Baby Cry Classification Using Ensemble Learning and Whisper Method Comparison

I Putu Yogi Prasetya Dharmawan [1]*, I Made Agus Dwi Suarjaya [2]*, Wayan Oger Vihikan [3]*
* Teknologi Informasi, Universitas Udayana
yogidharmawan6@gmail.com[1], agussuarjaya@it.unud.ac.id [2], oger_vihikan@unud.ac.id [3]

Article Info

Article history:

Received 2025-02-18
Revised 2025-02-20
Accepted 2025-02-21

Keyword:

Audio Classification,
Baby Cry Classification,
Ensemble Learning,
Whisper Model,
Machine Learning,

ABSTRACT

Baby cry classification is an important topic in Machine Learning, especially in the healthcare field, as crying is the primary form of communication for infants to convey their needs or conditions. Many inexperienced parents tend to interpret baby cries in a limited way, even though each cry has unique characteristics that represent specific needs such as hunger, discomfort, sleepiness, flatulence, and abdominal pain. With the advancement of technology, identification of baby cries can now be done automatically through AI-based applications, but the implementation is still limited. This study compares the performance of ensemble learning methods, namely Random Forest and XGBoost, with the Whisper model in classifying baby cries. The results show that the Whisper-small model has the best performance with precision 0.9115 and recall 0.9007, followed by XGBoost with slightly degraded performance after hyperparameter optimization. Random Forest showed the lowest performance among the three models. Transformer-based models such as Whisper-small proved to be superior in capturing the complex patterns of infant cries, compared to tree-based models. These findings indicate the great potential of accurate and reliable models to help parents understand the needs of infants more effectively, thereby improving the quality of infant care.

I. Introduction

Baby cry classification has become one of the significant topics in machine learning in various sectors, especially in healthcare. Infant crying is the primary form of infant communication used to convey certain states or feelings to parents and caregivers [1]. Inexperienced parents, especially those who are away from their families without guidance, tend to interpret baby's cries as a sign of hunger or tiredness. In fact, each type of cry has unique characteristics [2]. Baby cries such as hunger, urge to pass gas, urge to burp, discomfort, and sleepiness, each usually have different sound patterns [3]

As a first step in understanding infant communication, Priscillia Dunstan (2006) has identified five widely recognized types of infant cries, namely: “Neh” (hungry), ‘Eairh’ (wants to pass gas), ‘Eh’ (wants to burp), ‘Heh’ (uncomfortable), and ‘Owh’ (sleepy). This research serves as a reference for the identification of baby crying types. Along with technological advances, identification of baby crying sounds can now be done automatically [4].

This discovery encourages further research in automatic baby cry detection by utilizing artificial intelligence. Some previous studies have produced satisfactory results. Previous research utilized MFCC and CNN algorithms to classify baby cries based on amplitude as well as frequency of sound achieving an accuracy of 95% [5], classification of baby cries by comparing methods such as SVM, Random Forest, and Naïve Bayes with MFCC features, demonstrating that Random Forest achieved the highest accuracy of 84.4% [1], using a combination of CNN and RNN to capture spatial and temporal patterns that improve classification accuracy of 94.97% [6].

Beyond baby cry classification, ensemble learning methods have also been applied in other audio classification tasks. A study on piano and guitar chord recognition showed that XGBoost achieved 92.1% accuracy, outperforming Random Forest with 89.4% due to its ability to handle complex feature interactions [7]. Another study utilized Whisper's encoder with CNN and Bi-LSTM for speech-based classification, achieving 96.3% accuracy, demonstrating its effectiveness in extracting meaningful audio representations [8].

Although various studies have been conducted, no research has specifically compared ensemble learning methods (Random Forest and XGBoost) with Whisper for baby cry classification. Ensemble learning methods, such as Random Forest and XGBoost, have proven effective in feature-based audio classification, particularly when combined with feature extraction techniques like MFCC. Random Forest has also been successfully applied to baby cry classification, showing competitive results.

On the other hand, Whisper, originally developed for speech transcription, has demonstrated its ability to extract deep audio representations. In this study, Whisper is utilized solely as a feature extractor, transforming baby cry audio into meaningful representations, while the classification is performed using LSTM to capture temporal dependencies in the extracted features.

The choice of Random Forest and XGBoost in this study is based on their proven effectiveness in structured feature-based classification, making them suitable for comparison with deep learning approaches. Meanwhile, Whisper, as a Transformer-based model, offers an end-to-end approach that eliminates the need for manual feature extraction. This research aims to compare these approaches by evaluating their performance using precision, recall, and the confusion matrix to determine the most optimal method for baby cry classification.

II. Proposed Method

The methodology applied to this baby cry classification research can be seen in Figure 1. The computational process is carried out using the Python programming language and Google Colab with various libraries such as Pandas, Numpy, Scikit-learn, Matplotlib, librosa, and Tranformer.

This research begins with the collection of a dataset containing 2018 audio data from various sources, such as Kaggle, websites, and YouTube, which includes five categories of baby cries: tired, burping, belly pain, discomfort, and hungry. The labeling process is done using cosine similarity implemented through the librosa library to match the audio data with the appropriate labels. Preprocessing stages are performed to prepare the data before it is used in the model, by applying the MFCC process in frequency-based feature extraction for the ensemble learning model, while for the Whisper model, preprocessing includes resampling, padding, conversion to log-mel spectrogram, application of Conv1D layers, positional embedding, and encoder blocks.

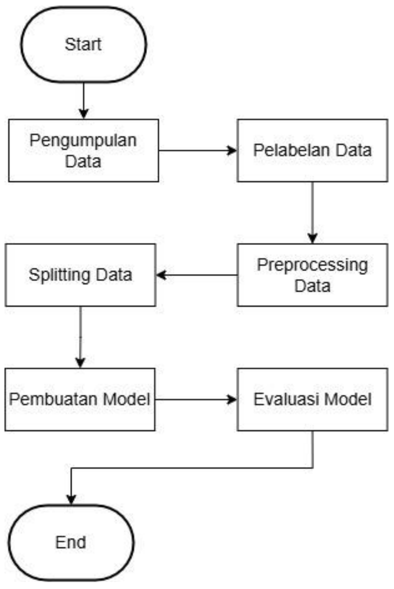

Figure 1. Flowchart of the research

After the preprocessing process is complete, the data is separated with a proportion of 80:20 for training and test data, where 80% will be processed for training and 20% for the testing process. After the data is processed and divided, this study uses five models for comparison, namely: Random Forest, Random Forest with hyperparameter optimization (using Random Search), XGBoost, XGBoost with hyperparameter optimization, and Whisper. Random Forest and XGBoost models are optimized with Random Search to find the best hyperparameter combination. Model evaluation is done using confusion matrix, focusing on precision and recall metrics, to determine the best model based on performance in classifying baby cries according to category.

