Experimental Modeling of Face Emotion Recognition Using Machine Learning Classification (SVM, KNN, Random Forest) and Deep Learning CNN

Shane Ardyanto Baskara[1*], Nina Setiyawati[2]

[1,2]Department of Informatics Engineering, Faculty of Information Technology, Satya Wacana Christian University, Salatiga, Central Java, Indonesia
E-mail: [1*]shanebaskara30@gmail.com, [2]nina.setiyawati@uksw.edu

(Received: 15 Apr 2025, revised: 9 May 2025, 27 May 2025, accepted: 28 May 2025)

Abstract

Facial Emotion Recognition (FER) is a technology that analyzes facial expressions to detect emotions, playing a growing role in psychology and Human-Computer Interaction. In Indonesia, mental health issues are rising, with emotional disorders increasing from 6.0% in 2013 to 9.8% in 2018. Over 19 million people aged 15+ were affected in 2018, a number likely worsened by the COVID-19 pandemic. Given the urgency of early detection, FER offers a non-invasive method to help identify mental health issues. It can support timely intervention and promote psychological well-being, especially in under-resourced settings. This study compares several Machine Learning (ML) and Deep Learning (DL) models—SVM, K-Nearest Neighbor, Random Forest, and Convolutional Neural Networks (CNN)—to classify facial emotions. The dataset used is the Facial Expression Recognition dataset by Jonathan Oheix from Kaggle. Images were preprocessed and used to train and evaluate each model. Traditional ML models relied on extracted features, while CNN learned features directly from images. Results show that CNN achieved the highest accuracy among the tested models. This suggests that FER, especially with CNN, can be a useful tool for early detection of emotional disorders in mental health contexts.

Keywords: Facial Emotion Recognition, Machine Learning, Deep Learning, Mental Health, Facial Expressions.

I. INTRODUCTION

This study conducts facial emotion recognition modelling using several machine learning and deep learning classification algorithms. Facial Emotion Recognition (FER) is a technology used to analyze emotions through various sources such as images and videos. Facial expressions are a form of non-verbal communication that provide cues about human emotions, contributing 55% to emotion recognition [1]. The interpretation of emotional expressions has become a major focus in psychological research and Human-Computer Interaction. Due to the important role of emotion recognition, the exploration of its technological applications is actively pursued [2].

Machine learning is a branch of Artificial Intelligence (AI) that focuses on developing systems capable of learning from data without the need for explicit programming by humans [3]. Meanwhile, Deep Learning (DL) is a subfield of Machine Learning that develops algorithms to model data abstractions at a higher level by using a series of nonlinear transformation functions arranged in a deep and layered structure [4].

The modelling experiment utilized several Machine Learning (ML) classification algorithms, including: (1) Support Vector Machine (SVM), which uses a hypothesis space in the form of linear functions on high-dimensional features, trained using optimization-based algorithms to produce classification or regression models [5]; (2) K-Nearest Neighbor (KNN), which can perform classification without requiring modelling of data distribution, making it suitable for datasets of varying sizes [6]; and (3) Random Forest (RF), an ensemble learning algorithm that uses a bootstrap (bagging) approach to build predictive models [7]. The Deep Learning algorithm used is Convolutional Neural Networks (CNN), which is capable of learning complex features to achieve high accuracy in recognizing expressions [4].

The model was developed using the Python programming language, which is commonly used for data analysis because it supports various programming paradigms, including object-oriented, functional, and rule-based programming [8]. The aim of this study is to conduct a FER modelling experiment and compare the performance of ML models such as SVM, KNN, RF, and the deep learning model CNN in detecting emotions based on facial expressions. The resulting FER

DOI: 10.34148/teknika.v14i2.1232

model is expected to serve as an alternative tool to assist in psychological treatment and maintaining dynamic mental health [9].

In March 2022, the Indonesian Psychiatric Association (PDSKJI) reported that 82.5% of users of self-assessment services experienced psychological problems, an increase from 80.4% in 2020 and 70.7% in 2019. These psychological issues included anxiety, depression, and psychological trauma. This data indicates that mental health problems in Indonesia are becoming increasingly alarming and require more serious attention and intervention.

Mental health is one of the crucial aspects contributing to individual and societal well-being. However, data shows that the prevalence of mental disorders in Indonesia continues to rise each year. According to the Ministry of Health of the Republic of Indonesia (Kemenkes), approximately one in ten people in Indonesia experiences psychological disorders. The prevalence of emotional mental disorders also showed a significant increase, from 6.0% in 2013 to 9.8% in 2018 among the adult population [10]. In 2018, the Basic Health Research (Riskesdas) revealed that more than 19 million individuals aged over 15 in Indonesia suffered from emotional psychological disorders. In addition, a survey by Khamelia and Terry [11] indicated that the COVID-19 pandemic also contributed to the rise in psychological problems.

II. LITERATURE REVIEW

In the research journal by Yousif Khaireddin and Zhuofa Chen, Facial Emotion Recognition (FER) is identified as a crucial element in human-computer interaction, particularly in clinical practice and behavioral assessment, helping to detect mental health conditions such as depression, anxiety, and mood disorders. However, the accuracy and robustness of FER models remain challenging due to the diversity of human faces and variations in images, such as differences in facial poses and lighting conditions. Deep learning-based models, particularly Convolutional Neural Networks (CNNs), have shown great potential due to their automatic feature extraction capabilities and computational efficiency. One recent study adopted the VGGNet architecture and achieved the highest accuracy of 73.28% on the FER2013 dataset without additional training data. The model's success was attributed to hyperparameter tuning and the exploration of optimization methods. With such high accuracy, CNNs hold great potential for integration into mental health detection systems, providing automated insights into a person's emotional state that can support early diagnosis and more efficient mental health management [12].

