University Of Sunderland

MSc Computer and Data Science

Giulio Leopaldi

CETM 26 - FINAL ASSESSMENT

NEURAL NETWORKS FOR IMAGE RECOGNITION

Total word count: 3433

1. INTRODUCTION

Computer vision is a field of artificial intelligence (AI) which has seen massive development in the latest years. It enables computers to elaborate, recognise and extract meaningful information from visual inputs like images or videos, and act based on that.
A lot of innovative deep learning techniques capable of excellent results have been proposed recently, although often requiring very powerful systems and high memory capacity. This research project aims to create models which can achieve good results (identified as achieving over 50% accuracy) in a short computational time by modeling an Artifical Neural Network (ANN) and a Convolutional Neural Network (CNN) to verify and compare the capability of these systems in image recognition. These models are then optimised to improve the result and tackle overfitting. The dataset taken into consideration is the Cifar-10, pubblicly available on UCI repository (CIFAR-10 and CIFAR-100 datasets, 2022), a widely studied dataset to test image recognition models.
An overview of state-of-the-art models and techniques currently available in scientific literature are described in Section 2. The description of the methodology and experiments conducted in this project can be found in Section 3 and 4, whilst Section 5 is for the discussion and conclusion that can be drawn. The final section (Section 6) explores common sectors of computer vision application from an ethical and moral perspective.

2. BACKGROUND

The recent great development of innovative algorithms and machine learning techniques has taken computer vision to a new state-of-the-art level, especially with deep learning models. This progress has been already included into several practical applications of daily life, such as biometric face-recognition, medical images classification, self-driving cars, and the market is continuing to grow. Considerable attempts and excellent results have been obtained so far. For instance, Krizhevsky et.al built an architecture based on a CNN structure for high resolution images. In order to represent the image features, 60 million parameters and about 50000 neurons composed the model. Although very powerful, it requires an extensive memory and a long training time (Krizhevsky, Sutskever and Hinton, 2017).
The Cifar-10 dataset is a collection of images and one of the most widely used datasets to practice and test image recognition models. It consists of 60000 images, each of 32x32 pixels of ten different classes, evenly distributed. Each image is distinctive of its own class, it can not be confused or included in any other class. Models capable of achieving impressive accuracies have been developed for this task. The most successful model for this dataset so far (Papers with Code - CIFAR-10 Benchmark (Image Classification), 2022) has been developed in 2021 by Touvron et al. (Touvron et al., 2021), who designed an architecture named Deeper Transformer Network capable of achieving a remarkable accuracy of 99.4% on this dataset. Most of the top-scorers have developed what are commonly called “very deep models”, which are extremely intricated and complex models, including a conspicuous amount of layers, transformations and parameter tuning that require very powerful systems to be able to run them within a reasonable amount of time. State of the art techniques include Deep Convolutional Networks, Residual Networks or Image Transformers.
More common approaches for image identification are CNN, which are neural networks with at least one convolutional layer for visual pattern recognition (Pandiya, Dassani and Mangalraj, 2020). The architecture has generally four identifiable layers:

1) The convolution layer: this is the main building block and contains a set of filters (or kernels) usually smaller than the image which is passed onto the image to extract features and learn patterns.
2) The pooling layer: this layer has the purpose of down sampling the spatial inputs of the image to reduce the number of computations.
3) The activation layer: like for ANN, the activation functions are mathematical operations that determine the output of the network. They act like a gate that determines whether the neuron will fire or not, based on the outcome vaue.
4) The fully connected layer: this forms the last layer in the network and indicates a probability of the input belonging to a certain class or another.

The more advanced techniques have allowed reaching an almost perfect result with the CIFAR-10 dataset; it is likely to see a shift of future benchmarking from further improving the accuracy towards discovering lighter and more time-efficient models (Smith, Amaral and Heywood, 2021).