A. *Data Collection*

The data collected for this study was done by collecting 2018 audio baby cries from various sources, namely Kaggle (457 data), websites (1531 data), and YouTube (30 data). This data was categorized into five categories of baby cries, namely discomfort, tired, burping, hungry, and belly pain. Data collection was done manually by downloading audio from Kaggle, the website, and YouTube using Python. The yt_dlp library was used for YouTube scraping. The dataset used in this study consists of 2018 baby cry audio samples collected from various sources: Kaggle (457 samples), websites (1531 samples), and YouTube (30 samples). The collected data was manually downloaded and categorized into five baby cry types: discomfort, tiredness, burping, hunger, and belly pain. The raw audio data was stored in (.wav) format. Data labeling was conducted based on metadata provided by the sources, followed by a verification process to ensure correct classification. Data obtained then went through a process of cleaning, labeling, and clustering according to the category of baby cries to support further analysis.

B. Data Labelling

The baby audio dataset that has been collected in this research will go through a labeling process before going to the next stage. The labeling process is done with the help of the Librosa library to calculate the cosine similarity value between the additional audio data and the reference data. The reference data is obtained from Kaggle, while the additional data is taken from various sources such as websites and YouTube. The complete flow of the data labeling process can be seen in Figure 2.

The labeling process starts by comparing each additional audio data with the reference data using cosine similarity. The results of the cosine similarity calculation are then saved in CSV format for further analysis. Based on the analysis, audio data that has a cosine similarity level above 80% with the reference data is considered valid and labeled according to the reference data category. After the data is saved, manual grouping is done to ensure that the audio data is in the appropriate folder according to its category.

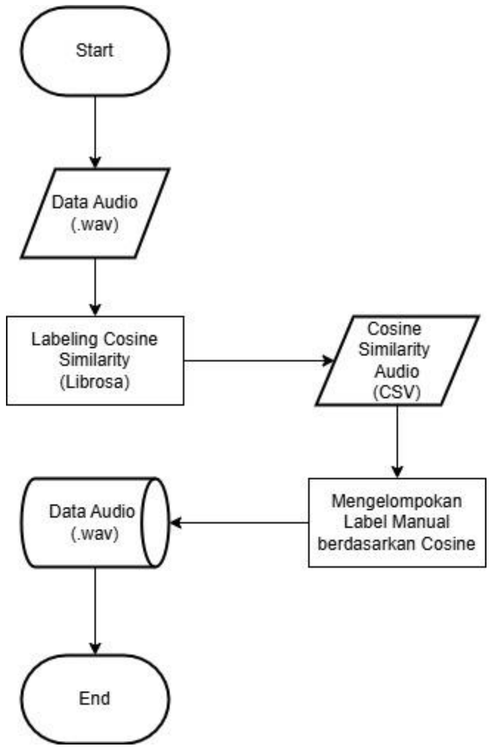
Start
Data Audio (.wav)
Labeling Cosine Similarity (Librosa)
Cosine Similarity Audio (CSV)
Mengelompokan Label Manual berdasarkan Cosine
Data Audio (.wav)
End

Figure 2. Flowchart of data labeling

For example, the results of cosine similarity calculation on baby audio data are shown in Table 1. In the table, the highest cosine similarity data from several categories is compared with the reference data. It can be seen that there are the same audio files in some categories, but with different cosine similarity values. This shows the need for manual validation to ensure the audio data is evenly distributed in each category.

TABLE I
COSINE SIMILARITY DATA

Index	Category	File Name	Similarity
1	Belly_pain	BabyCry306.wav	0.9124444
2	Burping	BabyCry321.wav	0.9108460
3	Discomfort	BabyCry321.wav	0.8696621
4	Tired	BabyCry307.wav	0.9109482
5	Hungry	BabyCry255.wav	0.9250045

C. Preprocessing Data

Feature extraction is a crucial stage in data preprocessing, especially in the classification of baby crying sounds. The preprocessing stage aims to obtain discriminative features from audio signals so that they can be used as input for machine learning algorithms [7]. In the context of audio signals, features can be extracted from both the time domain and frequency domain. Features from the time domain, such as zero-crossing rate, amplitude, and energy, although easy to calculate, tend to be less robust to disturbances and fluctuations. Therefore, features from the frequency domain, such as Mel-Frequency Cepstral Coefficients (MFCC), are more frequently used due to their ability to model the frequency characteristics of the signal [9] In this study, MFCC features were extracted directly from raw audio waveforms. Meanwhile, Whisper Encoder processes raw audio directly without prior feature extraction, as it internally computes the log-Mel spectrogram before passing it through convolutional layers and Transformer architectures.

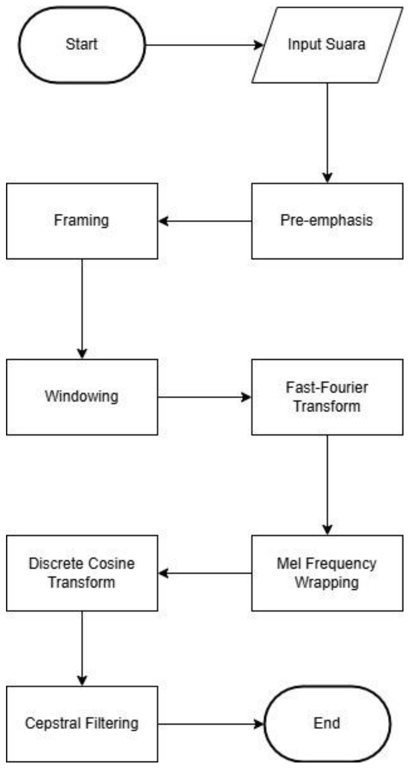

Figure 3. Flowchart of MFCC [11]