The study by Simcock et al. explores the relationship between facial emotion recognition ability and mental health in early adolescents. The study involved 40 twelve-year-old children who assessed symptoms of anxiety, depression, and somatization through a mental health questionnaire while also completing a facial emotion recognition task. The results showed that adolescents with higher psychological symptoms tended to exhibit biases in emotion processing, particularly toward negative expressions such as anger and fear. This bias may indicate underlying mental health issues from an early age, although the neurological mechanisms behind it still require further research. These findings highlight how facial emotion recognition can serve as a tool to detect early signs of mental disorders, which is important for faster diagnosis and intervention in adolescents [13].

Facial Emotion Recognition (FER) using artificial intelligence (AI) technology has shown significant potential in assisting medical professionals, especially in the field of mental and emotional health. Previous studies have demonstrated that FER systems utilizing machine learning and deep learning algorithms, such as Convolutional Neural Networks (CNNs), can accurately detect facial expressions, which can then be used as supplementary data for diagnosing mental disorders or emotional stress. By integrating FER technology into medical systems, healthcare professionals can gain real-time insights into patients' emotional conditions, which is crucial for early intervention and clinical decision-making. For example, a CNN-based model trained using the FER2013 dataset was able to achieve high accuracy in emotion classification with carefully tuned parameters, showing that this technology can be further adapted to help medical professionals understand and respond to patients' conditions. These advancements open up great opportunities for the development of more inclusive and responsive AI-based mental health systems [14].

III. RESEARCH METHODOLOGY

This research adopts a systematic approach to explore Face Emotion Recognition (FER) using machine learning and deep learning techniques. The process includes: 1) Introduction, 2) Literature Review, 3) Methodology, 4) Experiments and Results, 5) Discussion, and 6) Conclusion, as shown in Figure 1.

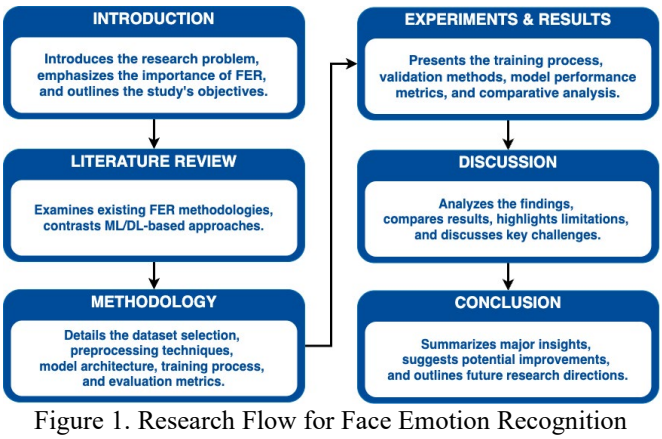

Figure 1. Research Flow for Face Emotion Recognition

This study compares the performance of several Machine Learning models—Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and Random Forest—with a Deep Learning model (Convolutional Neural Network or CNN) in the context of Facial Expression Recognition (FER). The

DOI: 10.34148/teknika.v14i2.1232

process begins with data collection and preprocessing. Facial expression images are first converted to grayscale, resized to 48×48 pixels, and stored using Python's pickle module, which handles the serialization (pickling) and deserialization (unpickling) of Python objects for efficient storage and reuse [15].

To enhance model performance and reduce the influence of irrelevant features, pixel value normalization is applied [16]. Label encoding is used to convert categorical emotion labels into numerical values, facilitating the training of machine learning models [17]. Principal Component Analysis (PCA) is employed for dimensionality reduction to reduce complexity, speed up training, and minimize the risk of overfitting [18]. Feature standardization is also performed using StandardScaler to ensure consistent feature scaling, which is essential for many learning algorithms [19].

Following preprocessing, the dataset is split into training and testing sets. Machine Learning models are optimized using Grid Search to determine the best hyperparameter combinations. Model evaluation is carried out using metrics such as accuracy and classification reports. For instance, the KNN model's performance is assessed to identify its effectiveness in classifying facial expressions.

Finally, the best-performing model is used to predict emotions on the test dataset. Visualization of the results helps demonstrate how well each model recognizes facial expressions. This research aims to offer insights into the comparative effectiveness of traditional Machine Learning approaches versus Deep Learning models in FER tasks. The framework for face emotion recognition is shown in Figure 2.

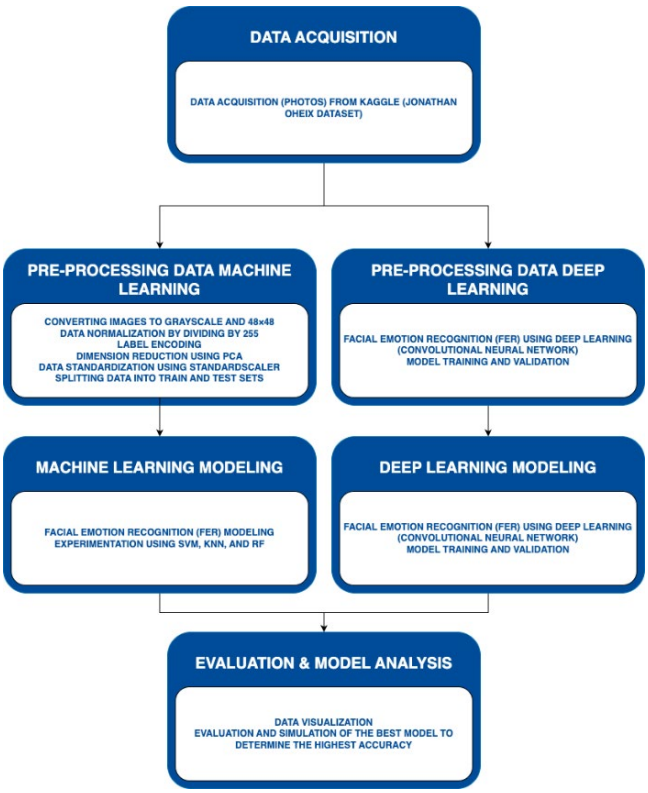

Figure 2. Framework for Face Emotion Recognition

A. Dataset

This study uses the Face Emotion Recognition Dataset provided by Jonathan Oheix on the Kaggle platform (https://www.kaggle.com/datasets/jonathanoheix/face-expression-recognition-dataset). The dataset consists of human facial images labeled into seven emotion categories: Angry, Happy, Sad, Surprised, Neutral, Fear, and Disgust. Each category includes images with varying lighting conditions, poses, and individuals, which increases the complexity of the emotion recognition process.

