# CNN ve Performans Metrikleri Üzerine Araştırma Raporu

Hazırlayan: Marya Hijazi

Tarih: 18.10.2025\_\_\_\_\_\_\_\_

**1.1.Introduction**

In the midst of the rapid technological revolution, artificial intelligence has become a fundamental component in many modern innovations that touch our daily lives, such as intelligent image classification, speech recognition, and medical disease diagnosis. At the core of this progress lies deep learning, which has been considered in recent years as the gold standard in the machine learning community, achieving remarkable results in numerous complex cognitive tasks that match or even surpass human performance.

Convolutional Neural Networks (CNNs) are among the most prominent and influential deep learning techniques in the field of computer vision, which focuses on developing technologies that enable computers to see and recognize objects and the surrounding environment by understanding the content of digital images and videos. When we first hear the term Convolutional Neural Networks, a strange mixture of biology, mathematics, and computer science comes to mind. However, their design resembles neural connections in the human brain and is significantly influenced by the organization and function of the visual cortex.

## 1.2.Current Challenges and Innovative Solutions

Despite the significant successes these networks have achieved in advanced applications, optimizing their hyperparameters remains a major challenge, as achieving optimal tuning requires multiple manual experiments that consume considerable time and resources. Additionally, researchers face traditional problems such as overfitting and data scarcity, which limit these models' ability to generalize and perform well in unfamiliar cases.

This is where evolutionary deep learning using genetic algorithms emerges as an innovative solution. These algorithms rely on concepts inspired by nature, such as natural selection and mutation, to extensively explore hyperparameters and automatically design optimal network architectures. Neuroevolution approaches have proven effective in improving neural network structures and training them without intensive human intervention, thereby reducing human effort and experimental time.

## 1.3.Wide-Ranging Applications and Future Prospects

Convolutional Neural Networks are currently used in extensive applications including image classification and segmentation, object detection, video processing, natural language processing, and speech recognition. Deep learning has outperformed traditional machine learning techniques in numerous fields, such as cybersecurity, bioinformatics, robotics and control, and medical information processing. Thanks to these technologies' ability to process massive amounts of data and self-learn without repetitive programming, they open new horizons in critical fields such as medical image analysis, autonomous vehicles, and intelligent robotics applications.

## **Convolutional Neural Networks (CNNs)**

Recently, the integration of deep learning across various domains has demonstrated significant potential. Consequently, researchers have begun utilizing deep learning methods, originally developed for computer vision and natural language processing, to identify important features from EEG signals, even when dealing with complex situations. These complex situations include EEG recordings affected by noise and non-stationary, overlapping signal components, which often complicate the classification process.

Deep learning techniques, such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), fusion networks, and fuzzy learning methods, have been widely applied in EEG classification, contributing to advances in Motor Imagery (MI) tasks for Brain-Computer Interfaces and clinical applications. The use of CNNs and RNNs for learning feature representations in images and time series equally applies to EEG signal sequences. This is achieved by extracting features from EEG signals and feeding them into a classifier. The classification accuracy of a particular pattern can affect the effectiveness of the final Brain-Computer Interface system.

**2.1.Convolutional Neural Networks**

Convolutional Neural Networks are deep learning models designed to process data with grid-like structure, such as images, and form the foundation for most modern computer vision applications for detecting features within visual data. These networks perceive images as collections of numbers, commonly known as matrices, where each number represents light intensity at a specific point called a pixel.

## **2.2.Key Components of Convolutional Neural Networks**

## **2.2.1.Convolutional Layers** These layers apply convolution operations to input images using filters or kernels to detect features such as edges, textures, and more complex patterns, helping to preserve spatial relationships between pixels. The convolutional layer is defined by three fundamental parameters: kernel size (the size of the sliding kernel filter), stride length (the number of kernels that slide before forming product points and generating output pixels), and padding (the size of the zero frame prepared around the input feature map).

The convolution mechanism is based on applying a filter of a specific size (such as 3×3 or 5×5) to the input image, where this filter analyzes small portions of the image at a time, and the filter is moved across the image using a specified stride. At each step, the convolutional output is calculated through dot product between pixel values in the image portion and filter values, and these values are then aggregated into a feature map.