1) *MFCC*

MFCC is a feature extraction method, which is one of the effective methods to model the frequency characteristics of audio signals, including baby crying sounds [10]. MFCC is able to mimic human auditory perception by accentuating the more physiologically relevant frequency components. The feature extraction process using MFCC includes several stages shown in Figure 3 consisting of:

a. *Pre-emphasis*: Preserves high frequency components that are often weakened during the sound production process [11]. The equation for the pre-emphasis stage can be seen in equation 1.

$$y_n = x_n - \alpha \times x_{n-1}, 0{,}9 \le \alpha \le 1 \quad (1)$$

Where y_n is the signal resulting from the pre-emphasis process and x_n s the signal before pre-emphasis is applied.

b. *Framing*: Dividing the signal into small frames of 20-40 ms duration with 30%-50% overlapping [11].

c. *Windowing*: Reducing the aliasing effect using Hamming Window [9]. The equation for the windowing stage can be seen in equation 2.

$$y_n = x_n \times w_n, 0 \le n \le N-1 \quad (2)$$

Where N refers to the number of samples in each frame. y_n is the signal value that has gone through the windowing process, while x_n represents the signal value at the nth frame. Finally, w_n acts as a window function, with the type of window used is the Hamming Window, which can be calculated using equation 3.

$$w_n = 0{,}54 - 0{,}46\, cos\frac{2\pi n}{N-1}, 0 \le n \le N-1 \quad (3)$$

d. *Fast Fourier Transform* (FFT): Converts the signal from the time domain to the frequency domain[9]. The equation for the FFT stage can be seen in equation 4.

$$x_k = \sum_{n=0}^{N-1} xn \times e^{\frac{-2\pi kn}{N}}, n = 0 \le n \le N-1 \quad (4)$$

Where x_k is the frequency value in the signal, k denotes the audio frequency before the FFT. xn s the signal value at the nth time, and N is the total time samples.

e. *Mel Frequency Warping*: Converts the frequency spectrum to the Mel scale [11] The equation for the Mel Frequency Warping stage can be seen in equation 5.

$$Mel\,(f) = 2595\, log_{10}(1 + \frac{f}{700}) \quad (5)$$

Where f is the signal frequency

f. *Discrete Cosine Transform* (DCT): Produces MFCC coefficients in the form of acoustic vectors [11]. Produces MFCC coefficients in the form of acoustic vectors.

$$C_m = \sum_{k=1}^{k} (logS_k) \cos\left[\mathrm{m}\left(\mathrm{k} - \tfrac{1}{2}\right)\tfrac{\pi}{k}\right], m = 1,2,\dots,K \quad (6)$$

Where S_k is the result of Mel Frequency Warping output and K is the number of coefficients generated from Mel Scale Cepstral Coefficients.

g. *Cepstral Filtering*: Smoothing the final MFCC result to make it more optimal for further analysis [11]. The equation for the Cepstral Filtering stage can be seen in equation 7.

$$W_n = 1 + \frac{L}{2}\sin\left[\frac{\pi n}{L}\right] \quad (7)$$

Where L is the total cepstral coefficients, and n is the index of the cepstral coefficients.

2) *Whisper Encoder*

After explaining the use of MFCC in frequency-based feature extraction, this research also adopts Whisper Encoder as a modern approach that enables a more in-depth analysis of audio signal characteristics [8].

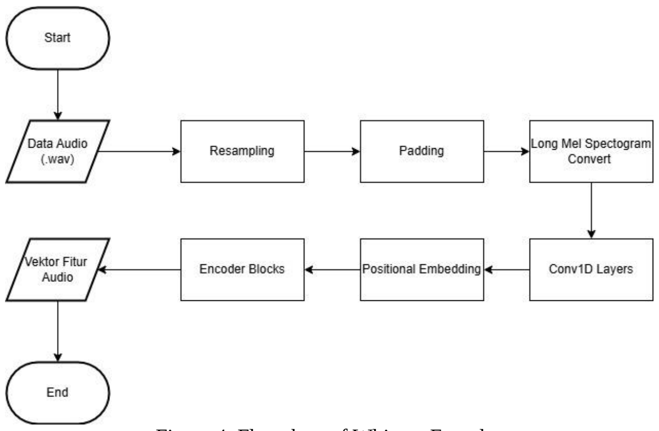

Figure 4. Flowchart of Whisper Encoder

The Whisper Encoder is an encoder-decoder-based model that functions as a feature extractor, capturing complex details of audio signals such as intensity, pitch, and duration. Unlike MFCC, which primarily focuses on spectral features, Whisper Encoder leverages Transformer-based attention mechanisms to extract contextual and hierarchical features from baby cry sounds [12]. Figure 4 shows the flow diagram of the Whisper Encoder work process which consists of:

a. Resampling**:** The sound signal is converted to 16 kHz [8].
b. Padding**:** The signal is equalized in duration to 30 seconds for input consistency [8].
c. Log-Mel Spectrogram**:** enerates a log-Mel spectrum with 80 channels, a window length of 25 ms, and a step of 10 ms. The spectrum values are then normalized in the range [-1, 1] [8].

d. *Conv1D Layers***:** Using two 1D convolution layers with GELU activation function [8].
e. Sinusoidal Positional Encoding**:** Adds positional encoding to recognize the data sequence [8].
f. Encoder Blocks**:** Uses a self-attention layer and a feedforward network to process the signal into a fixed dimensional vector (1 x 1500 x 512) [8].

D. Ensemble Learning

Ensemble learning combines predictions from multiple models to improve performance and accuracy in complex classification predictions, such as in the classification of baby crying sounds [13]. This approach reduces the variance and bias of individual models, making it more effective in handling variable and noise data [14]. In this study, the two ensemble techniques used are Random Forest and XGBoost, which have been proven effective for classification in many applications, including voice [15].

1) *Random Forest*

Random Forest can be defined as a method that combines a number of decision trees or ensembles by training randomly selected data, allowing the model to reduce overfitting and improve prediction stability [15]. In the process, each decision tree makes predictions independently, and the final result is determined by selecting the majority voting result from all trees [16].

