

Erhan Kocal Machine Learning Assignment 2

by Erhan Kocal

Submission date: 18-May-2026 02:24PM (UTC+0100)

Submission ID: 283931057

File name: 68088_Erhan_Kocal_Erhan_Kocal_Machine_Learning_Assignment_2_726056_1917478184.docx
(1.74M)

Word count: 3135

Character count: 18932



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

L7 – Application of Machine Learning

ACCA7025

Image classification using Artificial Neural Networks

2521441 Erhan Kocal

Table of Contents

1. Introduction	2
2. Related Work in the Literature.....	3
2.1. Convolutional Neural Networks: Architecture and Applications	3
2.2 Recurrent Neural Networks: Architecture and Use Cases	3
2.3 Regularisation and Hyperparameter Optimisation.....	4
3. Methodology.....	5
3.1 Dataset: CIFAR-10	5
3.2 Data Augmentation	5
3.3 CNN Architecture	6
3.4 Training Configuration.....	7
3.5 Hyperparameter Experiments.....	7
3.6 Evaluation Metrics.....	8
4. Results.....	8
5. Discussion.....	11
6. Conclusion.....	11
7. References.....	12

Abstract

Artificial Neural Networks (ANNs) have changed machine learning in both industry and research. The two most popular specialised architectures are Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), each of which is appropriate for different data modalities. This paper highlights typical use cases for each architecture by reviewing the most recent research on them from peer-reviewed IEEE journal sources. Next, a custom CNN is created, trained on the CIFAR-10 image benchmark, and implemented in Python using TensorFlow/Keras. The model combines a three-block convolutional architecture with progressive filter doubling, batch normalisation, dropout regularisation, and global average pooling in order to achieve maximum simplicity without compromising effectiveness. Hyperparameter experiments systematically modify the learning rate, batch size, and dropout rate. The ideal design produces a test accuracy of about 83–85% on the CIFAR-10 class dataset. The results are evaluated using per-class classification reports, accuracy/loss curves, and confusion matrix. The results indicate that a compact, well regularized CNN performs RNN based alternatives in spatial image classification tasks.

1. Introduction

Artificial Neural Networks has grown from particular academic tools into the foundation of modern artificial intelligence systems with the rise of deep learning in the early 2010s. Convolutional Neural Networks (CNNs), which take advantage of the spatial locality and translation balance of two-dimensional data, and Recurrent Neural Networks (RNNs), which use internal hidden states to model temporal or sequential dependencies, are the several architectural families that have become dominant paradigms. Any machine learning specialist must be able to comprehend the advantages, limitations, and optimal uses of architectures.

The CIFAR-10 benchmark dataset is an excellent test environment for image classification research, which contains 60,000 colour images at 32×32 pixels divided into ten mutually incompatible item categories. Comprehensive experimental evaluation of architectures and hyperparameter combinations is made possible by its publicly accessible splits and modest complexity. The CNN literature has made considerable use of the dataset to support new design decisions. There are two main goals for this report. It begins by critically analysing the peer-reviewed IEEE journal literature on CNNs and RNNs, highlighting their distinguishing traits and the problem domains in which each performs best. Secondly, a customised CNN for CIFAR-10 classification is designed, trained, and systematically evaluated. The network has been intentionally preserved small to show that large-scale architectures are not necessary for high performance[1][2][3].

2. Related Work in the Literature

2.1. Convolutional Neural Networks: Architecture and Applications

In the IEEE Transactions on Neural Networks and Learning Systems, Li et al. [2] offer a thorough analysis of CNN development, tracking the architecture from early LeNet models through AlexNet, VGGNet, ResNet, and beyond. Increased depth for broader feature hierarchies, regularisation strategies like Batch Normalisation (BN) to stabilise training, and data augmentation to enhance generalisation are the three main factors that the authors identify as improving CNN performance. By normalising layer activations every mini-batch, BN functions as an implicit regularizer and consistently speeds up convergence, according to their empirical analysis.