An important characteristic of convolutional layers is that they automatically learn the most important patterns in the data, where the first layers become responsible for capturing basic features such as edges and boundaries, while deeper layers specialize in analyzing more complex patterns such as shapes and complete objects. This allows the network to generalize and handle different forms of the same category without the need for manual feature design, making CNNs powerful in computer vision tasks.

**2.2.2.Pooling Layers**

These layers reduce the spatial dimensions of input data, thereby reducing computational complexity and the number of parameters in the network. The pooling layer combines two consecutive convolutional layers and reduces the number of parameters and computational loads by creating downsampled representations. Max pooling is a common pooling operation, where the maximum value is selected from a group of neighboring pixels. The function in the pooling layer can produce a maximum or average value, and maximization composition is often used to obtain an optimal function. The pooling layer also helps reduce overfitting or computational weights.

**2.2.3.1.Fully Connected Layer**

The third layer is the fully connected layer, commonly called the convolutional output layer. This layer resembles a feedforward neural network and is typically located in the lower layer of the network. The layer receives inputs from the final pooling or convolutional output layer and is flattened before being sent to the subsequent layer. The uniform distribution of output means unwrapping all result values obtained after the last pooling layer or convolutional layer into a vector (three-dimensional matrix). This method is a simple technique for studying high-level nonlinear combinations of features represented by the convolutional output layer. Each neuron in this layer connects to every neuron in the next layer and is responsible for making predictions based on the high-level features learned by previous layers.

**2.2.3.2.Customizing the Fully Connected Layer**

The fully connected layer plays a crucial role in transforming the information extracted from convolutional and pooling layers into final predictions. Customizing this layer requires making several important decisions that directly affect model performance.

**1.Number of Fully Connected Layers**

One or multiple fully connected layers can be used, depending on the complexity of the task. A single layer is used for simple tasks or when data is small in size, while multiple layers are used for complex tasks to learn more complex nonlinear representations. The model can contain only one output layer, or several hidden layers followed by the output layer.

**2.Number of Neurons in Each Layer**

The number of neurons significantly affects the model's learning capacity. A large number of neurons increases the model's expressive power but may lead to overfitting, while a small number reduces model complexity but may lead to poor performance (underfitting). Typically, the first layer contains 128, 256, 512, or 1024 neurons, with numbers gradually decreasing in intermediate layers (such as: 512, then 256, then 128), while the output layer must equal the number of classes to be classified.

**3.Choosing the Appropriate Activation Function**

The activation function differs depending on the layer's position in the network. For hidden (intermediate) layers, the ReLU function is the most common choice due to its speed and effectiveness, while Leaky ReLU can be used when facing the problem of dying neurons. Tanh or Sigmoid functions are rarely used in modern hidden layers. For the output layer, the Softmax function is used for multi-class classification, the Sigmoid function for binary classification or multi-label classification, while no activation function (or a linear function) is used for regression tasks.

**4.Adding Regularization Layers**

To prevent overfitting, several regularization mechanisms can be added. The Dropout technique randomly deactivates a proportion of neurons during training, with common rates ranging between 0.2 and 0.5. L1/L2 regularization can also be used, which adds a penalty on large weights, or Batch Normalization, which normalizes the distribution of inputs for each layer, helping to improve training stability and convergence speed.

**5.Weight Initialization**

The weight initialization method significantly affects the model's convergence speed. Xavier (or Glorot) initialization is the default for most cases, while He initialization is specifically recommended when using the ReLU function. Normal random initialization can also be used in some special cases.

**6.Customizing Learning Rate and Optimizer**

Choosing the optimizer and learning rate are critical decisions in model training. The Adam optimizer is the most common due to its balanced performance, while SGD with momentum can be used in cases requiring finer control. Common learning rates range between 0.001 and 0.01 and can be adjusted dynamically during training.