B. Pre-Processing Data for Machine Learning

1. Data Loading

The dataset is organized into folders labeled according to emotion categories (e.g., angry, happy, sad), making it easier to manage and process. The data is loaded using Python libraries such as os and Pillow, which facilitate efficient image file handling. To optimize computation and focus on essential facial features, all images are converted to grayscale using the .convert('L') method. This process removes unnecessary color information while preserving important texture details crucial for emotion recognition.

To ensure uniformity, each image is resized to 48×48 pixels, standardizing the input dimensions for the deep learning model. This step enhances consistency and simplifies the preprocessing pipeline, making the dataset more suitable for training. In addition, automatic labeling is performed by extracting class names from the folder names, allowing for seamless integration into the machine learning workflow while ensuring proper image categorization.

2. Cleaning and Data Transformation

After the images are loaded, the data needs to be processed to be suitable for training the machine learning model. This process includes pixel normalization, label encoding, dimensionality reduction using PCA (Principal Component Analysis), and data standardization.

Data Preprocessing Steps:

- **Normalization:** X = X / 255.0 is used to scale the pixel values to the range [0,1], making it easier for the model to learn patterns.
- **Label Encoding:** LabelEncoder() converts categorical emotion labels (e.g., happy, angry) into numerical values so they can be processed by the model.
- **PCA (Principal Component Analysis):** PCA(n_components=100) is used to reduce the data's dimensions to 100 main features, extracting the most relevant features for analysis.
- **Standardization:** StandardScaler() ensures that all features have a mean of 0 and a standard deviation of 1, which is important for maintaining consistent feature scaling across the dataset.

3. Data Splitting for Machine Learning Training and Evaluation

At this stage, the processed dataset is divided into two main parts: the training data and the testing data. The split is

DOI: 10.34148/teknika.v14i2.1232

performed using train_test_split, which helps users randomly divide the dataset into training and testing sets. This function is very useful in machine learning applications as it ensures that training and evaluation are done on separate data, preventing overfitting and providing reliable performance metrics.

This function has several optional parameters, such as test_size to determine the proportion of the test data. For example, test_size=0.3 means that 30% of the data is used for testing and the remaining data is used for training. Another important parameter is random_state, which guarantees reproducibility by generating the same data split each time the function is run with a specific seed value. This consistency is crucial for debugging, model comparison, and experiment repetition [20].

- **Data Split:**
  The dataset is split into 80% training data and 20% testing data. This split is done randomly using the train_test_split function from the scikit-learn library, ensuring that there is no overlap between the training and testing data.
- **Stratification:**
  The dataset is stratified, meaning the proportion of each emotion category is kept consistent in both the training and testing data. This prevents class imbalance, which could reduce the model's accuracy and performance.
- **Data Preparation for the Model:**
  The training and testing data are formatted to be compatible with various types of models, including machine learning algorithms (SVM, KNN, RF). The divided dataset is then used in the subsequent model training and evaluation stages.

C. Machine Learning Modeling

At this stage, the machine learning model is trained using the pre-processed data. Three main algorithms are applied: Support Vector Machine (SVM), Random Forest (RF), and K-Nearest Neighbors (KNN). Each algorithm is optimized using GridSearchCV, a method for finding the best parameter combination through cross-validation. The user defines a dictionary containing the hyperparameter names as keys and the list of values to be tested. GridSearchCV will test all combinations of these parameters on different subsets of the training data, analyze the results, and optimize the model. This optimization aims to produce a model with the best performance in classifying facial emotions based on the dataset used [21].

D. Machine Learning Evaluation

In the final evaluation stage, visualization is used to assess facial expression classification performance. Five grayscale test images are randomly displayed with true and predicted labels for visual comparison. A confusion matrix summarizes correct and incorrect predictions per emotion, highlighting common misclassifications.

An ROC curve is also plotted for each emotion class using binarized labels, with AUC values indicating the model's ability to distinguish emotions. A baseline for random predictions is included. These visual tools provide insights into the model's accuracy, reliability, and areas for improvement.

E. Face Emotion Recognition Using Deep Learning

Data preprocessing for deep learning models like CNNs differs from traditional machine learning. CNN preprocessing typically involves rescaling pixel values to the [0,1] range and reshaping images to fit the network's input size. Data augmentation techniques such as rotation, flipping, and zooming are often applied to improve generalization and prevent overfitting.

CNNs are designed to process image data by using convolutional layers that extract spatial features through learnable filters. These filters capture patterns like edges and textures. Pooling layers reduce the spatial dimensions of feature maps, and fully connected layers generate the final classification output.

Thanks to their ability to learn complex visual features, CNNs are widely used in tasks like image classification, object detection, and segmentation. Their performance can be tuned through parameters such as kernel size, number of filters, and activation functions.

1. Pre-Processing Data for Deep Learning

a. Data Loading

The dataset is organized into folders by facial expression class and loaded with ImageFolder, which assigns labels based on folder names. This structure helps the model learn to differentiate emotions effectively.

Data is loaded with a batch size of 64 using a DataLoader, with shuffling enabled to improve generalization. Four parallel workers speed up loading, and memory pinning is used to enhance CPU-to-GPU data transfer. These optimizations collectively boost training efficiency and model performance on unseen data.

b. Data Transformation

Before being fed into the Convolutional Neural Network (CNN), the input images undergo a series of preprocessing steps to enhance feature extraction and ensure consistency across the entire dataset. The first step is conversion to grayscale, where images are transformed into a single-channel format, removing color information and allowing the model to focus on facial texture and structure. This simplifies computation while retaining crucial details needed for emotion recognition.