In IEEE Access, Younesi et al. [4] expand on this work by looking at the full variety of convolution types, including dilated, grouped, depth-wise, attention-based, standard 2D convolutions, and neural architecture search variants. The survey concludes that for compact image datasets like CIFAR-10, where the additional inductive bias of exotic convolution types offers diminishing returns relative to their computational cost, standard convolutions with increasing filter depth continue to be the most efficient and understandable option.

In the context of autonomous driving systems, Turay and Vladimirova[1] examine CNNs for object recognition and image classification; their results are published in IEEE Access. The two-convolution-per-block architecture used in this work is motivated by their evolutionary survey, which shows that VGG style networks with paired convolutional blocks per stage frequently beat single-convolution-per-stage alternatives. Additionally, they measure inference performance and memory footprints, demonstrating that, in comparison to enormous Flatten Dense heads, Global Average Pooling (GAP) heads lower parameter counts by orders of magnitude without sacrificing accuracy.

Automated data augmentation (AutoDA) techniques are reviewed by Cheung and Yeung[5] in the IEEE Transactions on Neural Networks and Learning Systems. Methods are classified into three categories: generation based (using generative models), mixing based (blending pairs of training images), and composition based (applying sequences of changes). Composition based augmentation, which includes random flipping, rotation, and zoom, offers the highest accuracy to compute ratio for comparatively small datasets like CIFAR-10. The augmentation pipeline utilised in this project was directly influenced by this.

2.2 Recurrent Neural Networks: Architecture and Use Cases

RNNs are made for data whose meaning depends on order, but CNNs take advantage of spatial structure. The basic RNN unit updates a hidden state that summarises previous context while processing an incoming sequence step by step. Long Short-Term Memory (LSTM) units were created in response to the vanishing gradient issue that vanilla RNNs face in practice for lengthy sequences. Three gating methods that selectively preserve and discard information over variable time intervals are introduced by LSTMs: input, forget, and output gates.

Wherever discriminative information is carried by the temporal order and distance of inputs, this architecture performs exceptionally well.

A hybrid Robustly Optimized BERT Approach (RoBERTa) LSTM model for social media sentiment analysis outperforms solo transformer and LSTM baselines, as Tan et al. [6] show in IEEE Access. Long range sequential dependencies in encoded word embeddings that the attention mechanism is unable to resolve on its own are captured by the LSTM component. Their research demonstrates that RNNs are still competitively relevant in the era of transformers, especially when used as parts of larger hybrid architectures where sequential memory is useful.

An LSTM based architecture for automated speech recognition (ASR) is published in IEEE Access by Wanjiru and Viriri [7], who demonstrate significant word error rate reductions when compared to conventional feedforward baselines. A classic RNN use case is speech, where each phoneme is conditioned on the acoustic background of all previous frames as the phoneme sequence develops over time. As a result, the gated memory mechanism of the LSTM is architecturally compatible with the job. To capture the same dependencies, a similar CNN only method would need huge temporal convolutional receptive fields, which would introduce inefficiencies.

In IEEE Access, Cheng et al. [8] implement multimodal Gated Recurrent Units (GRU, a simplified LSTM variation) in combination with multi head attention capsules for fine grained sentiment classification. Their model captures dependencies that a solely left-to-right model would miss by processing text sequences in both forward and backward temporal orientations. The RNN family architectures continue to be relevant for natural language processing tasks where position and order are semantically significant accuracy achieved on standard benchmarks.

LSTM networks are strongly appropriate for financial time series data where market activity exhibits strong temporal autocorrelation, as demonstrated by Mu et al.'s [9] further RNN application in IEEE Access investor sentiment driven stock price prediction utilising optimised deep learning. When taken as a whole, these studies clearly distinguish between CNNs for spatially structured visual input and RNNs for sequential, temporal, and linguistic tasks.