## **2.2.4.Nonlinearity Layer (Activation Functions)**

The activation function plays a fundamental role in CNN layers, as it adds nonlinearity to the model by enabling it to learn more complex relationships in the data. The filtered output provides another mathematical function called activation, and the main goal of the activation function is to determine the final output of the neural network, such as "yes" or "no". The activation function maps output values between -1 and 1, or between 0 and 1, and so on.

Activation functions can be distinguished into two main categories:

1. **Linear Activation Functions**: Input values are multiplied by a constant parameter, which is the weight of each neuron, and the operation produces an output proportional to the inputs. Linear functions can perform more than step functions because they give only one final answer of yes or no and not multiple options.
2. **Nonlinear Activation Functions**: Used in modern neural networks, they allow the model to design complex mappings between network inputs and outputs, which is necessary for complex learning and modeling systems.

## **2.3.The most common activation functions in CNNs and other neural networks 2.3.1.**Sigmoid Function****

## This activation function uses real numbers as inputs and limits the output between 0 and 1. The curve of the sigmoid function is S-shaped

## **2.3.2.**Tanh Function****

## The tanh function is similar to sigmoid since both use real numbers as their inputs. However, the tanh function limits its output between -1 and 1.

## **2.3.3.**ReLU Function****

## ReLU is the Rectified Linear Unit [27] and is the most common activation function used in feature extraction using CNNs. All inputs are converted into positive numbers. The computational load of ReLU is relatively lower than other functions.

## **2.3..4.**Leaky ReLU Function****

## If the ReLU function is responsible for downscaling negative inputs, the Leaky ReLU function ensures that inputs are never ignored. This function is used to solve a dying issue in ReLU.

## **2.3.5.**Noisy ReLU Function****

## This function is used to perform Gaussian distribution.

**2.3.6.**Parametric Linear Units****

Most of this function adopts the concept of Leaky ReLU. The difference between both functions is shown in the leak factor updated through the training mode.

## **3. Evaluating Convolutional Neural Network Model Performance**

Evaluating the performance of deep learning models, particularly Convolutional Neural Networks, is a critical step in understanding the model's effectiveness in performing classification tasks. The evaluation process relies on a set of quantitative metrics that provide comprehensive insights into the model's ability to distinguish between different classes and identify its strengths and weaknesses.

### 3.1 Confusion Matrix

The confusion matrix is a fundamental analytical tool used to evaluate the performance of classification models by comparing the predicted values from the model with the actual true values. It displays the number of cases that were correctly or incorrectly classified for each class. A confusion matrix is defined as a square matrix where the number of rows equals the number of columns, both equal to the number of classes in the classification problem. This matrix displays, for each actual class (represented in rows), the number of samples the model classified into each predicted class (represented in columns).

The element on the main diagonal of the matrix (CMᵢᵢ) represents the number of correctly classified cases for class i, which is known as True Positive for that class. The remaining cells outside the main diagonal represent errors, i.e., cases that were assigned to incorrect classes. In binary classification, the matrix consists of four main components: True Positive (TP), which is the number of samples that actually belonged to the positive class and were correctly classified by the model; True Negative (TN), which is the number of samples that did not belong to the positive class and were also correctly classified by the model; False Positive (FP), which is the number of samples the model classified as positive when they were actually negative; and False Negative (FN), which is the number of samples the model failed to identify despite actually belonging to the positive class.

The importance of the confusion matrix lies in providing a detailed and accurate picture of the error patterns made by the model, helping researchers and developers direct corrective actions in a targeted manner. For example, if the matrix shows that the model frequently confuses two specific classes, efforts can be focused on improving the model's ability to distinguish between these two classes specifically.