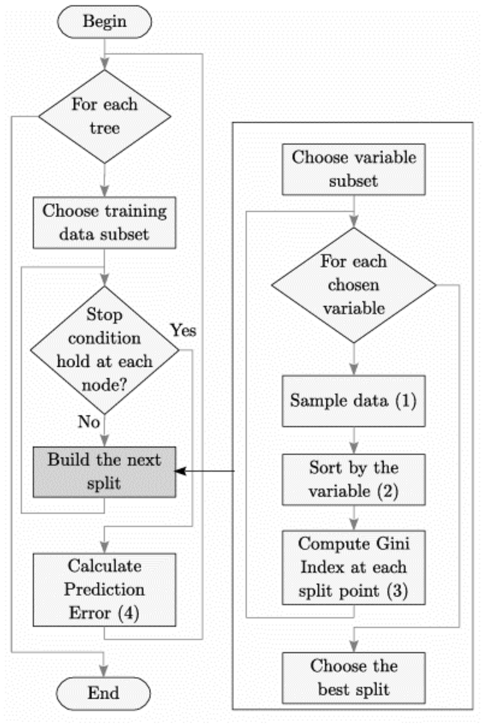

Figure 5. Random Forest workflow [17]

By using more trees, the model can handle data with noise better and provide more accurate predictions, which is especially important in the classification of variable baby crying sounds [18]. Figure 5 describes the Random Forest workflow which consists of several important steps, such as:

a. Begin: The Random Forest model starts by building multiple decision trees in parallel using a randomly selected subset of data (bootstrapping).
b. For each tree: The data is split by randomly selecting variables to build each tree. At each node, it is checked whether a stopping condition is met (e.g., the number of samples is too small or the tree depth is maximal). If not, the node will be further split based on the best variable.
c. Build the next split: If the stopping condition is not met, the process will select the best variable for splitting the data based on the Gini Index.
d. Calculate Prediction Error: After all the trees are formed, the prediction error is calculated based on the average prediction of all the trees.
e. End: The final prediction is obtained through majority voting (for classification) or averaging (for regression) of all trees in the ensemble.

TABLE II
RANDOM FOREST HYPERPARAMETER DEFAULT

Model Name	Hyperparameter Default
Random Search	n_estimators: 100, max_depth: 20, min_samples_split: 2, min_samples_leaf: 1

TABLE III
RANDOM FOREST RANDOM SEARCH OPTIMIZATION

Hyperparameter	Search Space
n_estimators	{5, 10, 15, 20, 50, 100, 200}
max_depth	{10, 15, 20, 25, 30, 35, 40, 50}
min_samples_split	{2, 3, 4, 5, 10}
min_samples_leaf	{1, 2, 4}

The baseline hyperparameters used aim to maintain a balance between bias and variance. The default hyperparameters, as shown in Table 2, allow the model to capture deeper patterns without being overly complex. The optimization process was carried out using Random Search to find the best combination of parameters that could improve model performance, with the hyperparameters presented in Table 3.

2) XGBoost

XGBoost is a boosting method that works by training models sequentially, where the new model focuses on correcting the errors made by the previous model [19]. This technique gives more weight to data that is difficult to classify, thus improving the accuracy of the model in identifying finer patterns [20].

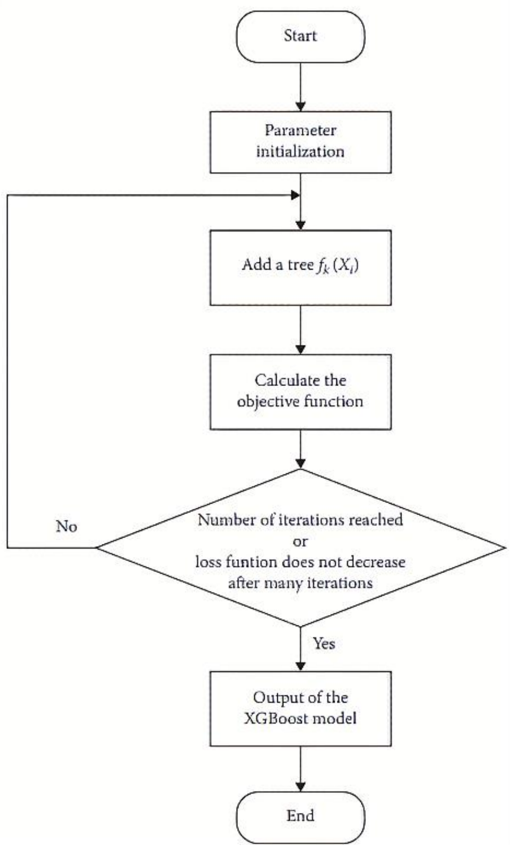
Start
Parameter initialization
Add a tree $f_k(X_i)$
Calculate the objective function
Number of iterations reached or loss funtion does not decrease after many iterations
No
Yes
Output of the XGBoost model
End

Figure 6. XGBoost workflow [21]

The advantages of XGBoost lie in its ability to handle imbalanced data, as well as its efficient regularization feature to avoid overfitting, which makes it suitable for use in the classification of baby crying sounds [21]. Figure 6 illustrates the workflow of XGBoost, which consists of the following steps:

a. Start: The process begins with the initialization of parameters such as the learning rate and the number of iterations to be performed.
b. Add a tree: XGBoost adds a new decision tree to correct errors from the previous tree, focusing on incorrect predictions.
c. Calculate the Objective Function: The objective function is used to measure the prediction error, usually using a loss function such as squared error.
d. Number of iterations reached or loss function does not decrease: The iteration process is stopped if the number of iterations is reached or the loss function does not decrease significantly.
e. Output of the XGBoost model: Once the iteration is complete, the model is ready to be used for prediction.
f. End: The XGBoost model is complete and ready for use.

TABLE IV
XGBOOST HYPERPARAMETER DEFAULT

Model Name	Hyperparameter Default
XGBoost	n_estimators: 100, max_depth: 5, learning_rate: 0.1

TABLE V
XGBOOST RANDOM SEARCH OPTIMIZATION

Hyperparameter	Search Space
n_estimators	{20, 50, 100}
max_depth	{1, 3, 5}
learning_rate	{0.001, 0.01, 0.1}

The baseline hyperparameters aim to capture more complex patterns without causing overfitting, as shown in Table 4. Optimization was performed using Random Search to obtain the best hyperparameter combination, as presented in Table 5.

