May 27, 2024

Joy Ngugi

Machine learning

Unit 11

neural networks in object recognition presentation transcript

Slide 1 : Title

This presentation is on the role of neural networks in object recognition. Specifically, in the application of Convolutional Neural Networks (CNNs) to the CIFAR-10 dataset.

Slide 2: Dataset Overview

Before going any further, we need to understand the dataset. CIFAR-10 was collected by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton in 2009.

It is named ‘CIFAR-10’ after the Canadian Institute for Advanced Research who funded the projected.

It consists 60,000 images each 32 by 32 pixels in size. These images are represented in three channels – Red, Green, and Blue, making them a rich source for training convolutional neural networks.

They are grouped into 10 classes, each representing different objects such as airplanes, automobiles, birds, cats, deer, dogs, frogs, horses, ships, and trucks.

With each of these classes containing 6000 images.

Slide 3: Data Preprocessing : Data Splitting

Getting into data preparation which was done in three steps: data splitting, normalistation and one-hot encoding, each step serving a specific and important purpose.

Data Splitting: The daset was divided into training, test and validation sets.

The training set contains 40,000 images, forming the majority of the dataset.

This allows the model to learn from a variety of features and examples, enhancing its ability to generalise across new, unseen data.

The validation set includes 10,000 images.

The rationale behind this is to have a set to fine-tune the model’s parameters and monitor for overfitting.

It acts as a checkpoint during training, to know when to adjust learning rates and other hyperparameters in order to optimise the model’s performance on unseen data.

Similarly, test set contains 10,000 images providing an unbiased evaluation of the final model.

This would, ideally, represent the real world scenarios.

Slide 4: Data Preprocessing : Normalisation

Moving on to normalisation.

The goal of normalisation is to transform the data into a uniform scale.

This is critical because it enhances the neural network's ability to converge more quickly and more stably during the training process.

To normalise the data, the pixel values of the images are scaled.

Originally, each pixel has a value ranging from 0-255, reflecting the intensity of RGB. To normalise these values, each is divided by 255 rescale the data in a range between 0 and 1.

This helps in speeding up the learning process of the model, as smaller, normalised numbers are easier for the model to process.

Slide 5: Data Preprocessing : One-hot encoding

Finally in the data preprocessing steps, One hot encoding, which is basically transforming the categorical class labels into a binary matrix format.

To do this, each class label is converted into a binary vector, that is all zeros except for a 1 at the index corresponding to the class.

This is very impactful as it allows the use of the categorical cross-entropy loss during training, which is highly effective for multi-class problems.

Slide 6: Data Preprocessing : One-hot encoding Images

These images show what the class labels were before and after the one hot encoding was applied.

Slide 7: Model Training: Artificial Neural Networks(ANNs)

Artificial Neural Networks are the backbone of CNNs.

ANNs are designed to simulate human brains.

So at the most basic level, an ANN consists of an input layer, several hidden layers and an output layer.

Each layer is made up of neurons and these layers are interconnected through nodes.

Data enters an artificial neural network through the input layer and is processed through multiple hidden layers using adjustable weights to enhance predictive accuracy.

The final output layer then converts these transformations into specific predictions, such as class scores for different categories like in the CIFAR-10 dataset.

The different types of ANNs include but not limited to Feedforward Neural Networks, recurrent Neural Networks and Convolutional Neural Networks.

Slide 8: Model Training: Convolutional Neural Networks Structure

Convolutional Neural Networks take advantage of the fact that the input consists of images and they constrain the architecture in a more sensible way.

They are engineered to automatically and adaptively learn spatial hierarchies of features, from low-level details to high-level patterns.

They are made of different layers

**Input layer** is essentially, in this case, the image.

**Convolutional layer** which is the core building block of a CNN.

This layer applies filters to the input to create feature maps. These filters detect specific features such as edges, shapes, or textures.

Each filter produces a different feature map, emphasising features that will help in the task of classification.

Following convolution, the **pooling layer** reduces the spatial size of the feature maps.

This reduction helps decrease the computational load, enhances the detection of features, and helps make the detection of features invariant to scale and orientation changes.

Common pooling methods include **max pooling**, where the maximum element is selected from the feature map region covered by the filter, and average pooling, which takes the average.

After several convolution and pooling layers, the high-level reasoning in the neural network is done through **fully connected layers**.

In a fully connected layer, neurons have full connections to all activations in the previous layer.

These layers essentially take an input volume (the output of the last pooling or convolutional layer) and output an N-dimensional vector where N is the number of classes being scored in the classifier.

The **output layer** is the last layer that outputs the predictions: image classes.

**ReLU** an activation function commonly placed after convolutional layer to introduce non-linearity. It is a technique of removing excess ‘noise’ from a picture in order to improve feature extraction.

