Classification of the 102-Category *Flowers* Dataset with Convolutional Deep Neural Networks

Examination Candidate #Y3898772

Submitted in partial fulfilment of the requirements of the Intelligent Systems: Machine Learning and Optimisation *module assignment at the University of York in the 2023/24 academic year.*

***Abstract*—The classification of data into discrete categories is an ancient problem, recently made accessible on extremely large datasets due to substantial advances in hardware capability; one such advancement is the introduction of graphics processing units (GPUs) in machine learning (ML) applications, particularly for the training and evaluation of deep neural networks (DNNs). This report defines and evaluates such a DNN for the classification of the 102-category *Flowers* dataset[1] from the Visual Geometry Group at the University of Oxford. For this purpose, a convolutional deep neural network (CDNN) was constructed, capable of attaining 41% test-accuracy on *Flowers*.**

I. Introduction

CLASSICAL classification-based computer vision problems are concerned with the training of DNNs to categorise images by a fixed set of 'labels'. Often, this involves recognising images from a wide range of highly disjointed categories, and significant work and experimentation has been conducted in this area [1]. In contrast to many existing datasets, the 102-category *Flowers* set consists of a large number of relatively similar categories, each representing a genus of flower that is commonly found on the British Isles [2]. Given the similarity between labels and the small volume of training data[2], *Flowers* presents a notably difficult problem.

Previous experiments utilising support vector machines (SVMs) equipped with weighted linear combinations of numerous kernels can attain impressive test-accuracies of up to 72.8% [2], where the optimum weights can be systematically learned [3]. Subsequently developed models based on *Inception-v3*, trained with the 14-million-sample *ImageNet*, have achieved 94% test-accuracy on *Flowers* [4].

As the demand on ML increases, the complexity of the datasets on which DNNs are expected to work will invariably rise, thus giving justification to the importance of research into the systematic learning of large-category datasets, often with restricted volumes of training samples. The DNN presented henceforth uses a combination of convolutions, batch normalisation, the rectified linear activation function, cross-entropy-softmax loss, and learning-rate scheduling to attain a test-accuracy of 41% with no pre-trained information.

II. Method

The designed CDNN follows a typical convolutional process: given a 16-image batch of three-channel (RGB) images each representable as tensors over $\mathbb{Q}_+$, the CDNN passes the batch through a sequence of hidden layers:

1) Using a 3×3 kernel, perform two-dimensional convolution over the three-channel input to produce a 32-channel output. Iterate over all batch images;
2) Execute batch-normalisation on the 32-channel tensor;
3) Activate the batch-normalised tensor with the rectified linear unit activation function; and
4) Perform two-dimensional max-pooling on the output with a 2-stride 2×2 kernel.
5) Repeat stages 1 to 4 with increasing numbers of input-output convolution channels: $32\to64$ and $64\to128$.
6) Collapse and recursively linearise the tensor to yield a 102-component probability tensor for each batch image.

Various aspects of the CDNN are now explored further.

A. Two-Dimensional Convolution

On a single C_{in}-channel image of dimensions W_{in}-by-H_{in}, the two-dimensional convolution operator Conv2d is such that

$$\mathsf{Conv2d}\colon \mathbb{Q}_+^{(C_{\text{in}},H_{\text{in}},W_{\text{in}})}\to\mathbb{Q}_+^{(C_{\text{out}},H_{\text{out}},W_{\text{out}})}, \tag{1}$$

where, for all tensors $\mathcal{A}\in\mathbb{Q}_+^{(C_{\text{in}},H_{\text{in}},W_{\text{in}})}$ containing channels $\mathcal{A}_1,\ldots,\mathcal{A}_{C_{\text{in}}}$,

$$\mathcal{A}\mapsto\sum_{i=1}^{C_{\text{in}}}[\mathsf{Weight}\,(C_{\text{out}},i)\star\mathcal{A}_i]+\mathsf{Bias}(C_{\text{out}}). \tag{2}$$

(The convolutional cross-correlation operator is denoted by $\star$, and Weight and Bias select the suitable channel-wise weight and global bias respectively.) All C_{out} two-dimensional components of the codomain of Conv2d, $\mathbb{Q}_+^{(C_{\text{out}},H_{\text{out}},W_{\text{out}})}$, are *feature maps* of the convolution. Convolution is an important stage of the feature-extraction process, whereby unimportant features are abstracted away from the attention of the learnable parameters of the CDNN. Kernel sizes of 3×3 were selected empirically, and based on the relevant literature [5].