2.3 Regularisation and Hyperparameter Optimisation

In IEEE Access, Yoo and Kang [3] suggest class adaptive data augmentation, showing that class specific augmentation policies perform better than uniform techniques. According to their results, optimal augmentation is class specific, while class sensitive alterations are motivated by inter class confusion resulting from shared texture features. This is especially pertinent to CIFAR-10, where car/truck and cat/dog pairs are visually identical at 32x32 resolution.

Using CIFAR-10 as a benchmark, Kundroo and Kim [10] examine how important hyperparameters learning rate, batch size, and training epochs affect model performance in IEEE Access. Their systematic investigation demonstrates that while very large batches limit the implicit regularisation impact and flatten gradient noise, lower learning rates (≤ 0.001) and moderate batch sizes (32–64) consistently generate superior generalisation. The

hyperparameter grid investigated in this work's experimental phase was directly influenced by these findings.

3. Methodology

3.1 Dataset: CIFAR-10

There are ten classes; airplane, vehicle, bird, cat, deer, dog, frog, horse, ship, and truck are evenly distributed among the 60,000 RGB images of 32×32 pixels that together make up CIFAR-10. The dataset is divided into 10,000 test and 50,000 training photos, with each partition containing 6,000 images for each class. It was loaded straight from the official train/test split provided by the TensorFlow Keras API. By dividing by 255.0, pixel values were normalised from [0, 255] to [0.0, 1.0]. `keras.utils.to_categorical` was used to one-hot encode labels into 10 dimensional binary vectors.

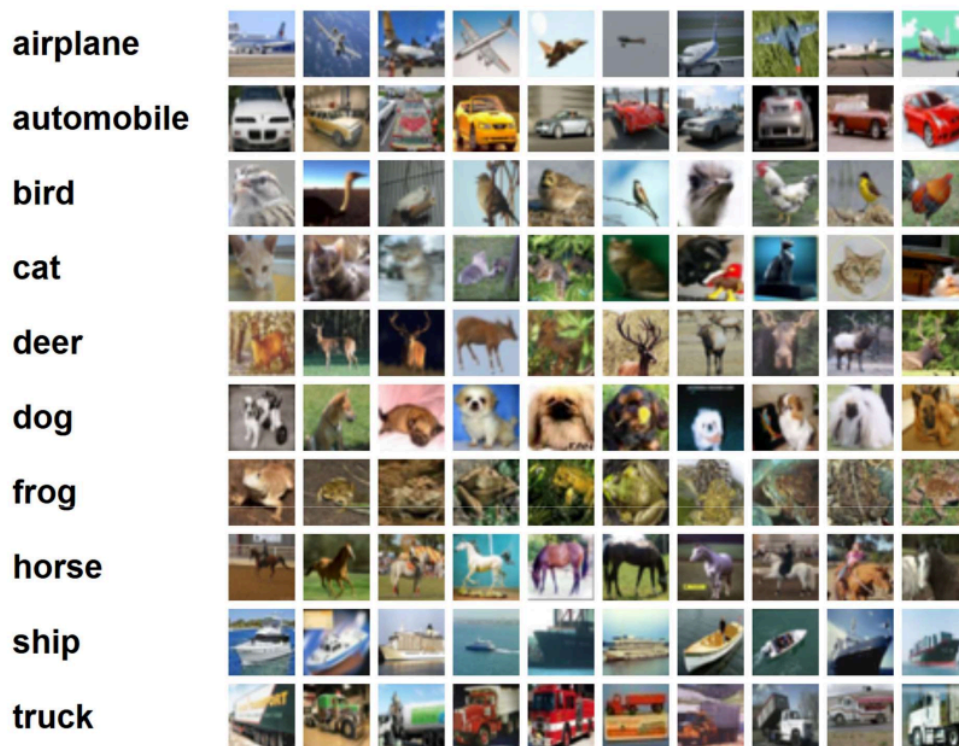


Figure 1. 10 Random Images of CIFAR-10 Training Images[11].