E. Whisper

Whisper is an automatic speech recognition (ASR) model based on a Transformer architecture, developed by OpenAI in September 2022 [12]. Unlike previous models such as wav2vec 2.0, which rely solely on unlabeled audio data, Whisper is trained on a large-scale dataset comprising 680,000 hours of multilingual audio, including cross-lingual translation data. This approach enhances the model's robustness and performance across various environmental conditions, enabling tasks such as voice activity detection, language identification, transcription, and automatic translation [8].

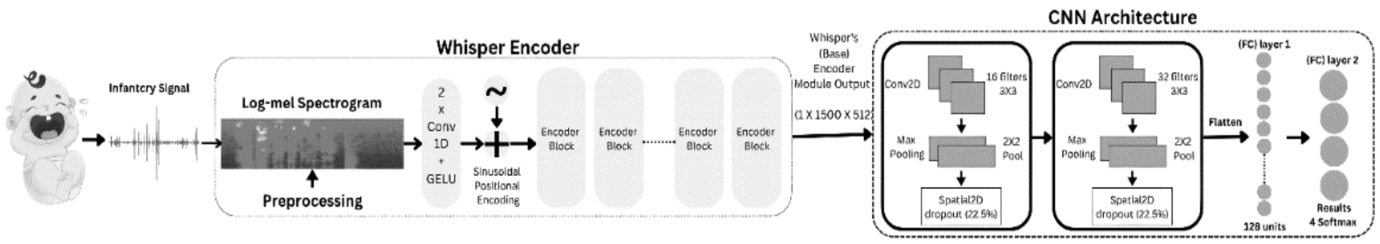

Figure 7. Whisper workflow [8]

The Whisper process begins with preprocessing, where audio is resampled to 16 kHz and padded to a duration of 30 seconds. Next, the Whisper Encoder extracts features using Log-Mel Spectrogram, Conv1D layer, sinusoidal positional encoding, and block encoder, resulting in a feature vector with fixed dimensions (1 x 1500 x 512). Figure 7 illustrates the working architecture of Whisper, which consists of a Whisper encoder followed by a CNN as a classifier [8]. Figure 7, illustrates the Whisper workflow, where the Whisper Encoder acts as a feature extractor. After this process, the extracted features are fed into an LSTM classifier, replacing the CNN used in the previous reference model.

In this research, we will apply the use of LSTM as a classifier in the process of classifying baby crying sounds, replacing CNN as a classifier in the reference model. The classifier in this study uses three LSTM layers with a hidden state size of 512 to capture the temporal pattern of the encoder output data. Afterward, there is a dropout layer (rate 0.2) to prevent overfitting, followed by one fully connected layer for dimension conversion, and ending with a softmax layer to generate classification probabilities. Since baby cries exhibit sequential dependencies over time, LSTM is a more suitable classifier than CNN, which primarily captures spatial patterns. Unlike Transformer models, LSTM is computationally more efficient while still effectively modeling long-term temporal dependencies.

F. Hperparameter Optimization

Random Search is a hyperparameter optimization technique that randomly selects combinations of various parameters from several predefined parameters. Unlike Grid Search that evaluates each combination systematically, Random Search is more efficient because it emphasizes random exploration to find the optimal parameters [7]. This method is effective in optimizing models such as Random Forest and XGBoost, which are used in this study. With Random Search, the optimization process becomes more efficient and flexible both in terms of time speed and computational resources, especially in dealing with complex data such as baby cries [22].

G. K-Fold Cross Validatin

K-Fold Cross Validation is interpreted as an evaluation technique on the model created by dividing or separating the dataset into K separate parts. Each part is used in turn as test data, while the other part is used for training. This reduces estimation bias and results in more stable and generalizable performance estimates. In this research, a 10-fold cross-validation method is used, where the dataset is divided or separated into 10 separate parts. Each part in turn acts as test data, while the other nine parts are used to train the model. The performance results of each iteration are averaged to obtain a more reliable and representative prediction error estimate against new data [1].

H. Confusin Matrix

Confusion Matrix is an evaluation carried out with the aim of evaluating the performance of a baby cry classification model by comparing the predictions generated and the actual class label. This matrix presents the results in a two-dimensional form, which indexes the actual class and the predicted class which can be seen in Table 6 [23].

TABLE VI
CONFUSION MATRIX

		Prediction Class	
		Positif	Negatif
Actual Class	Positive	TP	FN
	Negative	FP	TN

After using the Confusion Matrix, the precision and recall metrics are calculated to measure the quality of the model's predictions which can be seen in equations 8 and 9.

$$Precission = TP/(TP + FP) \qquad (8)$$

$$Recall = TP/(TP + FN) \qquad (9)$$

In the context of classification model evaluation, the terms TP, FP, FN, and TN have the following meanings:

a. TP (True Positive): A true positive prediction (the model predicts a positive, and it is indeed a positive).
b. FP (False Positive): False positive prediction (the model predicts positive, but it is actually negative).
c. FN (False Negative): False negative prediction (the model predicts negative, but it is actually positive).
d. TN (True Negative): A true negative prediction (the model predicts negative, and is actually negative).

In this study, precision and recall are chosen as the primary evaluation metrics because they provide insights into the model's ability to correctly classify baby cries while minimizing false positives and false negatives.

III. RESULTS AND DISCUSSION

The composition of the amount of data in the data splitting process with a ratio of 80:20 with a total data set of 2018 data is shown in Table 7 and 8 by obtaining data from Kaggle [24], websites, and YouTube. The collected data is in (.wav) format and will be labeled.

The main data is obtained from Kaggle with the categories of belly_pain, burping, discomfort, hungry, and tired. The total data collected was 457. An example of the data can be seen in Table 7.