B. Batch Normalisation (BN)

BN, typically performed on the convolved tensor, causes faster convergence of the CDNN[3]. Normalisation as a pre-processing technique is known to substantially improve the performance of a DNN, and BN extends the processing to each hidden layer of the network. (3) represents the transformation

Manuscript prepared with LaTeX and IEEEtran on 8th April 2026.

All submitted work, unless stated otherwise, is the sole creation of Examination Candidate #Y3898772.

[1] https://www.robots.ox.ac.uk/~vgg/data/flowers/102/

[2] Training: 1020 images; Validation: 1020 images; Testing: 6149 images.

[3] The usefulness of BN was originally thought to be explained by its supposed tenancy to reduce the DNN's *internal covariate shift* (ICS) [6]. In a groundbreaking 2019 study, it was shown that BN improves the β-*Lipschitzness* of the derivative of the cost function, hence smoothening the cost function, enabling larger learning rates during gradient descent [7].

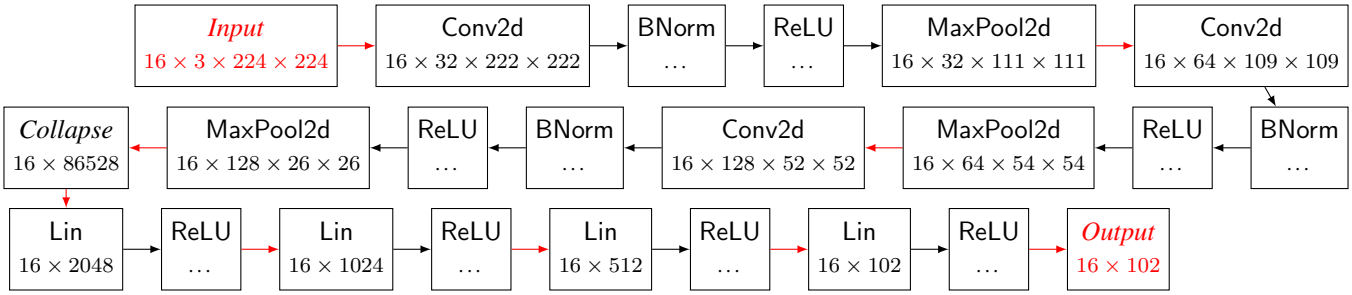

Fig. 1. The network architecture of the CDNN, where the top row indicates the function of the current layer, and the bottom row indicates the shape of the tensor *following* the transformation. With a batch size of 16, a four-way batch tensor of RGB 224×224 images is translated to a set of probability vectors for each image, over each of the 102 categories. An ellipsis (...) indicates that the tensor dimensions are unchanged by the corresponding transformation. Black and red arrows denote feed-forwards *within* and *across* DNN layers, respectively.

of the convoluted batch-tensor $\mathcal{B}$, where $\mu_{\mathcal{B}}$ and $\sigma_{\mathcal{B}}$ denote the mean and standard deviation of $\mathbb{Q}$-components of $\mathcal{B}$ respectively[4].

$$\mathsf{BNorm}(\mathcal{B}) = \frac{\gamma\,(\mathcal{B} - \mu_{\mathcal{B}})}{\sigma_{\mathcal{B}}} + \beta \tag{3}$$

Given the known regularising properties of BN [9], no further techniques were employed to discourage over-fitting.