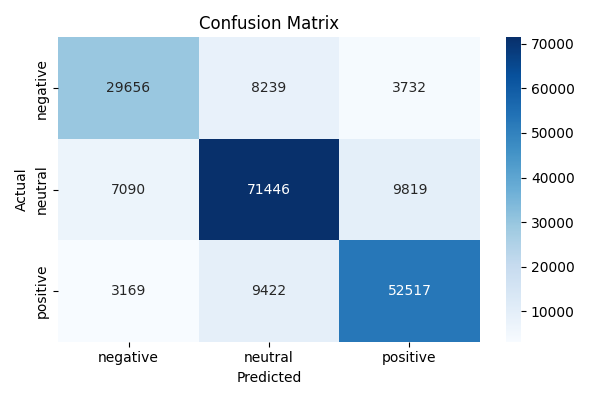


Figure:1.1Confusion matrix calculation

### **3.2 Precision (Kesinlik)**

Precision is defined as the ratio of the number of cases correctly classified as positive to the total number of cases the model predicted as positive. Mathematically, precision is calculated by the equation: Precision = TP / (TP + FP), where TP is the number of correct positive predictions for a specific class, and FP is the number of incorrect positive predictions that the model classified as a specific class when they were not.

The Precision value indicates the model's confidence and accuracy in its positive predictions. The importance of this metric increases significantly in cases where false positive errors are costly or sensitive. For example, in medical diagnostic applications, giving a false positive result to a healthy person may lead to unnecessary psychological distress or subjecting them to additional costly and unneeded medical procedures. Similarly, in systems for detecting emails requiring legal action or security alerts, a large number of false alerts costs the relevant authorities significant human and financial resources without real justification.

### **3.3 Recall**

Recall expresses the model's ability to detect all actual positive cases, i.e., the proportion of positive cases the model correctly identified among all true positive cases. Recall is calculated by the equation: Recall = TP / (TP + FN), where FN is the number of actual cases of the class that the model failed to detect.

The importance of Recall lies in situations where false negative errors (False Negatives) are serious or have severe consequences. For example, in detecting serious diseases, failing to detect an existing disease (FN) may lead to delayed treatment and deterioration of the patient's health condition. Similarly, in industrial environments, failure to recognize actual contamination or a safety system malfunction may lead to environmental disasters or serious accidents. In these cases, a model with high Recall value is preferred even if it issues some false alerts (False Positives), because the cost of missing a real case is much greater than the cost of dealing with additional false alarms.

In the context of practical applications such as sentiment analysis or Brain-Computer Interfaces, Recall becomes particularly important when the system is designed to detect harmful content or major problems requiring immediate intervention, such as monitoring critical complaints or safety warnings.

### **3.4 F1-Score**

The F1-Score is considered a harmonic balanced combination of Precision and Recall, used to achieve balance between them, especially in cases where there is significant imbalance in the number of samples between classes (class imbalance) or when both positive and negative errors equally affect system performance. The F1-Score is calculated by the equation: F1-Score = 2 × (Precision × Recall) / (Precision + Recall).

A high F1 value indicates that the model is balanced in its performance, meaning it achieves good accuracy in positive predictions (high Precision) while simultaneously detecting most true positive cases (high Recall). This metric is particularly useful when comparing different models, as it provides a single value summarizing overall performance rather than looking at Precision and Recall separately. In practical applications, the weighted average or macro average of the F1-Score across all classes is often used to obtain a comprehensive evaluation of model performance in multi-class classification tasks.

### **3.5 Accuracy**

Accuracy is mathematically defined by the equation: Accuracy = (TP + TN) / (TP + TN + FP + FN). Overall accuracy shows the proportion of correctly classified samples out of the total number of all samples in the test set.

Although overall accuracy is an intuitive and easy-to-understand metric, it should be used with caution, especially in cases of class imbalance. If class distribution is imbalanced, where one class is much more represented than other classes (such as having a neutral class representing 70% of the data), overall accuracy may be misleading. In such cases, a simple model that always prefers to predict the largest class may achieve high overall accuracy without actually being able to distinguish between less frequent classes. Therefore, it is always necessary to examine Precision, Recall, and F1 metrics for each class individually to obtain an accurate and comprehensive understanding of model performance, and to ensure that the model is not biased toward more represented classes at the expense of less represented ones.