Next, all images are resized to 48×48 pixels to ensure uniform input dimensions throughout the dataset. Standardizing image size helps improve computational efficiency and ensures the model processes inputs consistently. These steps not only reduce unnecessary complexity but also help the CNN focus on meaningful patterns, thereby enhancing classification performance.

c. Data Augmentation

To enhance model generalization and robustness, the dataset undergoes multiple data augmentation steps. Images are converted to grayscale and resized to 48×48 pixels to

DOI: 10.34148/teknika.v14i2.1232

ensure consistent input dimensions. Geometric augmentations include random affine transformations with rotations of ±10 degrees, translations up to 10%, and scaling between 0.9 and 1.1 to simulate natural variations in facial poses. Random horizontal flipping with a 50% chance increases diversity in orientation and symmetry, while brightness and contrast adjustments within a 0.8 to 1.2 range help the model adapt to different lighting conditions and reduce sensitivity to illumination changes.

Pixel values are normalized to the range [-1, 1] using a mean and standard deviation of 0.5 to stabilize training and prevent overfitting. Additionally, random erasing is applied with a 20% probability to simulate occlusions, making the model more resilient to partially blocked faces. For validation, a simpler preprocessing pipeline—grayscale conversion, resizing, and normalization—is used to provide a fair and unbiased evaluation of model performance without the influence of artificial augmentations.

2. Convolutional Neural Network Architecture

The proposed model consists of three main blocks based on Convolutional Neural Networks (CNN), combined with Residual Connections and Squeeze-and-Excitation (SE) blocks. The inclusion of SE blocks aims to enhance the model's ability to recalibrate the importance of features across channels (channel-wise).

a. Early Feature Extraction

In the initial stage, the model extracts basic features through two consecutive convolutional layers with 3×3 kernels and padding set to 1 to preserve spatial dimensions. Each convolutional layer is followed by Batch Normalization and the ReLU activation function. These initial layers serve to extract low-level features from the input images before passing them to the subsequent blocks.

b. Residual Convolutional Block with Squeeze-and-Excitation (SE)

The model consists of three residual convolutional blocks, each containing two convolutional layers enhanced with Batch Normalization, ReLU activation, Max Pooling, and Dropout for regularization. The number of filters increases progressively from 128 to 256 to 512, enabling the network to learn increasingly complex feature representations at deeper levels.

Each residual block is integrated with a Squeeze-and-Excitation (SE) block that performs channel-wise attention. The SE block includes:

- **Global Average Pooling:** Summarizes spatial information from each channel into a single value.
- **Two Fully Connected Layers:** With a reduction ratio of 1/16, using ReLU in the first layer and Sigmoid in the second to learn channel importance.
- **Feature Recalibration:** Applies learned weights to rescale channel responses, emphasizing important features and suppressing irrelevant ones.

This SE mechanism strengthens feature representation by allowing the model to focus on the most informative channels, improving its ability to recognize emotional expressions from facial images.

c. Shortcut Projections

To maintain dimensional consistency during residual learning, shortcut projections using 1×1 convolutions are applied in the first and second blocks. These shortcuts adjust the number of channels from 64 to 128 and from 128 to 256, respectively. The addition of these shortcuts also facilitates smoother gradient flow, thereby enhancing training stability and efficiency.

d. Global Average Pooling and Fully Connected Layers

After all features have been extracted, the features from the last block are processed using Global Average Pooling to reduce the spatial dimensions into a feature vector. This vector is then passed through several fully connected layers as follows:

- First layer: from 512 to 256 neurons with ReLU activation and dropout (p=0.5).
- Second layer: from 256 to 128 neurons with ReLU activation and dropout (p=0.5).
- Final layer: produces 7 outputs, representing the 7 emotion classes.

3. Training and Evaluation CNN Model

After the Convolutional Neural Network (CNN) architecture is defined, the next step is the training and evaluation process. This stage aims to optimize the model's weights using the backpropagation algorithm and evaluate its performance based on validation accuracy.

a. Initialization of the Model, Loss Function, and Optimizer

Before training, the CNN model is initialized and moved to the available device (CPU or GPU) to optimize computation, with GPUs used when available for faster training. The loss function is Cross-Entropy Loss, suitable for multi-class classification.

Label smoothing with a factor of 0.1 is applied to improve generalization by distributing some target probability across all classes, reducing overconfidence and overfitting. The AdamW optimizer is chosen for its adaptive learning rate and weight decay regularization, set at 0.05 to further prevent overfitting.

Training stability is enhanced using the OneCycleLR scheduler over 100 epochs, which gradually increases the learning rate to a peak of 0.001 before decreasing it via cosine annealing. Parameters such as a div_factor of 25.0 and final_div_factor of 1000.0 ensure smooth convergence and optimal performance.

b. Scaler and Loss Tracking

To improve training efficiency and stability, mixed precision training is used via GradScaler from torch.amp. This technique dynamically scales gradients to prevent numerical underflow, enabling faster computation and lower memory consumption on CUDA-supported GPUs.

DOI: 10.34148/teknika.v14i2.1232

In addition, early stopping is implemented to prevent overfitting and ensure optimal model performance. Validation accuracy is continuously monitored, and if no improvement is observed for 15 consecutive epochs, training is halted. This approach balances training duration and the model's generalization ability. To track training progress, both training loss and validation loss are recorded throughout the process, with the best model state saved for final evaluation.

4. Model Evaluation

After training, the model is evaluated on the validation dataset to measure performance and generalization. The model is set to evaluation mode (model.eval()), disabling dropout and stabilizing batch normalization for consistent predictions. Gradient calculations are turned off with torch.no_grad() to save memory and speed up inference, while mixed precision inference using torch.amp.autocast("cuda") further enhances efficiency.