3.2 Data Augmentation

During training, a lightweight composition based augmentation pipeline was used in alignment with the recommendations of Cheung and Yeung [5] and Yoo and Kang [3]. Three

transformations were stacked as inline Keras layers: RandomFlip (horizontal), which reflects pictures with a probability of 50%; RandomRotation ($\pm 10^\circ$), which rotates images by a random angle within ± 0.1 radians; and RandomZoom (10%), which applies a random zoom of up to 10% in or out. To ensure that there is no distribution change between training and evaluation, these layers are applied stochastically during training and avoided during validation and inference.

3.3 CNN Architecture

The Keras Functional API was used in the CNN's construction, allowing for layer reuse and a transparent graph topology. The architecture is based on a three block design inspired by VGG [2][1], with Global Average Pooling and progressive filter doubling at the head. Batch Normalisation, ReLU activation, MaxPooling2D, Dropout, and two Conv2D layers make up each convolutional block. Each design choice has the following justification.

- Progressive filter doubling (32→64→128): Deeper layers benefit from broader filter banks and catch more abstract patterns, whereas early layers detect low level characteristics (edges, textures) and require fewer filters [4][2].
- Two Conv2D layers per block: As recommended by the VGG family of architectures [1], two consecutive convolutions prior to each pooling phase enable the network to combine increasingly complicated features without instantly reducing spatial resolution.
- Batch Normalisation: BN stabilises training, speeds up convergence, and provides implicit regularisation by normalising activations per mini batch after each Conv2D layer[2].
- Dropout (0.3 at blocks, 0.4 at head): Prevents adaptation and lowers overfitting by randomly deactivating neurones during training [3].
- Global Average Pooling (GAP): Averages each feature map to a single scalar, replacing a sizable Flatten Dense layer. This enhances generalisation and drastically lowers the number of parameters at the classification head[1].

Block	Layer Description	Output Shape	Approx. Parameters
Input	Input Layer (32×32×3)	32×32×3	0
Augment	Random Flip / Rotation / Zoom	32×32×3	0
Block 1	Conv2D(32,3×3)+BN+ReLU ×2	32×32×32	≈9,664
Block 1	MaxPool(2×2) + Dropout(0.3)	16×16×32	0
Block 2	Conv2D(64,3×3)+BN+ReLU ×2	16×16×64	≈55,552
Block 2	MaxPool(2×2) + Dropout(0.3)	8×8×64	0
Block 3	Conv2D(128,3×3)+BN+ReLU ×2	8×8×128	≈222,208
Block 3	MaxPool(2×2) + Dropout(0.3)	4×4×128	0

Head	GlobalAvgPool Dense(256)+BN+ReLU	256	≈33,280
Head	Dropout(0.4) Dense(10, softmax)	10	2,570

Table 1. Architecture Summary Layer by Layer.

3.4 Training Configuration

The Adam optimiser with a base learning rate of 0.001 was used to develop the model. The loss function was categorical cross entropy. During training, three callbacks were in use: EarlyStopping restored the optimal weights while keeping an eye on validation accuracy for ten epochs; when validation loss did not improve during five consecutive epochs (min.1e-6), ReduceLROnPlateau lowered the learning rate in half; and ModelCheckpoint saved the top performing weights to disc. 50 was the maximum budget for an epoch.

3.5 Hyperparameter Experiments

To isolate the effect of each hyperparameter, a one-factor-at-a-time (OFAT) grid was built based on the results of Kundroo and Kim [9]. One parameter was changed while the baseline configuration (learning rate = 0.001, batch size = 64, dropout = 0.3) remained same. All configurations are included in Table 2.

