Convolutional Neural Networks for Image Classification

POOJA BHAVANI AVUTHU

Computer Science Department, University of Central Missouri  
1101 Innovation Parkway, Lee’s Summit, MO,64086

*pxa69360@ucmo.edu*

*Abstract* - Convolutional Neural Networks (CNNs) have emerged as a powerful tool for image classification tasks in recent years. Their ability to automatically learn hierarchical features from raw pixel data has led to significant advancements in various domains such as computer vision, medical imaging, and autonomous driving. This paper provides an overview of CNN architectures, starting from the basic building blocks like convolutional layers, pooling layers, and fully connected layers, to more advanced techniques such as residual connections, batch normalization, and transfer learning. Additionally, we discuss common challenges in training CNNs, such as overfitting and vanishing gradients, and present state-of-the-art solutions to address these issues.

Moreover, we examine emerging trends and innovations in CNN research, such as attention mechanisms, capsule networks, and adversarial training, elucidating their potential to advance the state-of-the-art in image classification. Through a comprehensive review of CNN architectures, training methodologies, and recent advancements, this paper aims to provide researchers and practitioners with a deeper understanding of the principles and practices driving the success of CNNs in image classification.

Keywords – Convolutional Neural Networks, CNNs, image classification, batch normalization, pixel data (key words)

# Introduction

In recent years, the field of computer vision has witnessed remarkable advancements fueled by the advent of Convolutional Neural Networks (CNNs). Among the myriad of tasks that CNNs excel at, image classification stands as one of the most prominent and extensively studied domains. The ability of CNNs to automatically learn hierarchical representations from raw pixel data has revolutionized the process of classifying images into predefined categories, with applications spanning from medical diagnosis to autonomous driving.

The inception of CNNs can be traced back to seminal works which demonstrated the efficacy of deep convolutional architectures in image recognition tasks. Since then, the landscape of CNN research has evolved rapidly, witnessing the emergence of increasingly complex and efficient architectures tailored for diverse applications.

This paper aims to provide a comprehensive overview of CNNs for image classification, covering both foundational concepts and state-of-the-art advancements. We begin by elucidating the basic building blocks of CNNs, including convolutional layers, pooling layers, and fully connected layers, and discuss their role in extracting meaningful features from input images. Subsequently, we delve into a detailed examination of prominent CNN architectures, highlighting their design principles, architectural innovations, and performance benchmarks on benchmark datasets.

Furthermore, we explore essential techniques for training CNNs effectively, addressing challenges such as overfitting, vanishing gradients, and training data scarcity. We discuss regularization methods, normalization techniques, optimization algorithms, and data augmentation strategies, showcasing their contributions to improving model generalization and robustness.

Moreover, we review recent developments and trends in CNN research, such as attention mechanisms, capsule networks, and adversarial training, which promise to push the boundaries of image classification performance even further.

By synthesizing insights from seminal works and recent advancements, this paper aims to provide researchers and practitioners with a comprehensive understanding of CNNs for image classification, fostering further innovation and application of deep learning in computer vision tasks

# MOTIVATION

The motivation behind exploring Convolutional Neural Networks (CNNs) for image classification stems from the inherent complexity and richness of visual data and the need for automated and accurate methods to analyze and classify images across various domains. Here are several key motivations:

**Complexity of Visual Data:** Images contain a wealth of information that is not easily represented by traditional algorithms. From textures and shapes to spatial relationships, visual data presents unique challenges that require sophisticated techniques for effective analysis and interpretation.

**Explosion of Image Data:** With the proliferation of digital imaging devices and online platforms, the volume of image data generated daily has grown exponentially. Manual annotation and classification of such vast datasets are impractical, necessitating automated approaches that can scale efficiently.

**Diverse Applications:** Image classification has numerous applications across domains such as healthcare (e.g., medical imaging diagnosis), surveillance, autonomous vehicles, agriculture, retail, and more. Accurate and efficient image classification models have the potential to streamline processes, enhance decision-making, and improve outcomes in these domains.