Validation accuracy is computed by comparing predicted labels—obtained via argmax on output probabilities—with ground truth labels. Overall accuracy reflects the proportion of correctly classified samples, while per-class accuracy highlights performance across different emotion categories, helping to detect class imbalances or common misclassifications.

A high validation accuracy indicates strong generalization, but a large gap between training and validation results may suggest overfitting, necessitating adjustments like hyperparameter tuning, data augmentation, or regularization.

5. Model Training Process

The training process ran for 100 epochs, with the model set to training mode (model.train()) to enable dropout and update batch normalization statistics for better generalization. Each batch underwent a forward pass to generate predictions, and Cross-Entropy loss was calculated against true labels. Mixed precision training (torch.amp.autocast("cuda")) improved computational efficiency while maintaining stability.

During the backward pass, gradients were computed and updated using the AdamW optimizer, with gradient clipping applied to prevent exploding gradients. Gradient scaling (torch.amp.GradScaler) enhanced convergence in mixed precision. The OneCycleLR scheduler dynamically adjusted the learning rate after every batch, promoting balanced training and avoiding premature convergence.

Accuracy was tracked continuously, with per-class accuracy monitored to detect class imbalance or misclassification. Early stopping with a patience of 15 epochs prevented unnecessary training when validation accuracy plateaued. The best model checkpoint was saved upon improved validation performance.

To manage GPU memory, cache clearing occurred every five epochs. After training, loss curves were plotted to visualize learning progress, and the model was restored to its best state, ready for deployment or further fine-tuning.

6. Final Training and Evaluation Results

Model training starts with the train_part function, which handles the training loop, including mixed precision, gradient clipping, learning rate adjustment, and early stopping to promote robust learning. After training, the model is evaluated on the validation set using the evaluate_model function.

During evaluation, the model switches to evaluation mode (model.eval()), disabling dropout and stabilizing batch normalization. Inference runs under torch.no_grad() to save memory and increase efficiency by skipping gradient calculations.

Validation accuracy is calculated as the percentage of correct predictions overall, while per-class accuracy provides insights into performance across different facial expressions, highlighting any class-specific strengths or weaknesses.

IV. RESULTS AND DISCUSSION

Experiments were conducted on a system with an Intel i7-12700F CPU, 32 GB RAM, and an RTX 3070 GPU, using PyTorch and scikit-learn libraries. This study compares the performance of machine learning models—K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Random Forest (RF)—and a deep learning model, Convolutional Neural Network (CNN), for facial emotion recognition. The goal is to identify the most effective model for early detection of psychological conditions like anxiety, depression, and stress by evaluating accuracy, precision, recall, and F1-score. Unlike traditional models relying on manual feature extraction, CNNs automatically learn complex patterns, making them strong candidates for this task.

With rising mental health disorders, early detection is vital. Integrating emotion recognition into healthcare through digital apps, wearables, and telemedicine can improve real-time monitoring and intervention. This study offers insights into AI-driven mental health tools to aid psychological assessment and treatment.

A. Experimental Results

Best KNN: {'metric': 'euclidean', 'n_neighbors': 1, 'p': 1, 'weights': 'uniform'}. Accuracy: 0.40 and Classification Report as shown in Table 1 and Figure 3-4:

Table 1. KNN Experiment Results

	Precision	Recall	F1-score	Support
Angry	0.38	0.29	0.33	970
Disgust	0.28	0.39	0.33	107
Fear	0.36	0.34	0.35	1,015
Happy	0.52	0.43	0.47	1,824
Neutral	0.30	0.47	0.37	1,260
Sad	0.34	0.29	0.31	1,226
Surprise	0.60	0.60	0.60	776
Accuracy			0.40	7,178
Macro Avg	0.40	0.40	0.39	7,178
Weighted Avg	0.41	0.40	0.40	7,178