C. Activation with the Rectified Linear Unit

The rectified linear unit (ReLU) activator is the function $\mathsf{ReLU}\colon \mathbb{R} \to \{0, x\}$ such that $x \mapsto \max(0, x)$ for all $x \in \mathbb{R}$ [10], which is known to improve the learning of DNNs due to its thresholding properties [11].

D. Max-Pooling

Pooling uses some operator to reduce information in fixed regions to an aggregate value [12], hence lowering the complexity of features on which the DNN operates. *Max-pooling* was used in the CDNN, whereby the pooling operator aggregates a region by selecting its maximal value [13]. A 2×2 region was empirically chosen.

E. Linearisation

Linearisation, denoted by $\mathsf{Lin}(\mathcal{A})$, applies a linear transform using the learned parameters to the given tensor $\mathcal{A}$. The four-stage linearisation process, each consisting of the 'activated composition' $\mathsf{ReLU} \circ \mathsf{Lin}$, reduce the large number of collapsed features into the 102-probability vector for each batch image.

F. Cross-Entropy Loss with Softmax

The cross-entropy loss composed with softmax, henceforth denoted SCELoss, was selected as the suitable objective-loss function given its established performance in classification-based computer vision tasks. The machine learning-relevant mapping is given by [14].

III. Network Architecture

Figure 1 shows a diagrammatic representation of the CDNN.

[4]The γ and β shift parameters can be learned to offset any misrepresentation as a result of the linear transformation. This is especially useful when BNorm is composed with the activation function; see [8].

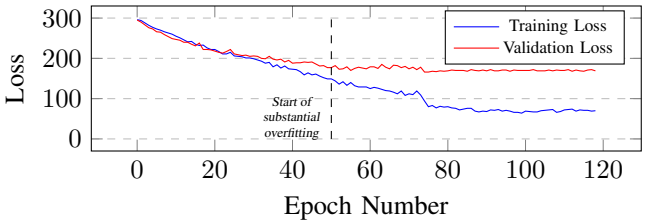

Fig. 2. The losses on training data and validation data over training epochs

IV. Results and Evaluation

The CDNN used an initial LR of 0.001, with SCELoss being used to inform a *learning rate (LR) scheduler*, such that the learning rate of the stochastic gradient descent (SGD) would reduce by a factor of $1/5$ in the event of a non-decreasing loss over multiple epochs [15]. A batch size of 16 was chosen based on the recommended literature and comparable experiments across different sets of hyperparameters [16]. SGD was utilised over alternative commonly used optimisers due to the superior generalisability of SGD [17], which is a property of particular importance considering the skewed split sizes of *Flowers*.

A single *NVIDIA A40* GPU was used in conjunction with an *AMD EPYC 7742* 64-core CPU[5]. Training occurred over 31 minutes and iterated over 118 epochs. Figure 2 displays the losses over epochs, where each epoch performed a self-test with the entire validation set, and the early-stopping mechanism intended to detect over-fitting was disabled. The eventual accuracy on the test data (evaluated after a full 118-epoch training cycle) was 41%.

V. Conclusion and Further Work

Considering the lack of pre-trained weights, coupled with the fast convergence over a small training split, this CDNN is deemed to have achieved reasonable, although certainly not maximal, test-accuracy. The modelled CDNN approximates initial experiments by the VGG team [2], albeit using a differing DNN architecture. Within the space of convolutional DNNs, the architecture described by Figure 1 is supported by sensible design principles, although additional work should be undertaken to determine the potential utility of adding hidden layers, or employing more sophisticated LR schedulers [18].

[5]This hardware is accessible via SSH at `csgpu13.cs.york.ac.uk`.

REFERENCES

[1] L. Chen, S. Li, Q. Bai, J. Yang, S. Jiang, and Y. Miao, "Review of image classification algorithms based on convolutional neural networks," *Remote Sensing*, vol. 13, no. 22, 2021. [Online]. Available: https://doi.org/10.3390/rs13224712

[2] M.-E. Nilsback and A. Zisserman, "Automated flower classification over a large number of classes," in *Indian Conference on Computer Vision, Graphics and Image Processing*, 2008.