**Previous Limitations:** Traditional computer vision techniques often relied on handcrafted features and shallow learning models, which struggled to capture the complexity of visual data. CNNs have demonstrated superior performance by automatically learning hierarchical features from raw pixel data, overcoming many limitations of traditional approaches.

**Potential for Innovation:** CNNs offer a fertile ground for innovation, with ongoing research focusing on improving model architectures, training techniques, and interpretability. Exploring CNNs for image classification not only addresses immediate practical needs but also drives advancements in deep learning methodologies and computer vision as a whole.

In summary, the motivation behind investigating CNNs for image classification lies in the need for robust, scalable, and automated solutions to analyze visual data across diverse applications, leveraging the power of deep learning to address the complexities inherent in images.

# CONTRIBUTIONS&OBJECTIVE

The main contributions and objectives of a paper on "Convolutional Neural Networks for Image Classification" typically include:

* Comprehensive Overview: Provide a comprehensive overview of Convolutional Neural Networks (CNNs) tailored specifically for image classification tasks. This includes discussing foundational concepts, common architectures, training methodologies, and evaluation metrics.
* Detailed Examination of Architectures: Explore prominent CNN architectures used for image classification, ranging from early models. Analyze the design principles, architectural innovations, and performance characteristics of these models.
* Training Techniques and Challenges: Discuss essential training techniques for CNNs, including regularization methods, normalization techniques, optimization algorithms, and data augmentation strategies. Address common challenges encountered during training, such as overfitting, vanishing gradients, and training data scarcity, and present state-of-the-art solutions to mitigate these challenges.
* Performance Evaluation: Provide a thorough performance evaluation of CNNs on benchmark image classification datasets, such as CIFAR-10, CIFAR-100, MNIST, and ImageNet. Report accuracy, precision, recall, and other relevant metrics to assess the effectiveness and robustness of different CNN architectures and training methodologies.
* Comparison with Baseline Methods: Compare the performance of CNNs with baseline methods, including traditional computer vision techniques and shallow learning models, to highlight the superiority of CNNs in terms of accuracy, efficiency, and scalability.
* Discussion of Emerging Trends: Review recent advancements and emerging trends in CNN research for image classification, such as attention mechanisms, capsule networks, and adversarial training. Discuss their potential impact on improving model performance and addressing current limitations.
* Practical Implications and Applications: Discuss practical implications and applications of CNNs for image classification across various domains, including healthcare, surveillance, autonomous vehicles, agriculture, retail, and more. Highlight real-world scenarios where CNNs can streamline processes, enhance decision-making, and improve outcomes.
* Future Directions: Identify key challenges and opportunities for future research in CNNs for image classification. Discuss potential avenues for innovation, including novel architectures, training techniques, and applications, to further advance the state-of-the-art in computer vision.

Overall, the main contributions and objectives of the paper aim to provide researchers and practitioners with a comprehensive understanding of CNNs for image classification, fostering further innovation and application of deep learning in this critical domain.

# RELATED WORK

The objective of this part is to build an algorithm that can identify photographs of cats and dogs. To create predictions, it examines input photographs of dogs and cats. The implemented model can be adjusted for mobile devices or websites.

We first initialize the CNN.

We are utilizing Adam Optimizer to compile the CNN.

The technique called Adaptive Moment Estimation (Adam) is used to determine the unique learning rates for every parameter. To calculate the loss function, we compare each of the predicted probabilities with the class output using Binary cross-entropy. The penalization score is then determined by subtracting the total distance from the expected value.

The process of applying various transformations to source photos to produce many altered copies of the same image is known as image augmentation. The photos differ from one another in certain ways due to the procedures of shifting, rotating, and flipping. Thus, to enhance our photographs, we are use the Keras ImageDataGenerator class.

In order to deliver the images to the network during training, we need a means to convert them into batches of data arrays in memory. For this, ImageDataGenerator is a convenient tool. Consequently, we import this class and execute the generator. With the help of the ImageDataGenerator class's flow\_from\_directory method, we are use Keras to retrieve photos from the disk.