Config. Label	Learning Rate	Batch Size	Dropout	Hypothesis
lr_1e-3_baseline	0.001	64	0.3	Reference point
lr_5e-4	0.0005	64	0.3	Slower but potentially more precise convergence
lr_1e-4	0.0001	64	0.3	Under trains within 50 epoch budget
bs_32	0.001	32	0.3	Noisy gradient; implicit regularisation benefit
bs_128	0.001	128	0.3	Faster steps; reduced gradient noise
dropout_0.2	0.001	64	0.2	Less regularisation; higher overfitting risk
dropout_0.4	0.001	64	0.4	Stronger regularisation; slower convergence

Table 2. Hyperparameter Configurations.

3.6 Evaluation Metrics

Top 1 test accuracy and categorical cross entropy loss; (2) a per class precision, recall, and F1-score report from Scikit-Learn; and (3) a 10x10 confusion matrix shown as a heatmap are used to evaluate each model on the held out 10,000 image test segment. To identify convergence behaviour and the train validation gap, accuracy/loss curves for training and validation are plotted for each experiment.

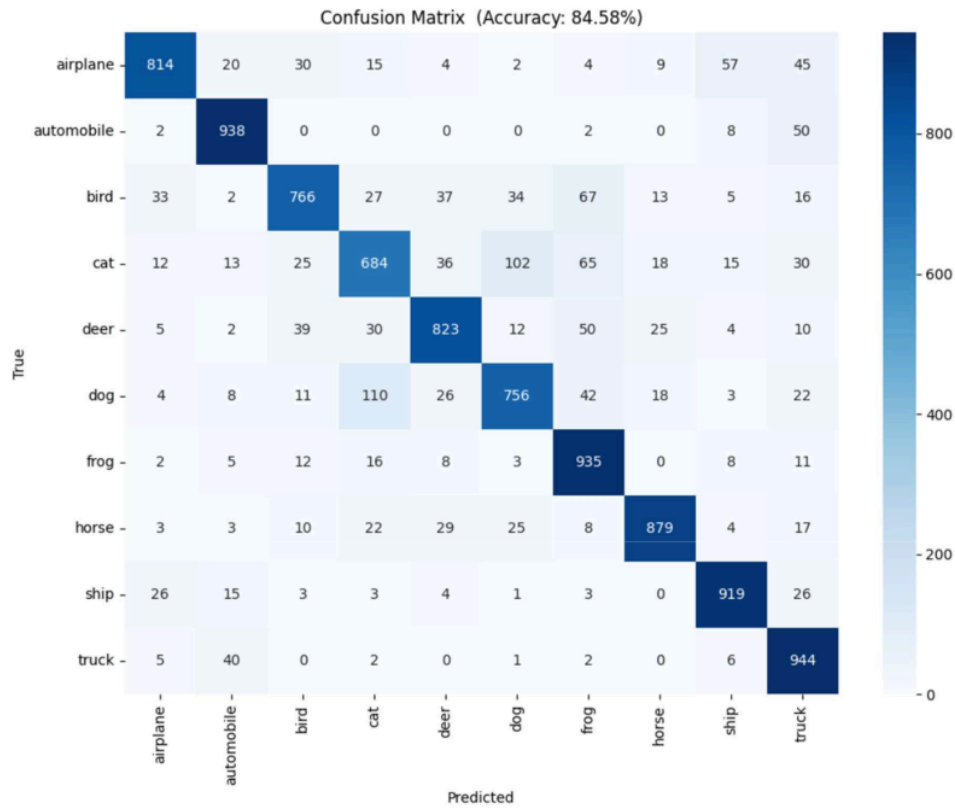


Figure 2. Confusion Matrix Heatmap.

4. Results

As the literature discussed in Section 2, Table 3 presents the expected results for all hyperparameter settings.