TABLE VII
KAGGLE DATA EXAMPLE

Category	Kaggle Data
Belly_pain	549a46d8-9c84-430e-ade8-97eae2bef787-1430130772174-1.7-m-48-bp.wav
Burping	5afc6a14-a9d8-45f8-b31d-c79dd87cc8c6-1430757039803-1.7-m-48-bu.wav
Discomfort	10A40438-09AA-4A21-83B4-8119F03F7A11-1430925142-1.0-f-26-dc.wav
Tired	02c3b725-26e4-4a2c-9336-04ddc58836d9-1430726196216-1.7-m-04-hu.wav
Hungry	03ADDCFB-354E-416D-BF32-260CF47F7060-1433658024-1.1-f-04-ti.wav

Additional data is obtained from several websites and Youtube with a total of 1,531 website data and 30 Youtube data that will be labeled and grouped into baby crying categories. Sample data can be seen in Table 8.

TABLE VIII
WEBSITE AND YOUTUBE DATA EXAMPLE

Website Data	Youtube Data
067015623-human-baby-baby-1-month-old-in	3 month baby cry
067015624-human-baby-baby-1-month-old-in	Adele crying & a shot of her feet [-OIY3tRRGwg]
067015626-human-baby-baby-1-month-old-in	Ariel Crying, 5 weeks
067015627-human-baby-baby-1-month-old-in	Baby crying
067015628-human-baby-baby-1-month-old-in	Baby Fake crying

Raw data obtained from YouTube and several other websites are labeled using cosine similarity from the librosa library, with data from Kaggle as a reference. The process starts by comparing the similarity between audio files based on MFCC (Mel-frequency Cepstral Coefficients) features. Each pair of audio files is compared, and only pairs with similarity values above the 0.8 threshold will be processed further.

TABLE IX
COSINE SIMILARITY DATA EXAMPLE

Kaggle Data	Random Data	Similarity
10A40438-09AA-4A21-83B4-8119F03F7A11-1430925142-1.0-f-26-dc.wav	Baby Cry 6	0.8053
10A40438-09AA-4A21-83B4-8119F03F7A11-1430925142-1.0-f-26-dc.wav	Baby Cry 7	0.8090
10A40438-09AA-4A21-83B4-8119F03F7A11-1430925142-1.0-f-26-dc.wav	Baby Cry 19	0.8015

The results of this comparison are used to categorize the audio files into the appropriate baby crying categories. Table 9 shows the results of the cosine similarity calculation between pairs of audio files obtained from kaggle and random data that has been collected from YouTube and websites, with similarity values recorded only if above 80%. This data is then analyzed and saved in CSV format for the labeling process.

Next, these labeling results were analyzed to map the audio files to the right categories, with a manual process performed to move the files to the appropriate categories. After the clustering is complete, the labeled data will be prepared for the preprocessing stage, which includes MFCC and Whisper Encoder feature extraction before being used in modeling using Ensemble Learning algorithms such as Random Forest and XGBoost and Whisper.

In the preprocessing stage for Ensemble Learning models, which includes Random Forest and XGBoost algorithms, MFCC is used in frequency-based feature extraction due to its ability to describe the frequency characteristics of sound.

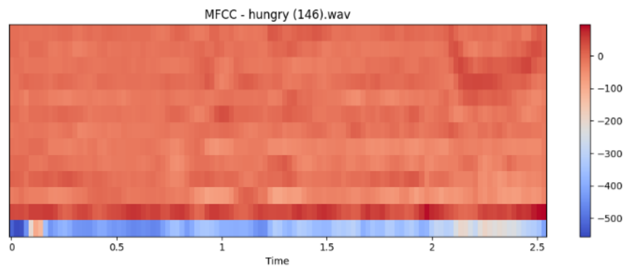

Figure 9. Extraction Feature MFCC result

In this experiment, MFCC features are extracted from audio files using standard parameters, i.e. 13 coefficients. The preprocessing process includes the reading of the audio file as well as the MFCC extraction presented in Figure 9. After extraction, the resulting MFCC features are processed with padding to ensure uniform feature length across the dataset. The figure shows the frequency energy distribution pattern over time, with the vertical axis depicting the MFCC coefficients and the horizontal axis depicting the audio duration.

In the Whisper model, a series of preprocessing steps were performed, including resampling the audio to a frequency of 16 kHz to ensure uniformity with the Whisper model, as well as converting the audio into Mel Spectrogram form which was then used as input into the model. Whisper also applies amplitude normalization and padding to ensure the audio data has a consistent length.

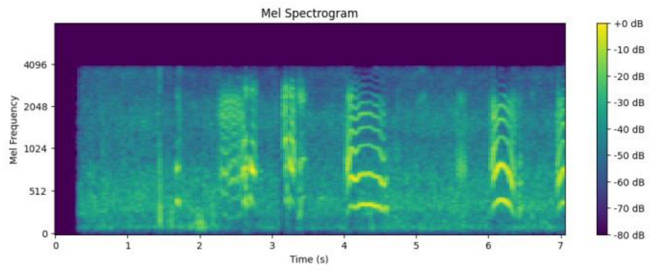

Figure 10. Whisper Encoder result

All these preprocessing steps are done automatically using WhisperProcessor from the Whisper library. The preprocessing results in a more stable and representative embedding of the raw audio, which is then used in further processing. Figure 10 shows the Mel Spectrogram visualization of the processed audio file, as well as information about the resulting audio embedding, including embedding dimensions and vector values.

After preprocessing, a data distribution analysis was conducted to determine the proportion of data in each category. The analysis results show that the dataset used is imbalanced, which means that some categories have more data than others. Details of the amount of data for each category are shown in Table 10.

TABLE X
DATA DISTRIBUTION

Category	Number of file
Belly pain	375
Hungry	624
Discomfort	336
Burping	251
Tired	432

It should be noted that the dataset used in this study tends to be unbalanced, with some categories having a much larger distribution of data than others. For example, the category “Hungry” has 624 data, while “Burping” only has 251 data. This imbalance could potentially affect the performance of the model, as it tends to recognize categories with more data better. However, this study does not apply any data balancing techniques, such as oversampling or augmentation, in order to evaluate the model’s ability to learn from naturally imbalanced data.

The dataset is then separated with a proportion of 80:20 for training and test data, where 80% will be processed for training and 20% for the testing process. The results of data separation or division can be seen in Table XI.

