Group 1 – Presentation transcript

Slide 1 - Title

This presentation is the Group 1 submission for the neural network models for object recognition assignment. Completed by Ruth, Nicholas, Tasween, Danilo and Lojayne. Due to challenges caused by an eleven hour time difference between group members a separate audio file has been recorded for each slide.

Slide 2 - Introduction

In an increasingly digitized world, the ability to comprehend and make sense of digital data is a critical component of many applications, such as self-driving cars or medical diagnostics. According to Voulodimos (2018), deep learning allows computational models to learn and represent data mimicking how the brain processes information to artificial neural networks. In this presentation, we will begin by reviewing how an artificial neural network is created for the task of computer vision. And we will then train a model to recognize images using the CIFAR-10 data set.

Slide 3 – The rationale for having a validation set

The rationale for having a validation set.

Definitions:

- Training set – a sample used to fit the model
- Validation set – The validation set plays a crucial role in preventing bias during model evaluation
- Test set – sample used to provide an unbiased evaluation of the final model fit

The incorporation of a validation set in machine learning is underpinned by a strategic rationale aimed at enhancing the model development and evaluation process. In a formal context, the following key justifications exist:

- Hyperparameter tuning: By evaluating the model's performance on the validation set, practitioners can choose the optimal hyperparameters, a process known as hyperparameter tuning.
- Overfitting mitigation: When the model's performance on the validation set differs from the model's performance on the training set (for example, a decrease in the former but an improvement in the latter), this indicates overfitting.
- Model Selection: By evaluating the performance of each model in the validation set, practitioners can make informed choices about which model shows superior performance.
- Bias Mitigation: It ensures that the evaluation of model performance remains unbiased and unaffected by the training data. Incorporating validation data into model construction can introduce bias, which is postponed until the final model is selected.

As machine learning is a highly iterative process, multiple models are tested using the training dataset. The validation set is useful for fine-tuning the hyperparameters or selecting the best model following the initial training. It is then incorporated into the model before the final test dataset is used.

Slide 4 – Details of the training and validation sets

Details of the data validation set training set

A commonly mentioned ratio is to split the data 60-80% for training, 10-20% for validation, and 10-20% for testing. In this exercise, we have used 75% : 15% : 10%

The dataset consists of 60,000 32x32 colour images containing one of 10 object classes, with 6000 images per class. It is split into a test set containing 10,000 images, and a train set containing 50,000 images. The validation set was created by removing 5,000 random images from the training dataset.

Training Set (75%): The training set was allocated 45,000 images, which is 75% of the entire dataset. This substantial training set size allows the model to learn from a vast amount of data, thus gaining the capacity to capture intricate patterns in the images.

Validation set (15%): To create the validation set, 4800 images were randomly selected from the training dataset. This validation set plays a crucial role in tuning the model and evaluating its performance during training.

Test set (10%): The test set consists of 10,000 images, which represents 17% of the total dataset. This separate test suite remains unchanged during the model development process and is only used for the final evaluation of the trained model.

Finally: Choosing a validation set size of 8% depends on the specific needs of the project. However, it is important to realize that allocating data to training, validation, and test sets is a critical decision in machine learning, and must be made taking into account several factors, including data set size, problem complexity, and hyperparameter tuning requirements.

Slide 5 – Structure of the ANN

This task uses a convolutional neural network, or CNN, for image classification. This is a regularized type of feed forward neural network. That learns feature engineering by filter or kernel optimization. The neural network is composed of two parts, a base for feature learning and a head for classification, as shown in figure one.

Slide 6 – CNN Base

The base of the CNN focuses on the feature learning. Holbrook and Cook (2019) say this takes place in a three step process of filtering, detecting and pool. Filters or kernels act as lenses that scan the input image and give a weighted sum of the pixel values. This is shown in figure two. A range of different filters are used to identify different features. Using different filters on the same image creates a feature map of the image giving our first convolutional layer, LeCun and Bengio (1995) say that once a feature is detected, its exact location is less important. So the convolutional layer is followed by a pooling layer to reduce the feature map's resolution, reducing the sensitivity of the output to shifts and distortions. The pooling layer is shown in figure three. Aloysius and Geetha (2017) say that this also helps to reduce computational costs and prevents overfitting. Alternating convolutional and pooling layers are often repeated to find features within the pooled feature maps.

Slide 7 – CNN head

The head of the CNN consists of fully connected layers Akbar et al. (2017) says it's required to learn non-linear combinations of learned features and allow for high level reasoning. Neurons are connected to all neurons in the previous layer and are one dimensional as shown in figure four. Therefore, convolution cannot occur after a fully connected layer. The final fully connected layer is the outputs and gives a probability for each desired class and can also be used in calculating the loss.

Slide 8 – Activation function

According to Jain (2019), an activation function is added to a neural network to help the model learn complex patterns. A weighting is applied to the input and if the specified threshold is reached, the output is triggered. So we have used two different activation functions. The rectified linear units ReLU activation function was used for the hidden layers. This is one of the most widely used activation functions in deep learning models. So unlike the sigmoid function, the ReLU function is unbounded. It's not limited to values between zero and one. This reduces the problem of vanishing gradient which is found with the sigmoid function when using multiple hidden layers. The ReLU activation function has previously been found to have high performance according to Obla et al (2020) with this particular dataset. The Softmax activation function was used for our output layer. Softmax regression is a generalization of logistic regression and is used when there are multiple classes according to Stanford University.

This normalizes the elements of the output vector and can be interpreted as a probability distribution. This means that instead of using mean squared error for the model, we can now instead measure the loss function.