3. METHODOLOGY AND EXPERIMENTS

The aim of this research project has been focused on finding a model that would provide a good rate of success, but also remain within a reasonable order of simplicity and execution time.
The criteria of success has been determined by comparing the validation accuracy of the models. Considering that the dataset has 10 different classification instances, by randomly guessing there would be a 10% probability of being correct, therefore any model that provides an accuracy of over 10% can be considered a positive improvement. In this case, a simple and light model which can provide an accuracy of over 50% would be considered a good result.
In scientific literature it is recognised how CNN seem to be among the best techniques for image recognition models. However, to achieve high or very high-performing results, these can be daunting in terms of time, system requirements and optimisation. This project wants to explore if ANN can be a valid alternative for achieving good results, which steps can be taken to improve the performance, and then make a comparison with a CNN model built on top of a similar structure.

Software description and pre-processing
This research project was created by using Google Colaboratory (Colab) notebook, which is like a Jupyter notebook, but runs in the cloud. It is integrated with other Google platforms, like Google Drive, therefore it benefits from a very easy set-up, access and sharing of projects. It has several Data Science libraries already installed and it gives the opportunity of taking advantage of the powerful computational capabilities of Google servers at the cost of a Premium membership fee, although this was not used for this project (Google Colaboratory, 2022).
The models have been created with Python, a very powerful open-source language and one of the most commonly used which was created in the 1980's with the idea of making a general-purpose and accessible language (3.10.4 Documentation, 2022). Being extremely popular, a large community supports the development of the language and the creation of a conspicuous amount of libraries that can be used for a number of different tasks.
The ones that have been selected in this project are the following:

- NumPy: a library for scientific computing, used for working with arrays, matrices and performing operations of linear algebra (NumPy Documentation, 2022).
- Pandas: this library allows data manipulation and analysis. Useful for exploring the dataset and performing pre-processing operations (pandas documentation - pandas 1.4.2 documentation, 2022).
- Matplotlib: a versatile and intuitive library for creating static, animated, and interactive visualizations in Python (Matplotlib — Visualization with Python, 2022).
- Tensorflow: a collection of workflows to develop and deploy machine learning models (TensorFlow, 2022).
- Keras: a high-level neural network library developed in Python that runs on top of TensorFlow. The ANN, CNN and hyper parameters tuning (with the keras.tuner extension) have been created with this library (Keras, 2022).

The CIFAR-10 dataset contains 60,000 32x32 colour images representing 10 different, mutually exclusive classes, which are: airplane, automobile, bird, cat, deer, dog, frog, horse,

ship, and truck. It can be downloaded and imported from the UCI repository or, since it is integrated in the keras library, it can simply be loaded in the notebook.
The dataset sits in a format which requires minimal pre-processing or manipulation to be ready for use. Moreover, it comes already divided into training and testing subsets with an 80/20 ratio: 50,000 instances for training and 10,000 for testing.
The first operation needed is normalisation; a scaler could be applied or, more simply, dividing the train and test sets by 255. In fact, each pixel of a coloured image is composed by a 3 dimensional array which represents a combination of the colours red, green and blue in a spectre that ranges from 0 to 255. The second operation necessary before starting to build the models is to apply a one-hot encoding: by creating a binary column for each label, it returns an array which is required in order to remove a relational link between elements. In fact, if these were labeled by integers it would potentially cause errors to the model. Applying a one-hot encoding to convert the labels into categorical values is very simple in Keras with the function “to_categorical()”.

Creation of the models

To start creating the different models, some intial fixed paramaters and hyperparameters have been chosen:

- The model type is set to “Sequential”, which is appropriate for a linear stack of layers to provide the training and the prediction to the model.
- A “Flatten()” layer is included before the neural network layers. It flatten the inputs, transforming the 2D image into a 1 dimensional array.
- Hidden layers and output layer are connected with “Dense” modality.
- The activation functions of the hidden layers are set to “relu”, (rectified linear unit activation function) which generally is more computationally efficient than other functions (Van Cauwenberge, 2018).
- Since it is a multi-class classification problem, the output layer activation function is set to “softmax”. It determines the predicted class based on a multinomial probability distribution.
- The loss function is set to be “categorical_crossentropy”, given the nature of the problem to be solved where an example can belong only to one out of the ten possible categories.
- The chosen optimiser is “Adam”, as it generally has faster computation time, lower memory requirements and results are very satisfying compared to other algorithms. It is often set as a default optimiser for many applications (Gupta, 2021).
- The metric that will be monitored is “accuracy” since it is the most relevant metric for this task.