TABLE XI
DATA DISTRIBUTION TRAINING AND TESTING

Category	Distribution
Training	(1614, 13, 300)
Testing	(404, 13, 300)
Training label	1614
Testing label	404

The training and testing data contains 3 parameters which show the number of data distributions, MFCC coefficients, and Max Length used. After the data sharing process, the model was trained and tested using three main algorithms, namely Random Forest, XGBoost, and Whisper. The comparison of model performance based on precision and recall values is presented in Table 12.

TABLE XII
MODEL PERFORMANCE RESULT BASELINE

Model Name	Precision (Baseline)	Recall (Baseline)
Random Forest	0.8775	0.8527
XGBoost	0.8944	0.8886
Whisper-small	0.9115	0.9007

TABLE XIII
MODEL PERFORMANCE RANDOM SEARCH

Model Name	Precision (Random Search)	Recall (Random Search)	Best Hyperparameter (Random Search)
Random Forest	0.8775	0.8527	n_estimators: 50, max_depth: 4, min_samples_split: 1 min_samples_leaf: 35
XGBoost	0.8738	0.8688	n_estimators: 20, max_depth: 5, learning_rate: 0.1
Whisper-small	-	-	-

The baseline hyperparameters were designed to capture more complex patterns without causing overfitting, as shown in Table XII. These default settings provide a balance between model complexity and its generalization ability. However, to further improve performance, hyperparameter optimization was conducted using Random Search, as presented in Table XIII. After obtaining the best hyperparameters from Random Search, the model was further evaluated using 10-fold Cross-Validation (10-CV) to ensure its robustness.

The test results show that Whisper-small outperformed the other algorithms, both before and after hyperparameter optimization. Based on precision and recall evaluations, Whisper-small achieved a precision of 0.9115 and a recall of 0.9007, indicating its superior ability to recognize baby crying sound patterns compared to other models.

Meanwhile, Random Forest maintained stable performance with a precision of 0.8775 and a recall of 0.8527, without significant changes after hyperparameter optimization. This indicates that the baseline hyperparameters were already well-balanced for this task, and there was no indication of either overfitting or underfitting in this model.

On the other hand, XGBoost experienced a decline in performance after optimization, with a precision of 0.8738 and a recall of 0.8688. This suggests the possibility of overfitting, where the model became too specialized to the training data, leading to reduced performance when tested on new data. However, there was no indication of underfitting in any of the models, as all of them were able to recognize patterns effectively from the baseline itself.

Overall, these results confirm that Whisper-small is the most effective model for baby crying sound classification in this study. Its ability to extract deeper audio representations and generate stable embeddings provides an advantage over Ensemble Learning models such as Random Forest and XGBoost. However, despite Whisper-small's strong performance, the issue of data imbalance remains a challenge that should be addressed in future research to further improve model generalization.

IV. CONCLUSION

This study aims to classify baby crying sounds using the Ensemble Learning method (Random Forest and XGBoost) and the Whisper model. The results indicate that the Whisper-small model achieves the best performance, with a precision of 0.9115 and recall of 0.9007, demonstrating its superior ability to capture voice features compared to Ensemble Learning methods. Whisper-small excels in recognizing baby crying sound patterns due to its robust architecture trained on large-scale audio data, allowing it to perform well under various acoustic conditions.

On the other hand, Random Forest maintained stable performance, with no significant changes after hyperparameter optimization, indicating that its baseline parameters were already well-balanced for this task. XGBoost, however, experienced a slight performance drop after hyperparameter tuning, suggesting a tendency to overfit when adjusting certain parameters. To ensure a robust evaluation, Cross-Validation (CV) was applied only to the best hyperparameters obtained from Random Search, validating the model's generalization ability and preventing overfitting.

An analysis of the data distribution also revealed class imbalance, which may have influenced the overall model performance. Despite this challenge, Whisper-small remained the most effective model for baby crying sound classification. These findings provide valuable insights into the challenges of infant voice classification, particularly regarding data imbalance and hyperparameter sensitivity in Ensemble Learning models.

For future research, exploring alternative deep learning models tailored for audio classification such as CNN-based models (e.g., VGGish, YAMNet), sequence modeling approaches (e.g., RNN/LSTM), or self-supervised learning models (e.g., Wav2Vec2, HuBERT) could provide further performance improvements. Additionally, integrating advanced audio preprocessing techniques and domain adaptation strategies may enhance robustness in real-world applications.

REFERENCES

[1] P. A. Riadi, M. R. Faisal, D. Kartini, R. A. Nugroho, D. T. Nugrahadi, and D. B. Magfira, "A Comparative Study of Machine Learning Methods for Baby Cry Detection Using MFCC Features," *Journal of Electronics, Electromedical Engineering, and Medical Informatics*, vol. 6, no. 1, pp. 73–83, Jan. 2024, doi: 10.35882/JEEEMI.V6I1.350.

[2] S. Vesangi and S. Reddy Regatte, "A Novel Approach to Predict the Reason for Baby Cry using Machine Learning," *International Journal of Computer Science Trends and Technology (IJCST)*, vol. 10, [Online]. Available: www.ijcstjournal.org

[3] T. H. Rochadiani, "Pendekatan Transfer Learning Untuk Klasifikasi Tangisan Bayi Dengan Imbalance Dataset," *The Indonesian Journal of Computer Science*, vol. 13, no. 2, Apr. 2024, doi: 10.33022/IJCS.V13I2.3834.

[4] S. A. Younis, D. Sobhy, and N. S. Tawfik, "Evaluating Convolutional Neural Networks and Vision Transformers for Baby Cry Sound Analysis," *Future Internet*, vol. 16, no. 7, Jul. 2024, doi: 10.3390/fi16070242.

[5] C. A. A. Soemedhy, D. P. Martiyaningsih, and V. A. Kurniawan, "Klasifikasi Tangisan Bayi Klasifikasi Tangisan Bayi Berdasarkan Amplitudo Frekuensi Suara Menggunakan Algoritma MFCC dan CNN," *Jurnal Teknik Industri, Sistem Informasi dan Teknik Informatika*, vol. 1, no. 1, pp. 39–48, Oct. 2022, Accessed: Dec. 13, 2024. [Online]. Available: https://ejournal.ubibanyuwangi.ac.id/index.php/jurnal_tinsika/article/view/17

[6] T. Nadia Maghfira, T. Basaruddin, and A. Krisnadhi, "Infant cry classification using CNN - RNN," *J Phys Conf Ser*, vol. 1528, no. 1, p. 012019, Jun. 2020, doi: 10.1088/1742-6596/1528/1/012019.