By utilizing the predict\_image function, which requires us to supply the path of the new image as the image path and apply the predict technique, we can use our model to predict new images. The image will show a dog if the likelihood is more than 0.5, else a cat.

# PROPOSED SYSTEM

The proposed method is used to differ cats and dogs using Convolutional neural networks.

The Convolutional Neural Network is a particular type of neural network inspired by biological visual systems and have been proven to be very effective for image recognition. The word” convolution” comes from the convolution layer which consists of a dot product of a square grid of pixels (” filter”). The grid is translated across the image to form a feature map. The application of various filters (defined during training) allows for the detection of features such as edges and even complicated shapes.

**Convolution**

**A diagram of a convolution filter

Description automatically generated**

Convolution involves linearly multiplying weights with the input. This multiplication occurs between an array of input data and a 2D array of weights called a filter or kernel. The filter is consistently smaller than the input data, and the dot product takes place between the input and filter array.

**Activation**

A diagram of a picture

Description automatically generated with medium confidence

We add the activation function to assist the Artificial Neural Network (ANN) in learning complex patterns within the data. The primary purpose of the activation function is to introduce non-linearity into the neural network.

**Pooling**

**A diagram of a number

Description automatically generated**

The pooling operation provides spatial variance making the system capable of recognizing an object with some varied appearance. It involves adding a 2Dfilter over each channel of the feature map and thus summarize features lying in that region covered by the filter.

So, pooling basically helps reduce the number of parameters and computations present in the network. It progressively reduces the spatial size of the network and thus controls overfitting. There are two types of operations in this layer: Average pooling and Maximum pooling. Here, we are using max pooling which according to its name will only take out the maximum from a pool. With the help of filters sliding through the input and at each stride, the maximum parameter is taken out, and the rest are dropped.

The pooling layer does not modify the depth of the network unlike in the convolution layer.

**Fully Connected**

The fully connected layer receives the flattened output from the final pooling layer.

A diagram of a diagram of a layer of paper

Description automatically generated with medium confidence

The Full Connection process practically works as follows:

The neurons present in the fully connected layer detect a certain feature and preserves its value then communicates the value to both the dog and cat classes who then check out the feature and decide if the feature is relevant to them.

Our aim in the project is to implement image classification on a dataset which includes images of cats and dog and classify them by training the CNN model and testing the model.

The workflow we will be using for our project is as below:

\* Understanding and Pre-processing Data

\* Creating the Model

\* Training the Model

\* Testing the Model

\* Improvising and repeating the process to increase its accuracy

* Understanding and Pre-processing Data

We start with understanding the dataset used for training. And classify them as cat and dog so that we can segregate the data for model to learn. We have species information for dataset in this block such as the split ratio and image size. Next, we focused on Data preparation where we do the following processes:

* Read image from data les.
* Resize it so that features can be extracted later.
* Convert image from string type to float.
* Append processed image along with the label.
* Creating the Model

The model consists of five convolution blocks with a max pool layer in each of them. There is also a fully connected layer with 256 units. This is activated using ReLU activation function. ReLU stands for recti ed linear unit and is the most used activation function CNNs. This function is linear for all positive values, and zero for all negative values. Using ReLU reduces the time required to train the models it doesn’t have complex mathematics associated with it.

We then complied the model using binary cross entropy loss function along with ADAM optimizer. After creating the model, we trained it based on CNN and tested the accuracy and loss percentage.

**Improvising of Computational Neural Network**

1. **Batch Normalization**

Batch normalization is a technique for training very deep neural networks that standardizes the inputs to a layer for each mini batch. This has the e act of stabilizing the learning process and dramatically reducing the number of training epochs required to train deep networks. Batch Normalization helps us as it converges fast i.e. loss starts to lower faster This could help us minimize the number of epochs required, if we stopped training after there are no more improvements.