Slide 9: Model Training: Why CNNs for CIFAR-10 dataset?

Its important to understand the reasons why CNN is the best suited model for the CIFAR-10 dataset.

**Preservation of Spatial Hierarchy:** CNNs excel in maintaining the spatial hierarchy between pixels by using convolutional filters that capture the local dependencies in an image. For instance, recognising that a certain pattern of pixels resembles an edge or a specific texture, which is critical in understanding and categorising images.

**Parameter Sharing and Efficiency:** A defining feature of CNNs is parameter sharing. The same filters are applied across different parts of the input image, which not only reduces the memory usage but also improves the efficiency of the model. This means that fewer parameters are needed, leading to faster and more efficient training processes.

**Depth Utilisation and Efficiency:** CNNs utilise layers with varying depths, which allows them to learn more complex features at higher layers. Early layers might capture basic features like edges and colors, while deeper layers can detect more complex features like shapes or specific objects, making it highly effective for image classification tasks.

**Adaptability to Image Data:** Lastly, CNNs are inherently adapted to image data. They can automatically and adaptively learn the most important features for classification, without the need for manual feature extraction. This makes them incredibly powerful for tasks like image recognition in the CIFAR-10 datset, where the dataset includes diverse images of animals and vehicles.

Slide 10: Model Training: CIFAR-10 CNN Model 1 Architecture

The model begins with a **Conv2D** layer with 32 filters of size 3x3 to capture basic image features.

Followed by a **MaxPooling2D** layer which reduces the spatial dimensions to half.

After the initial layers, the model includes additional **Conv2D layers** with an increasing number of filters—**64** and then **128**. Each convolutional layer is designed to capture more complex features in the image data. Each of these layers are followed by another **MaxPooling2D** layer to reduce the size of the feature maps, concentrating the information into more abstract representations.

Once the convolutional and pooling layers have extracted the features, a **Flatten** layer transforms the 2D feature maps into a 1D feature vector. This step is crucial as it prepares the data for the fully connected layers.

A **Dense** layer with 512 units follows, introducing the capability to learn non-linear combinations of the high-level features extracted by the convolutions. This layer is equipped with a **Dropout** layer with a rate of 50% to prevent overfitting by randomly dropping units during the training process.

The final layer in the architecture is another **Dense** layer, this time with 10 units corresponding to the 10 classes of the CIFAR-10 dataset, each outputting a score that corresponds to the likelihood of the input belonging to one of the classes.

This model has a total of **1,147,466 trainable parameters**, illustrating the complexity and depth of the network designed to handle the multi-class image classification task effectively.

Slide 11: Model Training: Model 1 Performance

With 10 epochs and batch size of 64 the model’s accuracy and loss is observed on the training and validation sets.

As seen in the chat the training accuracy continues to improve as training progresses, which is good.

However, the validation accuracy plateaus and even starts to decline slightly after a certain number of epochs.

This divergence suggests that while the model is getting better at predicting the training data, it is not improving on the validation set.

Similarly, the training loss decreases steadily, which is typical during training. However, the validation loss decreases initially but then begins to increase, which is a classic sign of overfitting. **This indicates that the model is starting to learn from the noise in the training data rather than generalizing from it.**

Slide 12: Model Training: Tuning Model 1

The model’s performance was also tested on 10 epochs and there was no improvement in the model’s performance. Infact the performace is almost identical to the 20 epochs.

Slide 13: Model Training: Model 2 Architecture

Turning the focus to a second model, with significantly increased complexity to enhance learning capabilities.

It consists of multiple convolutional layers, each followed by activation layers to introduce non-linearity. Several dropout layers are interspersed between the convolutional layers. Increasing the model’s overfitting regularization.

The Root Mean Squared Propagation (RMSprop) optimiser in this model is aimed at optimising the training process. It adapts the learning rate for each parameter and helps to minimise the loss function more efficiently compared to standard gradient descent.

Additionally, a learning rate schedule is employed—specifically, an exponential decay schedule that starts at 0.001 and decays every 100,000 steps by a factor of 0.96. This approach helps in fine-tuning the learning process, ensuring that the model learns effectively over time without taking too large or too small steps.

The model runs for 100 epochs, allowing sufficient time for the network to converge and learn from the data. The entire training process is steered towards minimising the categorical crossentropy loss, which is ideal for the multi-class classification problem. This loss function compares the predicted probability distribution across the 10 classes with the true distribution, enhancing the model's ability to predict accurately.

Slide 14: Model Training: Model 2 Performance

The performance of Model 2 showcases significant improvement with the accuracy plateauing at 82%, indicating that the model has effectively learned to recognize patterns from the training data. However, the validation accuracy stabilises at 78%, which is slightly lower than the training accuracy, suggesting a **modest overfitting** despite efforts to generalise. This gap between training and validation performance is indicative of the model learning features specific to the training set that do not generalise perfectly to new data.