To answer the research question, this project has followed four steps:

1) Creation of a baseline ANN
2) Optimisation of the baseline model by adding a drop out regolarisation and tuning the learning rate
3) Creating a CNN built upon the previous network
4) Optimising the CNN model

Step 1 - Creation of a baseline ANN

This projects aims to find a good model which is, at the same time, lightweight. It is known that a considerable number of hidden layers can dramatically increase computation time. In

the same way, the amount of neurons in each layer can have the same effect.
It is decided to use “keras.tuner” to find the best combination of number layers and neurons, limiting the number of layers to maximum 4, and the neurons between 128 and 512, with a 128 step. The tuner is built with “RandomSearch”, which runs the models with random hyperparameters among the ones selected. The number of maximum trials is set to 10 and execution per trial to 1 to maintain the training within reasonable time. The number of epochs is set to 20, although an “Earlystopping” callback with a patience of 2 is applied, to cut down training time where gains start being minimal. A computational time of 36 minutes allowed to discover the best model, corresponding to a 2 hidden layer network with 512 and 384 neurons in each of them. The validation accuracy achieved is 49.7%.

Part 2 - Tuning and improving the baseline ANN

An improved model is created by tuning the baseline. A “keras.tuner” and, again, a “RandomSearch” have given an option to test a Dropout regularisation and different learning rates on top of the most successful model of the previous step.
The Dropout option has been randomly set to “True” or “False” and to a value of 0.2 which means that, when applied, 20% of the connections are being switched off. The options for the learning rate are 0,01 and 0,0001. Same as before, the number of maximum trials has been set to 10 and execution per trial to 1, with 20 epochs per trial.
The best model, obtained in a computational time of 12 minutes, performs with a validation accuracy of 52.3% and it is achieved without Dropout and a learning rate of 0.0001.

Step 3: CNN

This step has the aim to create a CNN by adding one convolutional and one pooling layer on top of the previously obtained network.
It is a very typical scheme to have a convolutional layer followed by a pooling one, it optimises the result and improves computational time (Giusti et al., 2013). Therefore, a Conv2D layer with 32 filters and a kernel size of 4,4 is added. Also in this case the activation function is kept fixed with “relu”. A MaxPool2D layer with a pool size of 2 by 2 follows, which reduces the number of samples of the input by taking the maximum value over an input window for each channel of the input. Number of epochs had been set to 100, but an Early_stopping callback with a patience of 2 has stopped the processing after 11 epochs. This model has achieved a score of 68.2%.

Step 4: CNN regularisation

The previously obatained CNN has been regularised with a 0.3 Dropout and a 0.0001 L1-regularisation in each layer, in an attempt to improve overfitting (Yıldırım, 2020; Srivastava et al., 2014). The model obtained achieved a score of 70.2% after 19 epochs.

04. RESULTS

The measure of success for this project is the validation accuracy metric. The best results obtained in each step are summarised in Tab.1:

Table 1

RESULTS

	Model	Features	Validation accuracy
Step 1	Baseline ANN 512/384	2 layers, 512/384	49.7
Step 2	ANN optimised	Dropout: False, Learning rate 0.0001	52.3
Step 3	CNN	1 Conv2D + 1 MaxPool2D layer	68.5
Step 4	CNN optimised	Dropout + L1 reg	70.2

Tab.1: Summary of the results of the best models for each step

The following plots show the accuracy and validation accuracy trend per epoch for each model:

Figure 1

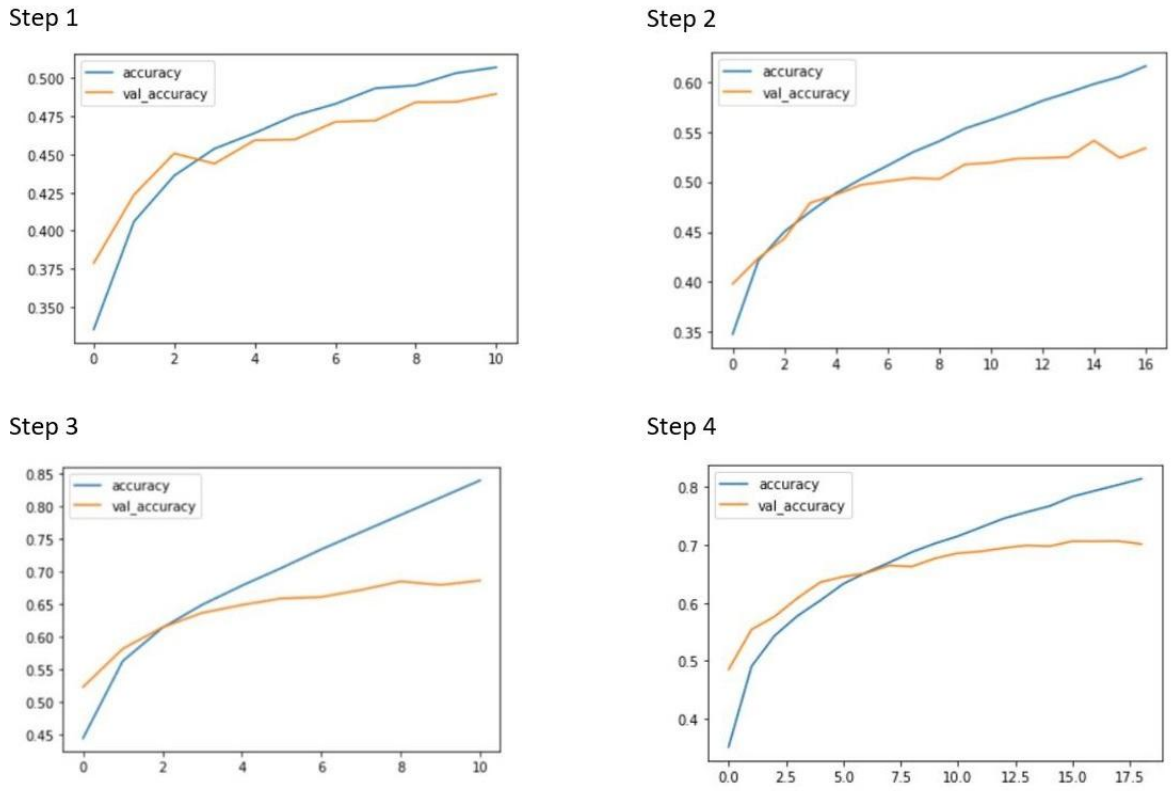

Fig.1: Plots of accuracy and validation accuracy per epoch

05. DISCUSSION AND CONCLUSION

The aim of this project has been to solve an image recognition problem with an ANN which could provide a “good” result, described as reacing an accuracy of over 50%. Subsequently, comparing it to a CNN built on top of the ANN to verify whether the improvements would be significant.
During step 1 the tested models have seen results that varied from a validation accuracy of 45%, to a best performance of 49.7%. This result highlights how much different architectures can influence the model performance and the importance of finding an optimal one by optimising the number of layers and neurons in each layer.
The best performing model in this step has been taken as the baseline model to be compared to the following models. As it can be observed in Tab.1, it achieved a validation accuracy of 49.7%. This does not yet satisfy the initial description of a good model, one that can achieve over 50% on the validation accuracy, but it is very close. As it can be observed in Fig.1, the convergence of the accuracy and validation accuracy is rather close, suggesting that the model should generalise well.

Step 2 has seen an optimisation of the baseline model by adding Dropout regularisation and testing different learning rates. The validation accuracy improved by almost 3%, bringing it up to 52.3% (a 5% improvement from the baseline). This step has showed results which differed widely. In fact, it could be observed how the models tested by the “RandomSearch” with learning rates of 0.01 achieved poor results (only 10% and 17.8%) . This highlights the importance of selecting a proper learning rate. The inclusion of the 0.2 Dropout regulator did not seem to affect positively the outcome, as the model where this was applied with the same learning rate achieved a lower score (49.9%) (Appendix - Fig.3). However, Fig.1 shows an increased distance between the accuracy and validation accuracy curves, suggesting that the best model has some degree of overfitting. It is plausible that including other regularisation techniques, the model can be optimised.