1. **Dropout**

Another technique to reduce obverting is to introduce dropout to the network. It is a form of regularization that forces the weights in the network to take only small values which makes the distribution of weight values more regular, and the network can reduce over on small training examples. We have applied 0.2 dropout to few layers and 0.3 to some to regularize the model. When you apply dropout to a layer it randomly drops out (set to zero) number of output units from the applied layer during the training process. Dropout takes a fractional number as its input value, in the form such as 0.1, 0.2, 0.4, etc. This means dropping out 10%, 20% or 40% of the output units randomly from the applied layer. When applying 0.1 dropout to a certain layer, it randomly kills 10 of the output units in each training epoch. We have applied 0.2 dropout to few layers and 0.3 to some to regularize the model.

1. **Decayed Learning Rate**

Learning rate controls the step we make along the gradient if the learning rate is high that means we are taking bigger steps, and hence decaying the learning rate for every epoch can help us increase the accuracy. When we decrease the learning rate, it allows our model to descend into areas of the loss landscape that are more optimal and would have otherwise been missed entirely. For GPU (NVIDIA GeForce- 940 MX), and 10,000 images to train, we chose the batch size of 8, and then reduced the learning rate to increase the accuracy.

* Training the Model

A table with numbers and letters

Description automatically generated

We ran the model multiple time to understand how the accuracy and loss rate is affected by the changes we make. As we increase the layers of Convolution, epochs and no. of images to train we can see the accuracy increasing. As we can see in the last rows for 10,000 and 15,000 images the model seems to be overfitting as the training accuracy is increasing linearly over time, whereas validation accuracy stalls around 81%-90% in the training process. Also, the difference in accuracy between training and validation accuracy is noticeably a sign of overfitting. When there are a small number of training examples, the model sometimes learns from noises or unwanted details from training examples to an extent that it negatively impacts the performance of the model on new examples. This phenomenon is known as overfitting. It means that the model will have a di cult time generalizing on a new dataset.

# DATA DESCRIPTION

When describing data for Convolutional Neural Networks (CNNs) for image classification, it's important to consider several key aspects:

* Image Dimensions: CNNs typically expect input images to have consistent dimensions. This means you need to decide on the width, height, and number of color channels for your images. Common choices include 224x224 or 256x256 pixels with 3 color channels (red, green, blue).
* Data Size: The size of your dataset matters. CNNs often require large amounts of data for effective training, especially when training from scratch. However, techniques like transfer learning allow you to leverage pre-trained models on smaller datasets.
* Data Augmentation: Data augmentation techniques can help increase the effective size of your dataset by applying random transformations to your images during training. Common augmentations include rotation, flipping, scaling, cropping, and adjusting brightness or contrast.
* Labeling: Each image in your dataset should have a corresponding label indicating its class. For classification tasks, these labels represent the categories you want your CNN to learn to classify images into.
* Class Balance: Ideally, your dataset should have a roughly equal number of examples for each class to prevent the model from being biased towards more prevalent classes. If class imbalances exist, techniques such as oversampling, undersampling, or class-weighted loss functions can be employed.
* Data Preprocessing: Before feeding images into a CNN, preprocessing steps are often applied to normalize the data and make training more efficient. Common preprocessing steps include resizing images to the network's input dimensions, normalizing pixel values to a certain range (e.g., [0, 1] or [-1, 1]), and possibly subtracting the mean pixel value of the dataset.
* Data Splitting: The dataset is typically split into three subsets: training, validation, and testing. The training set is used to train the model, the validation set is used to tune hyperparameters and monitor performance during training, and the test set is used to evaluate the model's performance after training.
* Data Format: CNNs commonly accept input data in formats like JPEG, PNG, or sometimes more specialized formats like TFRecord or LMDB, depending on the framework you're using for training.

By addressing these aspects in data description, provide a comprehensive overview of the dataset's characteristics and ensure that it is suitable for training CNNs for image classification tasks.

# RESULTS

A graph of a train and a train

Description automatically generated with medium confidence

Fig1: Results for 5000 images

A group of graphs showing the results of a training course

Description automatically generated with medium confidence

Fig2: Results for 10,000 images

A close-up of a letter

Description automatically generated

Fig3: Results for 18,000 images

The output for 18,000 images as training dataset was tested on 200 test images.