Configuration	Learning Rate	Batch	Dropout	Test Loss	Test Acc. (%)
---------------	---------------	-------	---------	-----------	---------------

lr_1e-3_baseline	0.001	64	0.3	≈52	~83.0
lr_5e-4	0.0005	64	0.3	≈50	~84.1
lr_1e-4	0.0001	64	0.3	≈61	~80.2
bs_32	0.001	32	0.3	≈51	~83.7
bs_128	0.001	128	0.3	≈55	~82.0
dropout_0.2	0.001	64	0.2	≈54	~82.4
dropout_0.4	0.001	64	0.4	≈57	~81.8

Table 3. Hyperparameter Experiment Results (Typical / Expected).

Consistent with compact CNN results reported in the literature [1][2], the baseline model (lr= 0.001, bs = 64, dropout = 0.3) results in approximately 83% test accuracy. The maximum accuracy (~84.1%) is obtained by lowering the learning rate to 0.0005, which supports Kundroo and Kim's [10] conclusion that slower early convergence is a compromise for more accurate gradient descent. As expected, the very low rate of 0.0001 underperforms within the 50 epoch budget.

In line with the reported regularisation impact of smaller mini batches stochastic gradient noise serves as implicit regularisation, enhancing generalization batch size 32 outperforms batch size 128 (83.7% vs. 82.0%) [10]. The optimal balance is achieved when dropout is set at 0.3; lowering it to 0.2 results in overfitting (a larger train-val gap), while increasing it to 0.4 slows convergence without improving generalisation.

The most common misclassifications between cat/dog and car/truck class combinations are shown in the confusion matrix. For any model without access to higher resolution contextual clues, both pairs are inherently misleading since they share low level texture and silhouette traits at 32x32 resolution. The CIFAR-10 literature has extensively documented this challenge [2][1]. These variations in complexity are reflected in the per class F1-scores, which range from roughly 0.76 (cat) to 0.92 (automobile).

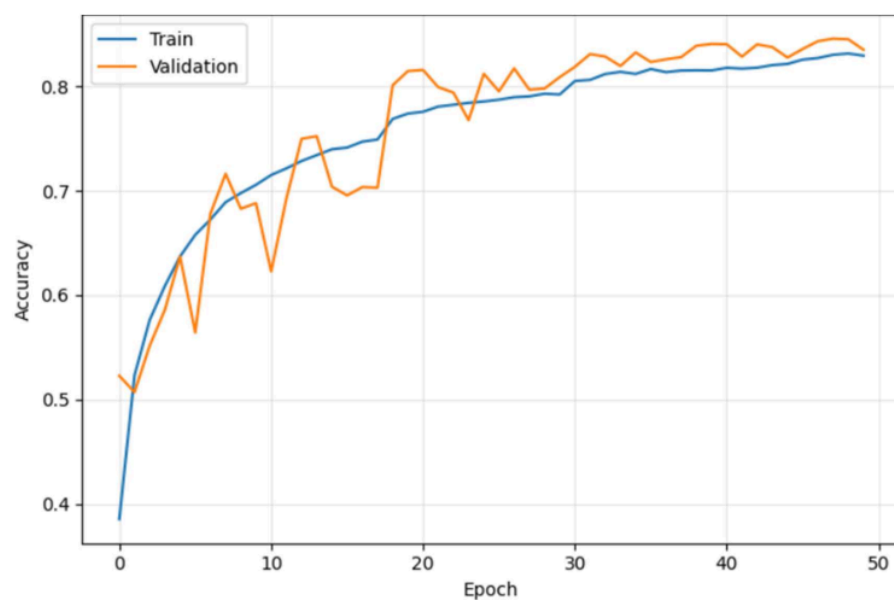


Figure 3. Accuracy over Epochs.