The training and validation loss graphs further illustrate this point, with the training loss decreasing and plateauing at 0.51, reflecting the model’s ability to minimize error on the training set effectively. Conversely, the validation loss starts higher but then reduces to 0.65 and stabilizes, highlighting the gap in performance on unseen data.

Slide 15: Model Training: Improving Model 2

To deal with the overfitting sevaral strategies were employed.

First, Batch Normalisation was incorporated across multiple layers. This technique normalizes the inputs of each layer to improve the stability and speed of the network's training by maintaining a consistent distribution of activation values throughout the model.

Next, Data Augmentation. By artificially expanding the training dataset through transformations like rotations, shifts, and flips, the model does not learn too specifically from the training data, enhancing its ability to generalise.

Early Stopping was employed. This method monitors the model's performance on a validation set specifically the validation loss and halts training when performance ceases to improve, thereby preventing overtraining and conserving computational resources.

The Adam optimiser was chosen for its efficiency in handling sparse gradients and adaptive learning rate capabilities, which contribute significantly to the rapid convergence of training.

Finally, Dropout was increased from 0.25 to 0.3

Slide 16: Model Training: Model 3 Performance

There has beensignificant strides in performance and efficiency:

Firstly, the introduction of Early Stopping played a crucial role. Initially the epochs were set out for 100 epochs, but the training was halted at 44 epochs because there were no further improvements in validation loss. This decision not only conserved resources but also prevented potential overfitting, thereby enhancing the model’s learning efficiency.

Moving on, the model displayed remarkable **Stability and Convergence**. The learning curve was smooth and consistent, indicating that the model was well-tuned and effectively adapting to the training data, which is vital for robust model performance in practical scenarios.

Regarding **Overfitting Management**, this model showed a superior capability in handling overfitting compared to its predecessors, especially Model 2. This is evident from the much closer training and validation accuracy and loss figures, indicating that the model's predictions are reliable and not just memorised responses to the training data.

Lastly, **Enhanced Generalisation** was achieved. The higher validation accuracy and notably lower validation loss underscore the model’s ability to perform well on unseen data, suggesting it has learned truly predictive features rather than merely fitting to the noise in the training dataset.

Slide 17: Model Evaluation: Classification Report

After settling on a well performing model, the model was evaluated on different metrics, the classification report and confusion matrix. There is outstanding performance observed in ‘automobile’ and ‘ship’ classes with F1-scores of 0.90 and 0.94 respectively.

However, the ‘cat’ class remains a challenge, demonstrating lower recall and F1-score.

Overall, the model shows high specificity across most classes, but shows varied sensitivity, reflecting some inconsistency in classifying certain categories effectively.

Slide 18: Model Evaluation: Confusion Matrix

The confusion matrix reveals notable insights into model performance across different classes. Classes such as 'Frog,' 'Automobile,' and 'Airplane' show high true positive rates, indicating strong model accuracy for these categories. However, 'Cats' pose a significant challenge, with frequent misclassifications as 'Frogs' and 'Dogs,' reflecting confusion with visually similar features. This issue also extends to 'Trucks' and 'Automobiles,' where the model struggles to differentiate between these related categories.

Slide 19: Recommendation

In light of the challenges and opportunities identified through the model's performance analysis, several strategic enhancements are recommended:

**Enhanced Feature Engineering:**

Improving feature extraction processes can help in capturing more relevant information from the images, which could be crucial for distinguishing between similar classes such as cats and dogs.

**Data Augmentation for Problematic Classes:**

Specifically targeting underperforming classes like 'Cat' with techniques such as image rotation, scaling, and cropping can help the model learn more robust features.

**Algorithm Tuning:** Further tuning of the model’s parameters and optimisation algorithms, including the learning rate and optimiser choice, could lead to better convergence on lower error rates.

**Advanced Techniques:**

Implementing deeper convolutional neural network architectures might allow the model to learn more complex patterns and subtle distinctions between classes, significantly enhancing classification accuracy.

Slide 20: Conclusion

In conclusion, through this project, convolutional neural networks (CNNs) have proven to be exceptionally effective for image recognition tasks due to their ability to preserve spatial hierarchies in data.

Despite the vast volume and diversity of data like that in the CIFAR-10 dataset, challenges such as overfitting and the intensive computational demands persist.

Yet, the adaptability of CNNs to a wide range of applications—from autonomous driving to medical image computing—demonstrates their significant potential.

Moving forward, addressing these challenges through improved model training strategies and optimisation techniques will be essential to fully leverage CNN capabilities in practical and diverse real-world scenarios as evidenced by this project’s findings.