, E. Z. Yang, J. Lian, J. Mahadeokar, J. Hwang, J. Chen, P. Goldsborough, P. Roy, S. Narenthiran, S. Watanabe, S. Chintala, V. Quenneville-Bélair, and Y. Shi, "Torchaudio: Building blocks for audio and speech processing," *arXiv preprint arXiv:2110.15018*, 2021.