Step 3 has seen the implementation of a Convolutional layer and a MaxPooling layer with default values on top of the previous network. The improvement in validation accuracy has been consistent, bringing it up to 68.2% (a 37% improvement from the baseline model). However, it can be observed (Fig.1) that there is a significant discrepancy between the accuracy and validation accuracy which might suggest that the model does not generalise well.

Step 4 has tried to address overfitting by adding some regularisation hyperparameters. Specifically, including a 0.3 Dropout and a L1 regularisation of 0.0001 in each layer. This has improved the overall score to 70.2% (a 41% improvement from the baseline model). The accuracy and validation accuracy curves are significantly closer, therefore the overfitting problem has seen progress.

Considering these 4 steps it can be concluded that an ANN can perform well in tasks of image recognition. A good model (> 50%) can be obtained without being overwhelmed with the number of options and hyperparameters to be tuned (52.3%), although further research need to be performed to improve the overfitting. A CNN model is capable of sensibly improve the score (68.5%), and applying a dropout and L1 regularisation not only improves

the performance (70.2%), but also brings a satisfactory improvement to the overfitting problem.
This project confirms how a CNN can perform better compared to an ANN to solve image recognition problems and how it should be the model of choice for solving these type of tasks, but an ANN is still capable to achieve good results and is lighter in terms of less computational timing. Future research can be focused on further improving overfitting by including a batch size, data augmentation, adding other convolutional and pooling layers, or trying a different optimser. In fact, according to Zhou et al., SGD optimiser may generalise better than Adam on new, unseen data and this is something that could be explored (Zhou et al., 2021).

06. REFLECTION ON ETHICAL USE OF AI

The great evolution of Artificial Intelligence and its different uses has had a substantial impact on peoples lives across many different sectors. This comes with important ethical implications which need to be addressed.
One of the main usage of computer vision is for biometric facial recognition. This has showed great potential, for instance, to solve crimes by identifying offenders from photographs or closed circuit television. However, ethical and privacy implications come in conjunction with these systems. For instance, in 2020 Governments used billions of social media images to identify suspects, leaving the question of whether it is acceptable to collect biometric information from private citizens without their knowledge or consent, although for a good cause like justice (Smith and Miller, 2021).
Medical imaging is another sector where computer vision is revolutionising the field. Virtuous use of it comes from automating the reading of medical images which can lead to an early and more accurate detection of cancer or other diseases, as well as relieve doctors from having to manually do it and freeing them time for higher tasks. However, to build effective predictive models, a huge amount of data needs to be available in the form of patients imaging datasets. Like with any type of private medical information, there are many barriers to obtain them. The Eu General Data Protection Regulation (General Data Protection Regulation (GDPR) – Official Legal Text, 2022) states that patients need to give explicit consent for allowing the use or sharing of their own private medical information. Programs like “The All of Us” launched by the National Institute of Health have the purpose of creating databases of medical records by involving over one million contributors from diverse backgrounds to share their health data for the benefit of important advancement in Healthcare (National Institutes of Health (NIH), 2022).
A field where computer vision is bringing important advancements is self-driving vehicles, which hold out the promise of being safer than regular ones. However, 100% safety can not be guaranteed and incident scenarios have to be part of the programmed system. The ethical question arise in relation of how to deal with them: should the car be programmed to minimise harm to their owners, or respond and act on the basis of other principles? (Hansson, Belin and Lundgren, 2021).
An area of application of computer vision where big debates are held is the military sector. On one hand, autonomous weapons have the advantage of employing robots instead of human beings to accomplish military goals, without putting at risk the lives of soldiers. On the other hand, military objectives often include the destruction of strategic targets and loss

of human lives, both on the military and civil side. The Human Rights Watch (Losing Humanity, 2022) has highlighted the dilemma of whether a machine should be given the power to decide over the life or death of a human being, since it does not hold the cognitive and empathic capability to understand the context and judge deeply each situation. It would not be consistent with international humanitarian law, increasing the risk of death or injury to civilians during armed conflict (Roach and Eckert, 2020).
Image recognition and computer vision is changing our daily lives, but each of its applications need to be evaluated and regulated from an ethical perspective.