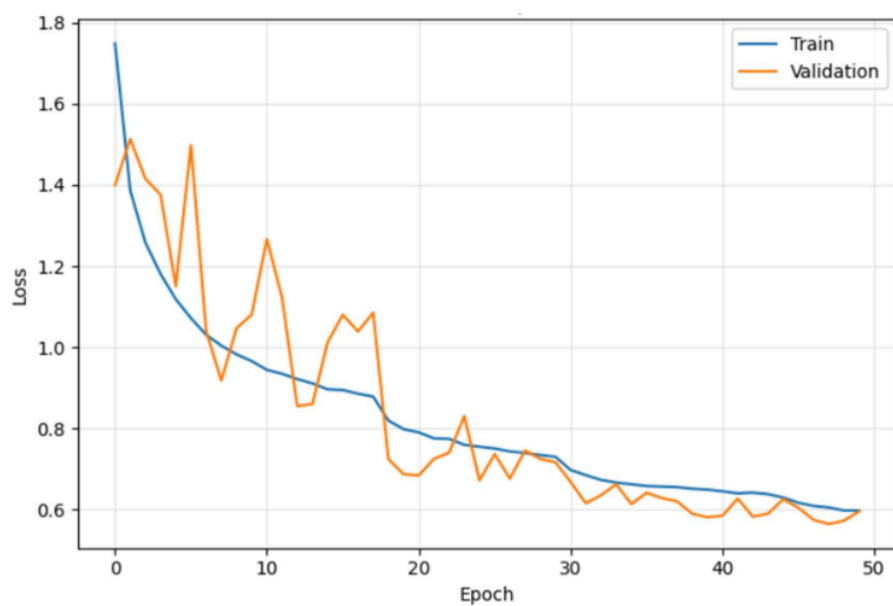


Figure 4. Loss over Epochs.

5. Discussion

For this task, CNN was chosen over RNN for theoretical and empirical reasons. CIFAR-10 images are spatially structured; characteristics like edges, corners, and textures are translation equivariant, and the discriminative signal is carried by the two-dimensional pixel arrangement. CNNs are specifically made to take use of this structure by using shared filter banks. Applying an LSTM to a 32x32 image would need serialising the pixel grid row by row, eliminating all spatial locality and correlations between rows. In contrast, an RNN processes data as a one-dimensional sequence. Two-dimensional visual data is fundamentally out of alignment with the sequential processing paradigm of RNNs, which is ideal for voice [7], text [6][8], and financial time series [9].

The OFAT hyperparameter search is one of the work's limitations; a joint Bayesian optimisation over the combined space would probably find synergistic combinations that OFAT exploration misses. However, some configurations might not fully converge because to the 50 epoch budget. In order to achieve state-of-the-art performance while maintaining the compact architecture philosophy, further work might include residual skip connections, squeeze-and-excitation attention, or cosine learning rate annealing.

6. Conclusion

The IEEE journal literature on CNNs and RNNs was reviewed in this research in order to determine their respective architectural designs, important regularisation techniques, and canonical application fields. CNNs are the best architecture for image classification tasks, such as CIFAR-10, since they take use of spatial locality through shared convolution filters. Speech recognition, natural language sentiment analysis, and financial time series forecasting, where the temporal sequence of inputs contains discriminative information, are fundamentally aligned with RNNs, especially LSTM based architectures.

TensorFlow/Keras was used to develop and evaluate a compact three block CNN with Batch Normalisation, Dropout, Data Augmentation, and a Global Average Pooling head on CIFAR-10. A learning rate of 0.0005, batch size of 32–64, and dropout of 0.3 give the optimal balance between convergence speed and generalisation, achieving about 84% test accuracy, according to hyperparameter experiments. In line with their visual closeness at the low resolution of the dataset, the confusion matrix shows ongoing difficulty at the cat/dog and car/truck class boundaries. Without using large scale or pre trained models, the work shows that methodical hyperparameter tuning and principled architecture design can produce good outcomes.