DOI: 10.34148/teknika.v14i2.1232

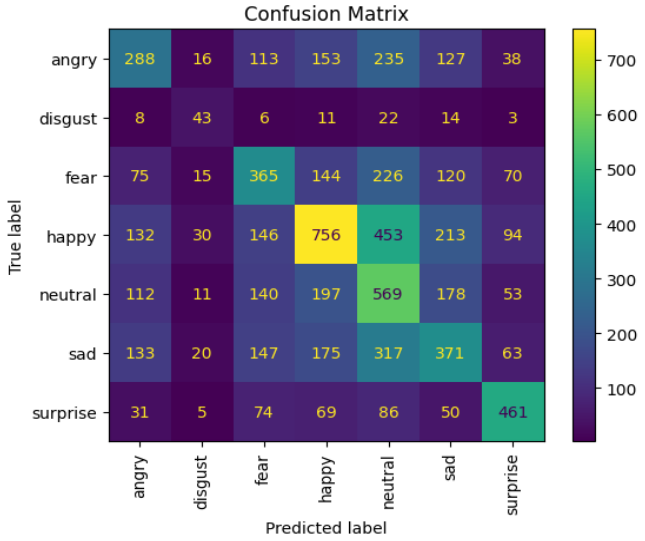

Figure 3. Confusion Matrix KNN

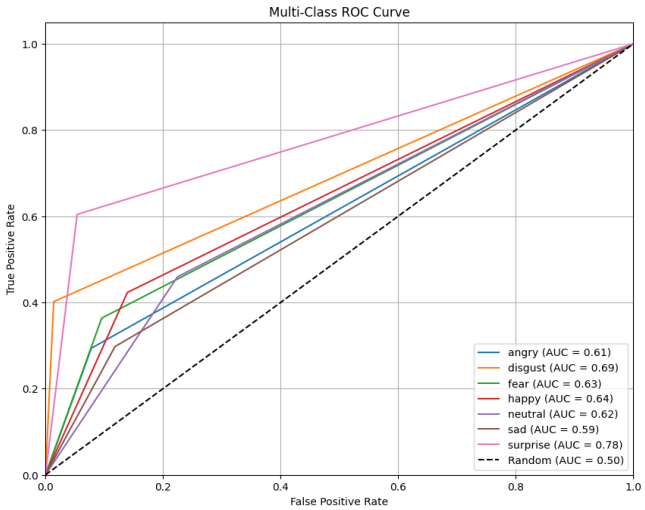

Figure 4. ROC Curve KNN

The K-Nearest Neighbors (KNN) algorithm was evaluated with various hyperparameter configurations. The best-performing KNN configuration utilized the Euclidean distance metric with $k = 1$, uniform weighting, and a Minkowski distance parameter $p = 1$. Despite the simplicity of this setup, the model achieved an accuracy of only 40%. This relatively low performance suggests that while KNN can offer baseline results, it may not be well-suited for capturing the complex patterns inherent in facial emotion data compared to more sophisticated models.

Below, the classification report provides a more detailed view of the KNN model's performance across individual emotion categories. Overall, the model's performance was inconsistent across classes, reaffirming the limitations of KNN in complex multi-class emotion classification tasks.

Best SVM: {'C': 10, 'degree': 3, 'gamma': 'scale', 'kernel': 'poly'}. Accuracy: 0.48 and Classification Report as shown in Table 2 and Figure 5-6:

Table 2. SVM Experiment Results

	Precision	Recall	F1-Score	Support
Angry	0.46	0.27	0.34	970
Disgust	0.67	0.34	0.45	107
Fear	0.47	0.29	0.36	1,015
Happy	0.49	0.75	0.60	1,824
Neutral	0.42	0.46	0.44	1,260
Sad	0.41	0.36	0.39	1,226
Surprise	0.70	0.61	0.65	776
Accuracy			0.48	7,178
Macro Avg	0.52	0.44	0.46	7,178
Weighted Avg	0.48	0.48	0.47	7,178

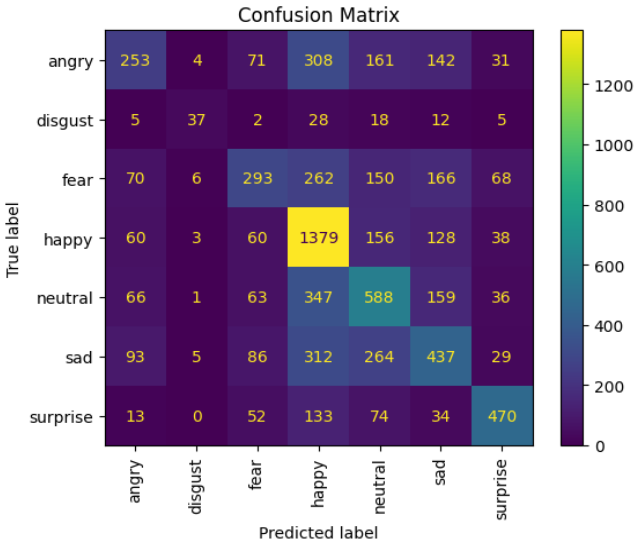

Figure 5. Confusion Matrix SVM

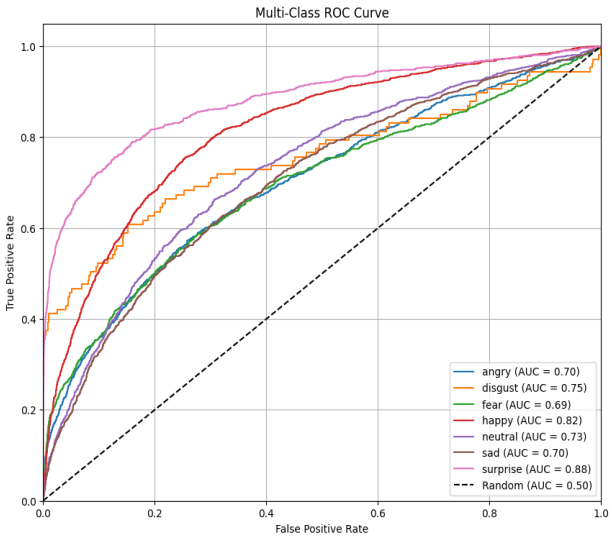

Figure 6. ROC Curve SVM

The Support Vector Machine (SVM) model was optimized through hyperparameter tuning, resulting in the best configuration using a polynomial kernel with degree 3, a

DOI: 10.34148/teknika.v14i2.1232

regularization parameter $C = 10$, and gamma set to 'scale'. This configuration achieved an overall accuracy of 48%, marking an improvement over the KNN model. The increased accuracy suggests that SVM, particularly with a nonlinear kernel, is better suited for handling the complex decision boundaries inherent in facial emotion recognition tasks.

The classification report for the SVM model is presented below and provides further insight into its class-wise performance. While a general improvement in F1-scores was observed across most emotion categories compared to KNN, the results also revealed varying levels of classification success.