APPENDIX

Figure 2

```
# comparing all best results obtained
tuner.results_summary()
```

```
Results summary
Results in my_dir/colab/baseline
Showing 10 best trials
<keras_tuner.engine.objective.Objective object at 0x7fc119d9aa50>
Trial summary
Hyperparameters:
layers: 2
units0: 512
units1: 384
units2: 256
units3: 128
Score: 0.49720001220703125
Trial summary
Hyperparameters:
layers: 4
units0: 512
units1: 512
units2: 128
units3: 384
Score: 0.4844000041484833
Trial summary
Hyperparameters:
layers: 2
units0: 384
units1: 384
units2: 512
units3: 256
Score: 0.4839000105857849
```

Fig.2: Displaying the best trials of step 1

Figure 3

```
# comparing all best results obtained
tuner.results_summary()
```

```
Results summary
Results in my_dir/colab2/ANN tune dropout and lr
Showing 10 best trials
<keras_tuner.engine.objective.Objective object at 0x7f16cc31d590>
Trial summary
Hyperparameters:
dropout: False
learning_rate: 0.0001
Score: 0.5231000185012817
Trial summary
Hyperparameters:
dropout: True
learning_rate: 0.0001
Score: 0.49900001287460327
Trial summary
Hyperparameters:
dropout: False
learning_rate: 0.01
Score: 0.17890000343322754
Trial summary
Hyperparameters:
dropout: True
learning_rate: 0.01
Score: 0.10000000149011612
```

Fig.3: Displaying the best trials of step 2

Figure 4

```
Epoch 1/100
1563/1563 [==============================] - 85s 54ms/step - loss: 1.5499 - accuracy: 0.4439 - val_loss: 1.3269 - val_accuracy: 0.5223
Epoch 2/100
1563/1563 [==============================] - 78s 50ms/step - loss: 1.2330 - accuracy: 0.5622 - val_loss: 1.1855 - val_accuracy: 0.5810
Epoch 3/100
1563/1563 [==============================] - 77s 49ms/step - loss: 1.1055 - accuracy: 0.6131 - val_loss: 1.1094 - val_accuracy: 0.6138
Epoch 4/100
1563/1563 [==============================] - 77s 49ms/step - loss: 1.0042 - accuracy: 0.6484 - val_loss: 1.0492 - val_accuracy: 0.6360
Epoch 5/100
1563/1563 [==============================] - 77s 49ms/step - loss: 0.9221 - accuracy: 0.6777 - val_loss: 1.0069 - val_accuracy: 0.6481
Epoch 6/100
1563/1563 [==============================] - 76s 49ms/step - loss: 0.8436 - accuracy: 0.7049 - val_loss: 0.9989 - val_accuracy: 0.6582
Epoch 7/100
1563/1563 [==============================] - 76s 49ms/step - loss: 0.7712 - accuracy: 0.7333 - val_loss: 0.9968 - val_accuracy: 0.6603
Epoch 8/100
1563/1563 [==============================] - 77s 49ms/step - loss: 0.6970 - accuracy: 0.7597 - val_loss: 0.9724 - val_accuracy: 0.6712
Epoch 9/100
1563/1563 [==============================] - 76s 49ms/step - loss: 0.6260 - accuracy: 0.7860 - val_loss: 0.9569 - val_accuracy: 0.6842
Epoch 10/100
1563/1563 [==============================] - 76s 49ms/step - loss: 0.5510 - accuracy: 0.8125 - val_loss: 0.9823 - val_accuracy: 0.6787
Epoch 11/100
1563/1563 [==============================] - 77s 49ms/step - loss: 0.4811 - accuracy: 0.8391 - val_loss: 0.9852 - val_accuracy: 0.6855
<keras.callbacks.History at 0x7f7e22f72f10>
```