Slide 9 - Epochs

Defining the optimal number of epochs in a convolutional neural network (CNN) is a critical task in training deep learning models. The number of epochs represent the number of times the entire dataset is passed forward and backwards to the network (Soon et al 2018). According to Ali et al (2021) , selecting a lower number of epochs helps reducing the computational time of the model, but on the other hand, a large number of epochs may sometimes lead to a model overfitting. In general the CNN model traing should also include other hyperparameter tuning, which strongly reflects the model performance Goel et al (2021).

Nurnoby et al (2020) stated that overfitting is a common problem in CNN model training. For that reason, he proposed to apply early stopping rules with a defined patience value during the training to avoid overfitting. Taking all the above into consideration of the proposed model was trained with a 150 epochs and an early stopping criteria with patience of 10, besides the other hyperparameters. As a consequence, the model reached a stable performance after 45 epochs, for both training and validation data.

Slide 10 – Loss function

There are two main types of loss function which correlate to the two major types of neural network - regression and classification models. According to Yathish (2022) a loss function measures how well the neural network models the training data, and the loss function is used to determine the quantity that should be minimized in training. According to TensorFlow, (no date) the categorical cross entropy loss function should be used for multi class classification models, which is what we have here. We did use this initially, but after initial testing when we moved on to the later versions of our model, we changed to the sparse categorical cross entropy loss function. This is better for samples where the classes are mutually exclusive. Again, this is the case with the current data. Plotting loss against time indicates how well the model performs. So figure one here shows that the model performs well for the training data which is the blue line in the initial model. We have a steady decrease in losses. But when the validation set is added in, while this decreases initially, it does start to increase again. According to Baeldung (no date), this shows that the model is overfitting to the training data. By modelling the training data very narrowly, it can't recognize when there's variance in the validation set. So this overfitting means that actually the model is getting worse at predictions when additional data is added.

There are a number of adjustments that can be made to address this overfitting, such as increasing the number of filters, changing the kernel size, increasing the number of neurons or the number of hidden layers, as well as adding a dropout layer. By careful adjustment of these factors, we reduced overfitting as shown in figure two. Once this issue was addressed, we could then go on to improve the accuracy of the model.

Slide 11 – Choice of CNN

CNNs, represent a sophisticated approach to image analysis. Their efficacy is particularly highlighted when processing complex datasets such as CIFAR10, emphasized by Akwaboah in 2019.

The choice of CNN and its components are not arbitrary. Central to the CNN's architecture is the convolutional layer. As explained by Goodfellow et al. in 2010, this layer operates using filters that systematically slide over an image to extract localized features. These filters identify patterns, such as edges or textures, which are foundational to understanding the image's content.

But recognizing patterns isn't enough. The pooling layer's role, as emphasized by Alwabi et al. in 2017, is pivotal. It distills the image's features, retaining dominant patterns while reducing data dimensionality. This ensures computational efficiency without sacrificing the quality of feature extraction.

With such refined data, dense layers take over, processing and interpreting the distilled information to produce a coherent classification or prediction of the image.

Slide 12 – Enhancing CNN efficiency

However, the choice of CNNs isn't just about its structural layers. It's also about the mechanisms that drive its learning. The ReLU activation function, as highlighted by Alzubaidi et al. in 2020, ensures the network is dynamic, introducing non-linearity that's pivotal for complex pattern recognition.

And training a CNN? It's made efficient with the Adam optimizer. As pointed out by Kingma & Ba in 2014, Adam adaptively modulates learning rates, ensuring that the network's training is not just swift, but also precise.

In summary, CNNs stand at the intersection of strategic architectural design and computational efficiency, marking a significant advancement in the field of image analysis.

Slide 13 – Comparison of models

Now, we will compare the two distinct CNN architectures tailored for the CIFAR-10 dataset. Starting with Model 1, we have a streamlined design. The input images, each of size 32×32×3, pass through two convolutional layers. The absence of dropout or batch normalization in this model makes it straightforward and computationally efficient.

On the other hand, for Model 2 it is evident that this design is more intricate. With the same input shape, the images encounter four convolutional layers. What's noteworthy is the inclusion of batch normalization after each convolution, which aids in stabilizing and accelerating the learning process. To mitigate overfitting, we've incorporated a series of dropout layers with rates incrementing from 20% to 50%.

In essence, while Model 1 offers a rapid and efficient approach and Model 2 delves deeper, employing techniques to capture finer details and ensure model robustness. The choice between these models depends on the balance one seeks between computational speed and nuanced pattern recognition.

Slide 14 - Conclusion

In summary, this work proposed a Convolutional Neural Network design for object recognition using CIFAR-10 dataset available on Kaggle (Sharma, 2020). Our initial setup achieved an accuracy of 76% however suffered from over-fitting. To address this challenge, we fine-tuned the network by optimizing the number of filters, adjusting the neuron configuration, and introducing a dropout layer. Through these modifications, we substantially improved the model's accuracy, reaching an impressive 87.38%, while effectively mitigating over-fitting.

One significant challenge we encountered while striving to enhance accuracy was the notable increase run time for each epoch. Increasing the number of filters used gave more accurate results but had a dramatic impact on the run time leaving the program at risk of interrupted connections and failure. While increasing accuracy was wanted it had to be balanced with the stability and practicality of the development environment.

To further enhance the proposed model's performance and minimize loss, future works can explore advanced techniques such as data augmentation and leverage transfer learning from pre-trained models. These strategies offer the potential of pushing the boundaries of our model achieving even greater results.

Furthermore, techniques such as data augmentation and the adoption of pre-trained models through transfer learning can be explored to enhance the proposed model even further.

Slide 15 – References

The next slides show our references. Thank you for listening.