[3] M. Varma and D. Ray, "Learning the discriminative power-invariance trade-off," in *IEEE 11th International Conference on Computer Vision*. IEEE, 2007, pp. 1–8. [Online]. Available: https://doi.org/10.1109/ICCV.2007.4408875

[4] X. Xia, C. Xu, and B. Nan, "Inception-v3 for flower classification," in *2017 2nd International Conference on Image, Vision and Computing (ICIVC)*. IEEE, 2017, pp. 783–787. [Online]. Available: https://doi.org/10.1109/ICIVC.2017.7984661

[5] J. Wang, J. Lin, and Z. Wang, "Efficient convolution architectures for convolutional neural network," in *2016 8th International Conference on Wireless Communications & Signal Processing (WCSP)*. IEEE, 2016, pp. 1–5. [Online]. Available: https://doi.org/10.1109/WCSP.2016.7752726

[6] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, arXiv pre-print. [Online]. Available: https://arxiv.org/abs/1502.03167

[7] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry, "How does batch normalization help optimization?" 2019, arXiv pre-print. [Online]. Available: https://arxiv.org/abs/1805.11604

[8] T. van Laarhoven, "l_2 regularization versus batch and weight normalization," 2017, arXiv pre-print. [Online]. Available: https://arxiv.org/abs/1706.05350

[9] P. Luo, X. Wang, W. Shao, and Z. Peng, "Towards understanding regularization in batch normalization," 2019, arXiv pre-print. [Online]. Available: http://arxiv.org/abs/1809.00846

[10] R. H. R. Hahnloser, R. Sarpeshkar, M. A. Mahowald, R. J. Douglas, and H. S. Seung, "Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit," *Nature*, vol. 405, pp. 947–951, 2000.

[11] A. F. Agarap, "Deep learning using rectified linear units (relu)," 2019, arXiv pre-print. [Online]. Available: https://arxiv.org/abs/1803.08375

[12] M. Sun, Z. Song, X. Jiang, J. Pan, and Y. Pang, "Learning pooling for convolutional neural network," *Neurocomputing*, vol. 224, pp. 96–104, 2017. [Online]. Available: https://doi.org/10.1016/j.neucom.2016.10.049

[13] J. Nagi, F. Ducatelle, G. A. Di Caro, D. Cireşan, U. Meier, A. Giusti, F. Nagi, J. Schmidhuber, and L. M. Gambardella, "Max-pooling convolutional neural networks for vision-based hand gesture recognition," in *2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*, 2011, pp. 342–347. [Online]. Available: https://doi.org/10.1109/ICSIPA.2011.6144164

[14] J. Cao, Z. Su, L. Yu, D. Chang, X. Li, and Z. Ma, "Softmax cross entropy loss with unbiased decision boundary for image classification," in *2018 Chinese Automation Congress (CAC)*, 2018, pp. 2028–2032. [Online]. Available: https://doi.org/10.1109/CAC.2018.8623242

[15] J. Konar, P. Khandelwal, and R. Tripathi, "Comparison of various learning rate scheduling techniques on convolutional neural network," in *2020 IEEE International Students' Conference on Electrical,Electronics and Computer Science (SCEECS)*, 2020, pp. 1–5. [Online]. Available: http://doi.org/10.1109/SCEECS48394.2020.94

[16] I. Kandel and M. Castelli, "The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset," *ICT Express*, vol. 6, no. 4, pp. 312–315, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405959519303455

[17] M. Hardt, B. Recht, and Y. Singer, "Train faster, generalize better: Stability of stochastic gradient descent," 2016, arXiv pre-print. [Online]. Available: https://arxiv.org/abs/1509.01240

[18] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts," 2017, arXiv pre-print. [Online]. Available: https://arxiv.org/abs/1608.03983

Manuscript prepared with

LaTeX

TikZ

IEEEtran