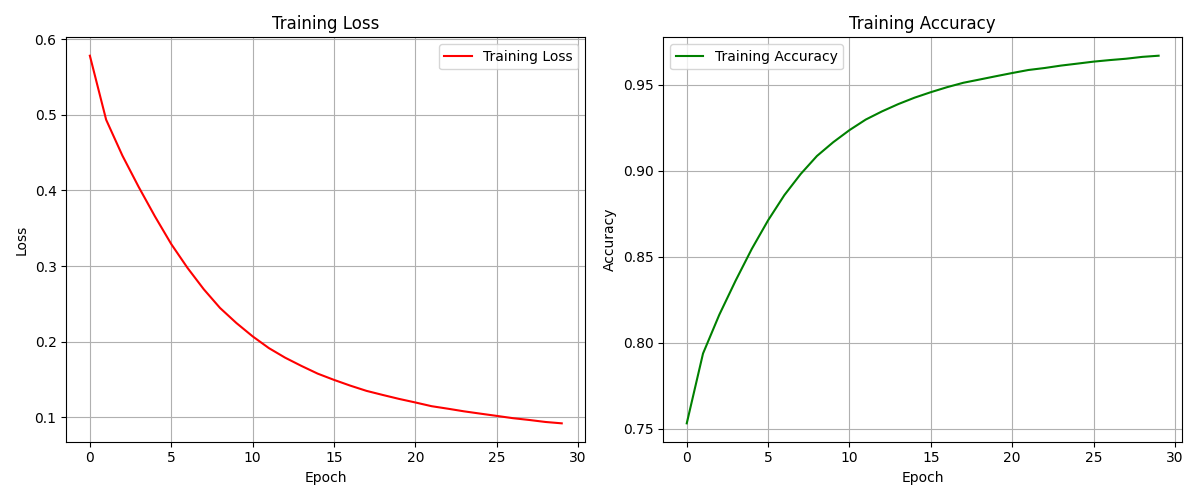


Figure:1.2 Training Accuracy calculation

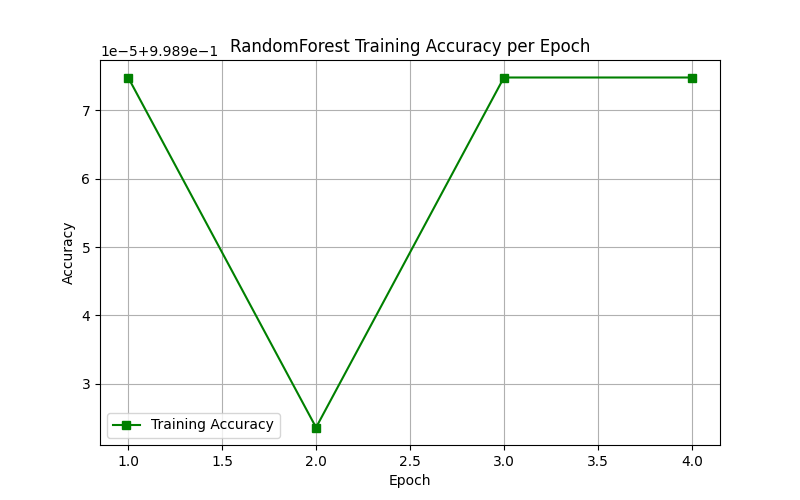


Figure:1.3 Accuracy per epoch

## 4.1. Applications of Convolutional Neural Networks

Convolutional Neural Networks have proven highly effective in a wide range of practical applications, making them the optimal choice for many computer vision and visual data analysis tasks. These applications are diverse, covering multiple fields ranging from everyday applications to advanced applications in medical, industrial, and security domains.

### **4.1.1.Image Classification**

Convolutional Neural Networks are the most advanced and effective models in the field of image classification. These networks can classify images into different categories with high accuracy, such as distinguishing between cats and dogs, classifying types of plants and animals, or recognizing different objects in images. CNNs excel in this field due to their ability to automatically learn distinctive features for each class without the need for manual feature engineering, making them capable of handling a wide range of challenges such as variations in lighting, camera angles, size, and background. This technology is used in numerous applications such as organizing photos on smartphones, image search on the internet, and medical imaging systems.

### **4.1.2.Object Detection**

The capabilities of Convolutional Neural Networks extend beyond mere image classification to include detecting and localizing different objects within a single image. These networks can identify the presence of multiple objects in an image such as people, cars, buildings, traffic signs, and others, while determining their precise location by drawing bounding boxes around each detected object. This process is known as localization and is fundamental in many applications such as autonomous vehicles that need to locate pedestrians, other vehicles, and obstacles, security and surveillance systems, and augmented reality applications.