[7] J. Elektronik Ilmu Komputer Udayana, I. Dewa Agung Adwitya Prawangsa, and A. Eka Karyawati, "Perbandingan Metode Ensemble Learning Random Forest Dan Adaboost Pada Pengenalan Chord Instrumen Piano Dan Gitar," *JELIKU (Jurnal Elektronik Ilmu Komputer Udayana)*, vol. 12, no. 4, pp. 809–816, May 2024, doi: 10.24843/JLK.2024.V12.I04.P07.

[8] M. Charola, A. Kachhi, and H. A. Patil, "Whisper Encoder features for Infant Cry Classification," in *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, International Speech Communication Association, 2023, pp. 1773–1777. doi: 10.21437/Interspeech.2023-1916.

[9] K. Makna *et al.*, "Klasifikasi Makna Tangisan Bayi Menggunakan CNN Berdasarkan Kombinasi Fitur MFCC dan DWT," *JATISI (Jurnal Teknik Informatika dan Sistem Informasi)*, vol. 8, no. 2, pp. 599–610, Jun. 2021, doi: 10.35957/JATISI.V8I2.470.

[10] N. F. Muhammad, R. Dewan, J. Pusppanathan, and F. A. Suryanata, "Baby Crying Sound Classification using Convolutional Neural Network," *Journal of Human Centered Technology*, vol. 3, no. 1, pp. 67–74, Feb. 2024, doi: 10.11113/HUMENTECH.V3N1.66.

[11] S. Y. Yusdiantoro and T. B. Sasongko, "Implementasi Algoritma MFCC dan CNN dalam Klasifikasi Makna Tangisan Bayi," *Indonesian Journal of Computer Science*, vol. 12, no. 4, Aug. 2023, doi: 10.33022/IJCS.V12I4.3243.

[12] M. Charola, S. Rathod, and H. A. Patil, "Robustness of Whisper Features for Infant Cry Classification," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Springer Science and Business Media Deutschland GmbH, 2023, pp. 421–433. doi: 10.1007/978-3-031-48312-7_34.

[13] N. S. F. Putri, A. P. Wibawa, H. Al Rasyid, A. Nafalski, and U. R. Hasyim, "Boosting and bagging classification for computer science journal," *International Journal of Advances in Intelligent Informatics*, vol. 9, no. 1, pp. 27–38, Mar. 2023, doi: 10.26555/ijain.v9i1.985.

[14] A. Maiti, C. Dutta, J. S. Banerjee, and P. Sarigiannidis, "Ai For Infant Well-Being: Advanced Techniques In Cry Interpretation And Monitoring," *Journal of Mechanics of Continua and*

Mathematical Sciences, vol. 19, no. 2, pp. 39–65, 2024, doi: 10.26782/jmcms.2024.02.00003.

[15] Y. Zayed, A. Hasasneh, and C. Tadj, "Infant Cry Signal Diagnostic System Using Deep Learning and Fused Features," *Diagnostics*, vol. 13, no. 12, Jun. 2023, doi: 10.3390/diagnostics13122107.

[16] C. Ji, T. B. Mudiyanselage, Y. Gao, and Y. Pan, "A review of infant cry analysis and classification," *EURASIP J Audio Speech Music Process*, vol. 2021, no. 1, pp. 1–17, Dec. 2021, doi: 10.1186/S13636-021-00197-5/FIGURES/5.

[17] N. Kunhare, R. Tiwari, and J. Dhar, "Particle swarm optimization and feature selection for intrusion detection system," *Sadhana - Academy Proceedings in Engineering Sciences*, vol. 45, no. 1, pp. 1–14, Dec. 2020, doi: 10.1007/S12046-020-1308-5/FIGURES/12.

[18] G. Ashari Rakhmat and W. Mutohar, "MIND (Multimedia Artificial Intelligent Networking Database Prakiraan Hujan menggunakan Metode Random Forest dan Cross Validation," *Journal MIND Journal | ISSN*, vol. 8, no. 2, pp. 173–187, 2023, doi: 10.26760/mindjournal.v8i2.173-187.

[19] V. R. Joshi, K. Srinivasan, P. M. D. R. Vincent, V. Rajinikanth, and C. Y. Chang, "A Multistage Heterogeneous Stacking Ensemble Model for Augmented Infant Cry Classification," *Front Public Health*, vol. 10, p. 819865, Mar. 2022, doi: 10.3389/FPUBH.2022.819865/BIBTEX.

[20] C. Y. Chang, S. Bhattacharya, P. M. D. Raj Vincent, K. Lakshmanna, and K. Srinivasan, "An Efficient Classification of Neonates Cry Using Extreme Gradient Boosting-Assisted Grouped-Support-Vector Network," *J Healthc Eng*, vol. 2021, no. 1, p. 7517313, Jan. 2021, doi: 10.1155/2021/7517313.

[21] K. Gu, J. Wang, H. Qian, and X. Su, "Study on Intelligent Diagnosis of Rotor Fault Causes with the PSO-XGBoost Algorithm," *Math Probl Eng*, vol. 2021, 2021, doi: 10.1155/2021/9963146.

[22] K. Danach, H. Kanj, K. Hamze, and I. Moukadem, "Optimizing Learning-Based Combinatorial Optimization Algorithms: Advanced Hyperparameter Techniques and Real-World Applications," *Nanotechnol Percept*, vol. 20, no. S15, pp. 2996–3017, Dec. 2024, doi: 10.62441/NANO-NTP.VI.4434.

[23] M. Dewi Renanti, A. Buono, K. Priandana, and S. Hartono Wijaya, "Evaluating Noise-Robustness of Convolutional and Recurrent Neural Networks for Baby Cry Recognition," 2024. [Online]. Available: www.ijacsa.thesai.org

[24] "Infant cry audio corpus." Accessed: Jan. 29, 2025. [Online]. Available: https://www.kaggle.com/datasets/warcoder/infant-cry-audio-corpus