Fig.4: Processing of the model in step 3

Figure 5

```
Epoch 1/100
1563/1563 [==============================] - 99s 62ms/step - loss: 1.7626 - accuracy: 0.3511 - val_loss: 1.4338 - val_accuracy: 0.4849
Epoch 2/100
1563/1563 [==============================] - 90s 58ms/step - loss: 1.4117 - accuracy: 0.4905 - val_loss: 1.2613 - val_accuracy: 0.5541
Epoch 3/100
1563/1563 [==============================] - 91s 58ms/step - loss: 1.2826 - accuracy: 0.5432 - val_loss: 1.1954 - val_accuracy: 0.5763
Epoch 4/100
1563/1563 [==============================] - 90s 58ms/step - loss: 1.1892 - accuracy: 0.5775 - val_loss: 1.1165 - val_accuracy: 0.6083
Epoch 5/100
1563/1563 [==============================] - 91s 58ms/step - loss: 1.1098 - accuracy: 0.6044 - val_loss: 1.0474 - val_accuracy: 0.6360
Epoch 6/100
1563/1563 [==============================] - 91s 59ms/step - loss: 1.0455 - accuracy: 0.6331 - val_loss: 1.0136 - val_accuracy: 0.6453
Epoch 7/100
1563/1563 [==============================] - 93s 59ms/step - loss: 0.9875 - accuracy: 0.6526 - val_loss: 0.9949 - val_accuracy: 0.6515
Epoch 8/100
1563/1563 [==============================] - 92s 59ms/step - loss: 0.9402 - accuracy: 0.6698 - val_loss: 0.9616 - val_accuracy: 0.6647
Epoch 9/100
1563/1563 [==============================] - 92s 59ms/step - loss: 0.8879 - accuracy: 0.6881 - val_loss: 0.9506 - val_accuracy: 0.6629
Epoch 10/100
1563/1563 [==============================] - 91s 58ms/step - loss: 0.8511 - accuracy: 0.7028 - val_loss: 0.9217 - val_accuracy: 0.6770
Epoch 11/100
1563/1563 [==============================] - 91s 58ms/step - loss: 0.8113 - accuracy: 0.7154 - val_loss: 0.8995 - val_accuracy: 0.6860
Epoch 12/100
1563/1563 [==============================] - 91s 58ms/step - loss: 0.7677 - accuracy: 0.7307 - val_loss: 0.8873 - val_accuracy: 0.6890
Epoch 13/100
1563/1563 [==============================] - 92s 59ms/step - loss: 0.7284 - accuracy: 0.7461 - val_loss: 0.8771 - val_accuracy: 0.6949
Epoch 14/100
1563/1563 [==============================] - 91s 58ms/step - loss: 0.6976 - accuracy: 0.7571 - val_loss: 0.8727 - val_accuracy: 0.6994
Epoch 15/100
1563/1563 [==============================] - 91s 58ms/step - loss: 0.6637 - accuracy: 0.7674 - val_loss: 0.8767 - val_accuracy: 0.6980
Epoch 16/100
1563/1563 [==============================] - 91s 58ms/step - loss: 0.6171 - accuracy: 0.7839 - val_loss: 0.8543 - val_accuracy: 0.7067
Epoch 17/100
1563/1563 [==============================] - 90s 58ms/step - loss: 0.5911 - accuracy: 0.7941 - val_loss: 0.8525 - val_accuracy: 0.7065
Epoch 18/100
1563/1563 [==============================] - 99s 64ms/step - loss: 0.5582 - accuracy: 0.8042 - val_loss: 0.8592 - val_accuracy: 0.7070
Epoch 19/100
1563/1563 [==============================] - 92s 59ms/step - loss: 0.5284 - accuracy: 0.8146 - val_loss: 0.8869 - val_accuracy: 0.7017
<keras.callbacks.History at 0x7fe0000f7150>
```

Fig.5: Processing of the model in step 4

REFERENCES

Colab.research.google.com. 2022. *Google Colaboratory*. [online] Available at: <https://colab.research.google.com/> [Accessed 8 May 2022].