### **4.1.3.Image Segmentation**

Convolutional Neural Networks are used for image segmentation, which is the process of identifying and labeling each pixel in an image according to the object or class it belongs to. Unlike object detection which identifies bounding boxes, image segmentation provides precise boundaries for each object at the pixel level. This technique is of great importance in medical applications, where it can be used to determine the boundaries of tumors or organs in X-ray images and MRI scans, helping doctors with diagnosis and surgical planning. It is also used in robotics to help robots accurately understand their surrounding environment, and in advanced image editing applications where different elements in an image can be separated with high precision.

### **4.1.4.Video Analysis**

The capabilities of Convolutional Neural Networks extend to video analysis, where they can track objects across consecutive frames (object tracking), detect different events and activities in videos, and recognize movements and gestures. These capabilities are particularly useful in video surveillance applications, where the system can automatically detect suspicious or unusual behaviors and alert authorities. They are also used in traffic monitoring to analyze congestion patterns, detect accidents, and count vehicles. In sports, they are used to analyze player performance and track their movements, and in entertainment for producing special effects and motion capture techniques.

## 4.2.Conclusion

Optimizing the parameters of Convolutional Neural Networks is one of the most prominent challenges in developing accurate and efficient artificial intelligence models, as these parameters directly affect model accuracy, training speed, and computational resource efficiency. Although traditional methods such as Grid Search and Bayesian Optimization provide solutions for optimizing these parameters, they suffer from high computational cost and difficulty dealing with high-dimensional search spaces.

In this article, we have reviewed the main components of Convolutional Neural Networks, starting from convolutional layers that extract basic features from data, through pooling layers that reduce spatial dimensions and enhance computational efficiency, to fully connected layers that transform extracted information into final predictions. We also addressed the importance of activation functions in adding nonlinearity to the model, enabling it to learn complex relationships in data.

We also explained how to customize fully connected layers by choosing the appropriate number of layers, determining the number of neurons, selecting suitable activation functions, and adding regularization mechanisms to prevent overfitting. These design decisions play a crucial role in achieving balance between the model's ability to learn and its ability to generalize to new data.

We also discussed the fundamental evaluation metrics used to assess the performance of classification models, including confusion matrix, precision, recall, F1-Score, and overall accuracy. Understanding these metrics and choosing the appropriate ones according to the nature of the application is essential for developing effective models that meet specific requirements.

The wide applications of Convolutional Neural Networks in the fields of image classification, object detection, image segmentation, and video analysis demonstrate the enormous practical value of this technology. With continued development in algorithms and computational capabilities, we expect Convolutional Neural Networks to continue playing a pivotal role in advancing the field of artificial intelligence and computer vision, with new and innovative applications emerging in various scientific, industrial, and medical fields.

## **5. Kaynakça**

[1] <https://www.geeksforgeeks.org/deep-learning/convolutional-neural-network-cnn-in-machine-learning/>

[2] <https://journalofbigdata.springeropen.com/articles/10.1186/s40537-021-00444-8>

[3]<https://www.researchgate.net/publication/367157330_Understanding_of_Convolutional_Neural_Network_CNN_A_Review>

[4] <https://www.sciencedirect.com/science/article/pii/S1746809425000527>

[5] <https://fihm.ai/tutorials/introduction-to-convolutional-neural-networks-cnn-part-1/>

[6] <https://wpu.edu.sy/wpus/%D8%AA%D8%AD%D8%B3%D9%8A%D9%86-%D9%85%D8%B9%D8%A7%D9%85%D9%84%D8%A7%D8%AA-%D8%B4%D8%A8%D9%83%D8%A7%D8%AA-cnn-%D8%A8%D8%A7%D8%B3%D8%AA%D8%AE%D8%AF%D8%A7%D9%85-%D8%A7%D9%84%D8%AA%D8%B9%D9%84%D9%8A%D9%85/>