Best RF: {'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 150}. Accuracy: 0.44 and Classification Report as shown in Table 3 and Figure 7-8:

Table 3. RF Experiment Results

	Precision	Recall	F1-score	Support
Angry	0.57	0.15	0.24	970
Disgust	1.00	0.21	0.34	107
Fear	0.55	0.23	0.33	1,015
Happy	0.39	0.82	0.53	1,824
Neutral	0.42	0.34	0.38	1,260
Sad	0.39	0.34	0.36	1,226
Surprise	0.69	0.53	0.60	776
Accuracy			0.44	7,178
Macro Avg	0.57	0.37	0.40	7,178
Weighted Avg	0.49	0.44	0.41	7,178

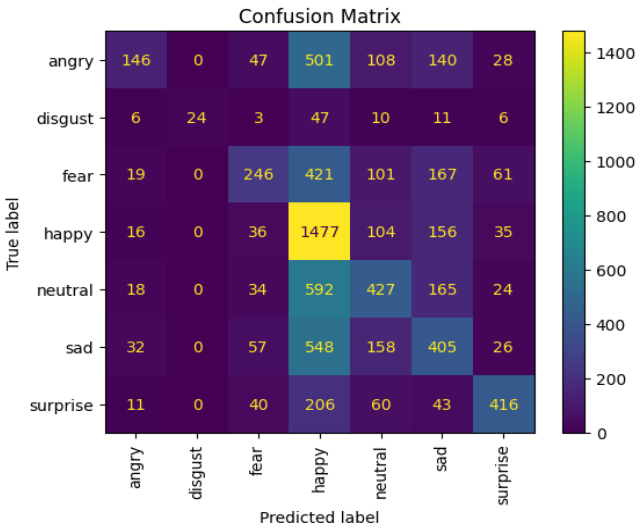

Figure 7. Confusion Matrix RF

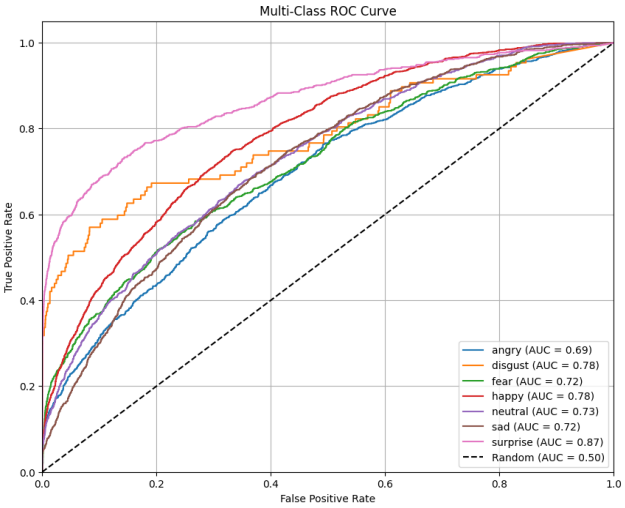

Figure 8. ROC Curve RF

The Random Forest classifier was evaluated using a grid search over several hyperparameters, and the best-performing configuration was identified with 150 estimators, no maximum depth constraint, a minimum of 1 sample per leaf, and a minimum of 5 samples required to split an internal node. This setup achieved an overall accuracy of 44%, placing its performance between that of the KNN and SVM models. The relatively modest accuracy reflects the model's ability to capture some discriminative patterns within the data, though its performance was still limited by the inherent complexity of the facial emotion classification task. Deep learning results is shown in Figure 9 and Table 4.

Figure 9. Deep Learning Results

Table 4. Deep Learning Results

Epochs	Learning Rate	Train Loss	Train Accuracy	Validation Loss	Validation Accuracy
1	0.000043	1.8579	23.39%	1.8162	28.04%
2	0.000050	1.8050	27.40%	1.7495	31.73%

3	0.000063	1.7356	32.81%	1.6528	39.99%
4	0.000081	1.6616	37.27%	1.5630	43.11%
5	0.000104	1.6167	39.93%	1.5416	44.69%
…..	…..	…..	…..	…..	…..
90	0.000050	0.8763	81.11%	1.0814	72.73%
91	0.000040	0.8699	81.30%	1.0820	72.36%
92	0.000032	0.8695	81.24%	1.0780	72.62%
93	0.000025	0.8663	81.61%	1.0771	72.36%
94	0.000018	0.8592	82.01%	1.0792	72.90%
95	0.000013	0.8620	81.94%	1.0813	72.63%
96	0.000008	0.8613	81.93%	1.0803	72.53%
97	0.000005	0.8572	81.93%	1.0799	72.52%
98	0.000002	0.8614	81.89%	1.0794	72.62%
99	0.000001	0.8555	82.11%	1.0785	72.64%
100	0.000001	0.8529	82.30%	1.0816	72.80%

The CNN model trained using CrossEntropyLoss with label smoothing (0.1), AdamW optimizer (lr=0.01, weight_decay=0.05), and OneCycleLR scheduler (max_lr=0.001, epochs=100) achieved the highest validation accuracy of 72.90% in the Facial Expression Recognition (FER) task. Mixed precision training with torch.amp.GradScaler("cuda") further improved computational efficiency.

This performance significantly outperforms traditional machine learning models such as SVM (48%) and Random Forest (44%), highlighting CNN's strength in automatically learning hierarchical features from images—essential for robust emotion classification. In contrast, traditional models rely on handcrafted features that are less effective in capturing spatial and contextual variations in facial expressions.

Key evaluation metrics such as precision, recall, and F1-score confirmed the CNN model's reliability, with the confusion matrix showing strong performance in distinguishing subtle emotions like sadness and surprise. These findings align with prior research and demonstrate the suitability of CNN-based approaches for real-world applications, including human-computer interaction and emotion-aware systems.

Further improvements can be pursued through hyperparameter tuning and dataset augmentation to enhance generalization and accuracy.

V. CONCLUSSION

The results clearly demonstrate that Convolutional Neural Networks (CNNs) significantly outperform traditional machine learning models in Facial Expression Recognition (FER). With a well-tuned setup—including label smoothing, AdamW optimizer, OneCycleLR scheduler, and mixed precision training—the CNN model achieved a strong validation accuracy of 72.90%, showcasing its superior ability to learn complex features from facial images.

In comparison, traditional models like SVM and Random Forest, which rely on handcrafted features, showed limited performance, with maximum accuracy reaching only 48%. This highlights the limitations of traditional approaches in handling the nuanced and variable nature of facial expressions.

Evaluation metrics such as precision, recall, and F1-score, along with a detailed confusion matrix, further validated the robustness of the CNN model in correctly identifying subtle emotions like sadness and surprise. These findings reinforce the strength of deep learning approaches in capturing intricate patterns essential for FER tasks.