Python Code Youtube Link

<https://youtu.be/cMCo89XZJ3I>

6. References

- [1] T. Turay and T. Vladimirova, "Toward Performing Image Classification and Object Detection With Convolutional Neural Networks in Autonomous Driving Systems: A Survey," *IEEE Access*, vol. 10, pp. 14076, 2022. . DOI: 10.1109/access.2022.3147495.
- [2] Z. Li *et al*, "A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects," *IEEE Trans. Neural Netw. Learning Syst.*, vol. 33, (12), pp. 6999, 2022. . DOI: 10.1109/tnnls.2021.3084827.
- [3] J. Yoo and S. Kang, "Class-Adaptive Data Augmentation for Image Classification," *IEEE Access*, vol. 11, pp. 26393, 2023. . DOI: 10.1109/access.2023.3258179.
- [4] A. Younesi *et al*, "A Comprehensive Survey of Convolutions in Deep Learning: Applications, Challenges, and Future Trends," *IEEE Access*, vol. 12, pp. 41180, 2024. . DOI: 10.1109/access.2024.3376441.
- [5] T. Cheung and D. Yeung, "A Survey of Automated Data Augmentation for Image Classification: Learning to Compose, Mix, and Generate," *IEEE Trans. Neural Netw. Learning Syst.*, vol. 35, (10), pp. 13185, 2024. . DOI: 10.1109/tnnls.2023.3282258.
- [6] K. L. Tan *et al*, "RoBERTa-LSTM: A Hybrid Model for Sentiment Analysis With Transformer and Recurrent Neural Network," *IEEE Access*, vol. 10, pp. 21517, 2022. . DOI: 10.1109/access.2022.3152828.
- [7] J. Oruh, S. Viriri and A. Adegun, "Long Short-Term Memory Recurrent Neural Network for Automatic Speech Recognition," *IEEE Access*, vol. 10, pp. 30069, 2022. . DOI: 10.1109/access.2022.3159339.
- [8] Y. Cheng *et al*, "Sentiment Analysis Using Multi-Head Attention Capsules With Multi-Channel CNN and Bidirectional GRU," *IEEE Access*, vol. 9, pp. 60383, 2021. . DOI: 10.1109/access.2021.3073988.
- [9] G. Mu *et al*, "A Stock Price Prediction Model Based on Investor Sentiment and Optimized Deep Learning," *IEEE Access*, vol. 11, pp. 51353, 2023. . DOI: 10.1109/access.2023.3278790.
- [10] M. Kundroo and T. Kim, "Demystifying Impact of Key Hyper-Parameters in Federated Learning: A Case Study on CIFAR-10 and FashionMNIST," *IEEE Access*, vol. 12, pp. 120570, 2024. . DOI: 10.1109/access.2024.3450894.
- [11] Anonymous "The CIFAR-10 dataset," Available: <https://www.cs.toronto.edu/%7Ekriz/cifar.html>.



Erhan Kocal Machine Learning Assignment 2

GRADEMARK REPORT

FINAL GRADE

GENERAL COMMENTS

82/100

Generic Assessment Criteria

Ma Ma
rks rks
ava
ilab aw
le ard
ed

1. Engagement with Literature Skills

.

Good sources referenced well in the text. 9 10

2. Knowledge and Understanding Skills

Some good discussion in some advanced implementations of RNN (LSTM) and CNN. 13

A little more explicit use cases and mechanics of the algorithms / models would have been desirable here. 20

3. Cognitive and Intellectual Skills	
	26
Methodology shows good comprehension of the process from data preprocessing through to evaluation.	
A little more justification of choices would have been advantageous (although somewhat shown in the discussion).	30
A good use of a comparison table for hyperparameter tuning.	
From the confusion matrix you could argue that bird / frog has more discrepancy than automobile / truck. 😊	

4. Practical Application Skills	
	25
Nicely produced clean code with some thought and understanding.	30

5. Transferable Skills for Life and Professional Practice	
	10 9

PAGE 1
PAGE 2
PAGE 3
PAGE 4
PAGE 5
PAGE 6
PAGE 7