# CONCLUSION

Convolutional Neural Networks (CNNs) have revolutionized the field of image classification, demonstrating remarkable performance in various tasks. Their hierarchical architecture, which includes convolutional, pooling, and fully connected layers, allows them to automatically learn meaningful features from raw pixel data, eliminating the need for handcrafted feature extraction.

Key advantages of CNNs include:

Feature Learning: CNNs automatically learn hierarchical representations of data, starting from simple features.

Translation Invariance: CNNs can detect features irrespective of their location in the input image, making them robust to translations, rotations, and scale variations.

Shared Parameters: CNNs exploit parameter sharing, where the same set of weights is used across different spatial locations, reducing the number of parameters and improving generalization.

Hierarchical Representation: CNNs capture hierarchical patterns in data, enabling them to understand complex relationships between features and make accurate predictions.

Transfer Learning: Pre-trained CNN models, trained on large datasets like ImageNet, can be fine-tuned on smaller, domain-specific datasets, allowing for effective transfer of learned features and reducing the need for large, labeled datasets.

Data Efficiency: CNNs require less preprocessing compared to traditional computer vision techniques, as they can directly learn relevant features from raw pixel data.

However, CNNs also have some limitations:

Data Requirements: CNNs typically require large amounts of labeled data for effective training, which may not always be readily available, especially for specialized domains.

Computational Resources: Training deep CNNs can be computationally intensive and may require specialized hardware such as GPUs or TPUs.

Interpretability: Understanding why CNNs make specific predictions can be challenging due to their complex, non-linear nature.

Overall, CNNs have emerged as a powerful tool for image classification, offering state-of-the-art performance across various domains and applications. Continued research in architecture design, optimization techniques, and interpretability methods is expected to further enhance the capabilities and understanding of CNNs in the future.

# REFERENCES

[1] Hongchun Lu, Min Han, Chaoqing Wang, Junlong Cheng, "AMLNet: Attention Multibranch Loss CNN Models for Fine-Grained Vehicle Recognition", IEEE Transactions on Vehicular Technology, vol.73, no.1, pp.375-384, 2024.

[2] Sha Lu, Kaixing Wu, Jinxin Chen, "Solar Cell Surface Defect Detection Based on Optimized YOLOv5", IEEE Access, vol.11, pp.71026-71036, 2023.

[3] Feiyang Li, Jiangtao Wang, "Remote Sensing Image Scene Classification via Regional Growth-Based Key Area Fine Location and Multilayer Feature Fusion", IEEE Geoscience and Remote Sensing Letters, vol.20, pp.1-5, 2023.

[4] Xu, Peifang Deng, Hong Huang, "Vision Transformer: An Excellent Teacher for Guiding Small Networks in Remote Sensing Image Scene Classification", IEEE Transactions on Geoscience and Remote Sensing, vol.60, pp.1-15, 2022.

[5] Yu Zhang, Tao Zhou, Wei Wu, Hua Xie, Hongru Zhu, Guoxu Zhou, Andrzej Cichocki, "Improving EEG Decoding via Clustering-Based Multitask Feature Learning", IEEE Transactions on Neural Networks and Learning Systems, vol.33, no.8, pp.3587-3597, 2022

[6] Kejie Xu, Hong Huang, Peifang Deng, Yuan Li, "Deep Feature Aggregation Framework Driven by Graph Convolutional Network for Scene Classification in Remote Sensing", IEEE Transactions on Neural Networks and Learning Systems, vol.33, no.10, pp.5751-5765, 2022.

[23:39, 4/17/2024] Pooja Reddy: Lin Bai, Qingxin Liu, Cuiling Li, Zhen Ye, Meng Hui, "A Lightweight and Fine-Grained Feature Fusion Network for Remote Sensing Scene Classification", 2021 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC), pp.01-06, 2021.

**IEEE conference templates contain guidance text for composing and formatting conference papers. Please ensure that all template text is removed from your conference paper prior to submission to the conference. Failure to remove template text from your paper may result in your paper not being published**