In conclusion, CNN-based models are highly effective and reliable for real-world FER applications. Future improvements through hyperparameter optimization and data augmentation could push performance even further, making them ideal candidates for integration into emotion-aware technologies and systems.

REFERENCES

[1] Y. Miao, H. Dong, J. Aljaam, and A. El Saddik, "A Deep Learning System for Recognizing Facial Expression in Real-Time," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 15, pp. 1–20, May 2019, doi: 10.1145/3311747.

[2] N. Andalibi and J. Buss, "The Human in Emotion Recognition on Social Media: Attitudes, Outcomes, Risks," in *Conference on Human Factors in Computing Systems - Proceedings*, Association for Computing Machinery, Apr. 2020. doi: 10.1145/3313831.3376680.

[3] B. Deepa and K. Ramesh, "Epileptic seizure detection using deep learning through min max scaler normalization," *Int J Health Sci (Qassim)*, pp. 10981–10996, May 2022, doi: 10.53730/ijhs.v6ns1.7801.

[4] A. A. Soebroto, "Buku Ajar AI, Machine Learning & Deep Learning," 2019. [Online]. Available: https://www.researchgate.net/publication/348003841

[5] D. Sri Rahayu, J. Afifah, and S. Intan, "Classification of Diabetes Mellitus Using C4.5 Algorithm, Support Vector Machine (SVM) and Linear Regression Klasifikasi Penyakit Diabetes Melitus Menggunakan Algoritma C4.5, Support Vector Machine (SVM) dan Regresi Linear," *SENTIMAS: Seminar Nasional Penelitian dan Pengabdian Masyarakat*, Aug. 2023. [Online]. Available: https://journal.irpi.or.id/index.php/sentimas

[6] H. Jurnal, R. Alfiatul Karima, and Z. Fatah, "Implementasi Metode K-Nearest Neighbor Untuk Klasifikasi Penyakit Paru-Paru Pada Anak," *Jurnal Ilmiah Multidisiplin Ilmu*, Dec. 2024, doi: 10.69714/yx4smf68.

[7] D. Syaripudin, A. Walad, and F. Kesehatan dan Teknik, "Teknik Resampling Untuk Meningkatkan Nilai Akurasi Algoritma Random Forest Pada Data Prediksi Kecacatan Perangkat Lunak Resampling Technique To Increase The Accuracy Value Of Random Forest Algorithm On Software Defect Prediction Data," *JICN: Jurnal Intelek dan Cendikiawan Nusantara*, vol. 1, no.

DOI: 10.34148/teknika.v14i2.1232

3, 2024, [Online]. Available: https://jicnusantara.com/index.php/jicn
[8] S. Maesaroh *et al.*, *Bahasa Pemrograman Python*. 2024.
[9] D. V. Fakhriyani, *Kesehatan Mental*.
[10] Badan Penelitian dan Pengembangan Kesehatan, "Laporan Nasional Riskesdas 2018," Jakarta, Jun. 2020. Accessed: Apr. 02, 2025. [Online]. Available: https://repository.badankebijakan.kemkes.go.id/id/eprint/3514/
[11] Khamelia and Karina Terry, "Masalah Psikologis 2 Tahun Pandemi COVID-19 di Indonesia," Perhimpunan Dokter Spesialis Kedokteran Jiwa Indonesia. Accessed: Apr. 02, 2025. [Online]. Available: https://www.pdskji.org/home
[12] Y. Khaireddin and Z. Chen, "Facial Emotion Recognition: State of the Art Performance on FER2013," May 2021.
[13] G. Simcock *et al.*, "Associations between facial emotion recognition and mental health in early adolescence," *Int J Environ Res Public Health*, vol. 17, no. 1, Jan. 2020, doi: 10.3390/ijerph17010330.
[14] M. A. Hasnul, N. A. A. Aziz, S. Alelyani, M. Mohana, and A. A. Aziz, "Electrocardiogram-based emotion recognition systems and their applications in healthcare—a review," Aug. 01, 2021, *MDPI AG*. doi: 10.3390/s21155015.
[15] Python, "Pickle - Python Object Serialization," Python. Accessed: Jan. 05, 2025. [Online]. Available: https://docs.python.org/3/library/pickle.html.
[16] Gde Agung Brahmana Suryanegara, Adiwijaya, and Mahendra Dwifebri Purbolaksono, "Peningkatan Hasil Klasifikasi pada Algoritma Random Forest untuk Deteksi Pasien Penderita Diabetes Menggunakan Metode Normalisasi," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 5, no. 1, pp. 114–122, Feb. 2021, doi: 10.29207/resti.v5i1.2880.
[17] Scikit-Learn, "LabelEncoder," Scikit-Learn. Accessed: Dec. 31, 2024. [Online]. Available: https://scikit-learn.org/1.5/modules/generated/sklearn.preprocessing.LabelEncoder.html.
[18] Scikit-Learn, "PCA," Scikit-Learn. Accessed: Dec. 31, 2024. [Online]. Available: https://scikit-learn.org/1.5/modules/generated/sklearn.decomposition.PCA.html.
[19] F. Aldi, F. Hadi, N. A. Rahmi, and S. Defit, "Standardscaler's Potential in Enhancing Breast Cancer Accuracy Using Machine Learning," Aug. 2023.
[20] Scikit-Learn, "Test Train Split," Scikit-Learn. Accessed: Dec. 31, 2024. [Online]. Available: https://scikit-learn.org/1.5/modules/generated/sklearn.model_selection.train_test_split.html#sklearn.model_selection.train_test_split.
[21] Scikit-Learn, "Grid Search," Scikit-Learn. Accessed: Dec. 31, 2024. [Online]. Available: https://scikit-learn.org/1.5/modules/generated/sklearn.model_selection.GridSearchCV.html.

DOI: 10.34148/teknika.v14i2.1232