Cs.toronto.edu. 2022. *CIFAR-10 and CIFAR-100 datasets*. [online] Available at: <http://www.cs.toronto.edu/~kriz/cifar.html> [Accessed 8 May 2022].

Docs.python.org. 2022. *3.10.4 Documentation*. [online] Available at: <https://docs.python.org/3/> [Accessed 8 May 2022].

General Data Protection Regulation (GDPR). 2022. *General Data Protection Regulation (GDPR) – Official Legal Text*. [online] Available at: <https://gdpr-info.eu/> [Accessed 8 May 2022].

Giusti, A., Ciresan, D., Masci, J., Gambardella, L. and Schmidhuber, J., 2013. Fast image scanning with deep max-pooling convolutional neural networks. *2013 IEEE International Conference on Image Processing*.

Gupta, A., 2021. *A Comprehensive Guide on Deep Learning Optimizers - Analytics Vidhya*. [online] Analytics Vidhya. Available at: <https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-on-deep-learning-optimizers/> [Accessed 8 May 2022].

Hansson, S., Belin, M. and Lundgren, B., 2021. Self-Driving Vehicles—an Ethical Overview. *Philosophy & Technology*, 34(4), pp.1383-1408.

Human Rights Watch. 2022. *Losing Humanity*. [online] Available at: <https://www.hrw.org/report/2012/11/19/losing-humanity/case-against-killer-robots> [Accessed 8 May 2022].

Krizhevsky, A., Sutskever, I. and Hinton, G., 2017. ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6), pp.84-90.

Matplotlib.org. 2022. *Matplotlib — Visualization with Python*. [online] Available at: <https://matplotlib.org/> [Accessed 8 May 2022].

National Institutes of Health (NIH). 2022. *National Institutes of Health (NIH)*. [online] Available at: <https://www.nih.gov/> [Accessed 8 May 2022].

Numpy.org. 2022. *NumPy Documentation*. [online] Available at: <https://numpy.org/doc/> [Accessed 8 May 2022].

Pandas.pydata.org. 2022. *pandas documentation — pandas 1.4.2 documentation*. [online] Available at: <https://pandas.pydata.org/pandas-docs/stable/> [Accessed 8 May 2022].

Pandiya, M., Dassani, S. and Mangalraj, P., 2020. Analysis of Deep Learning Architectures for Object Detection - A Critical Review. *2020 IEEE-HYDCON*.

Paperswithcode.com. 2022. *Papers with Code - CIFAR-10 Benchmark (Image Classification)*. [online] Available at: <https://paperswithcode.com/sota/image-classification-on-cifar-10> [Accessed 8 May 2022].

Roach, S. and Eckert, A., 2020. *Moral responsibility in twenty-first-century warfare*.

Smith, M. and Miller, S., 2021. The ethical application of biometric facial recognition technology. *AI & SOCIETY*, 37(1), pp.167-175.

Smith, R., Amaral, R. and Heywood, M., 2021. Evolving Simple Solutions to the CIFAR-10 Benchmark using Tangled Program Graphs. *2021 IEEE Congress on Evolutionary Computation (CEC)*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R., 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15, pp.1929-1958.

Team, K., 2022. *Keras documentation: Getting started with KerasTuner*. [online] Keras.io. Available at: <https://keras.io/guides/keras_tuner/getting_started/> [Accessed 8 May 2022].

TensorFlow. 2022. *TensorFlow*. [online] Available at: <https://www.tensorflow.org/> [Accessed 8 May 2022].

Touvron, H., Cord, M., Sablayrolles, A., Synnaeve, G. and Jegou, H., 2021. Going deeper with Image Transformers. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*.

Yıldırım, S., 2020. *L1 and L2 Regularization — Explained*. [online] Medium. Available at: <https://towardsdatascience.com/l1-and-l2-regularization-explained-874c3b03f668> [Accessed 8 May 2022].

Zhou, P., Feng, J., Ma, C., Xiong, C., HOI, S. and E, W., 2021. Towards Understanding Why Lookahead Generalizes Better Than SGD and Beyond. *35th Conference on Neural Information Processing Systems*.