Hyper Parameters Selection for Image Classification in Convolutional Neural Networks

Sajid Nazir
School of Engineering and Built Environment
Glasgow Caledonian University
Glasgow, G4 0BA, UK
sajid.nazir@gcu.ac.uk

Shushma Patel, Dilip Patel
Department of Engineering
London South Bank University
London, SE1 0AA, UK
{shushma, dilip}@lsbu.ac.uk

***Abstract*—Cognitive image processing is made possible by the availability of faster and cheaper memories, increased processing power of multicore processors and the explosive growth in visual data generation. Object classification remains an active research area with rapid advancements. The recent advances in deep neural networks are providing better than expected results for image classification. The process involves training models with large labelled datasets to learn the underlying features from various image classes to support cognitive inferences on the test data. By training on large annotated image datasets contextual and semantic information can be automatically extracted from the images. An important determinant of better results is the choice of hyper parameters which is difficult to get right. In this paper we empirically investigate the effect of selected hyper parameters of a convolutional neural network on CIFAR-10 dataset and provide results to demonstrate their effect and importance for image classification.**

Keywords—deep learning; cognitive image processing; neural networks; hyper parameters; convolution

I. INTRODUCTION

Vision is the predominant sense amongst humans. About 80% of the brain is used for visual data processing [1]. The increased use of Internet of Things (IoT) [2], social, web and mobile applications have resulted in an increased emphasis on visual communications. According to Facebook, in 2016 more than 265 million new monthly active users were added. Similarly, 300 hours of videos are uploaded to YouTube, each hour. The images acquired through these modalities are generating large data sets that can be usefully mined to generate feature rich, enhanced images.

The vast volumes of images and videos are beyond the cognitive ability of humans to manage [1]. The exponential increase in images generated and stored requires intelligent processing algorithms to be developed to classify the content. The traditional pixel-based image processing paradigm needs to pave the way for the cognitive image processing paradigm [1].

Cognitive image processing is deeply rooted in mathematical concepts and exists at the convergence of computer vision, machine learning and statistics. The objective is for a machine to mimic human like cognitive visual

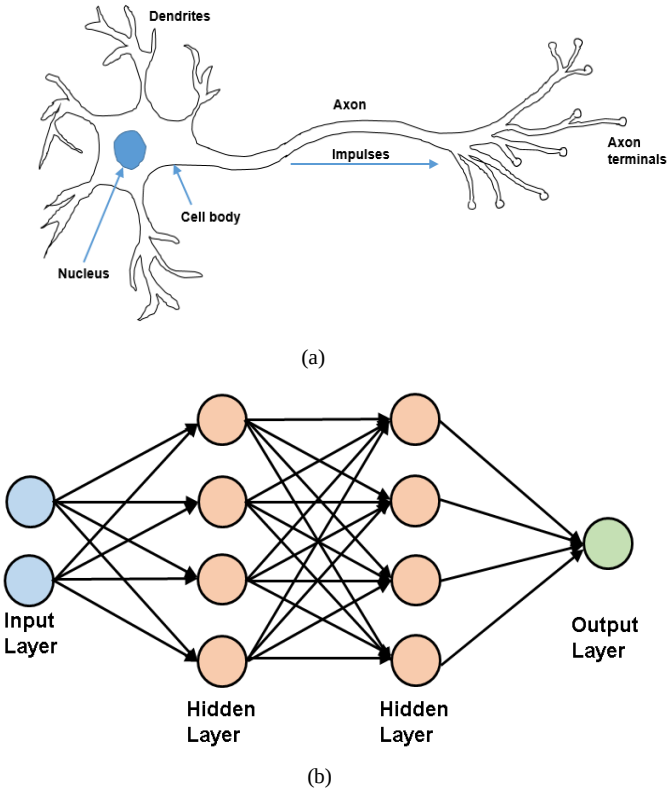

Fig. 1. (a) Biological Neuron (b) Artitifical Neural Network (ANN).

abilities [3]. Deep learning algorithms are complex to develop, train and evaluate.

A neural net [4] with 60 million parameters and 650,000 neurons took a long time to train on ImageNet [5], in order to classify 1.2 million images. Cognitive image processing techniques have been applied to a vast array of application areas. Biological neurons (Fig 1a) have inspired the Artificial Neural Networks (ANN) (Fig1b) interconnections and activations [6].

Cognitive vision focuses on developing interoperable, robust, resilient and adaptable systems, with the ability to learn, evaluate alternatives, analyse, interpret, apply to new contexts and communicate with other systems [7]. The increased research interest in neural networks is due to the promising

results obtained for ImageNet competitions [4]. The two leading types are the Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN). Cognitive vision can support the development of systems that are more robust, flexible and smarter. [8] has demonstrated the application of cognitive vision in many areas (surveillance, industrial inspection, stock photo databases, industrial robotics, film, TV and entertainment, life sciences and aerospace). Cognitive image processing has been applied to Standard Platform League (SPL) game scenarios for goal and ball detection [9]. The process of describing the cognitive environment to the model is a complex problem. To overcome the limitation, cognitive models were applied for object detection in two games, 3D driver and Mars rover [10]. CNNs have been used for large image datasets [4], [11]. CNN have also been applied for video classification [12]. The application of deep learning for different medical image modalities is provided in [13].

An important aspect of the neural networks performance is the hyper parameters or the model's parameters, and their effect on the recognition results. This information is critical to design and develop efficient models. This paper describes the CNN's hyper parameters in detail and investigates the effects of changes to selected hyper parameters to arrive at better choices.

The effect of some hyper parameters for activation, such as Rectified Linear unit (ReLU) and Sigmoid were investigated in [14]. However, our focus is on investigating a large selection of important hyper parameters for learning rates activation layer, momentum, and batch size while at the same time making use of regularization (dropout) [4], [15] and batch normalization [16]. Therefore a simple architecture has been chosen that lets us easily investigate the effect of parameter change on improving image classification results. We do not optimize the hyper parameters through grid search [17] but investigate one parameter at a time for better results.

The paper makes the following contributions:

- describing the CNN architecture and the importance of hyper parameters
- investigating the effect of important hyper parameters on the model's accuracy to help researchers to make informed choices.

Section II outlines the relevant key concepts underpinning our approach. The use of CNN is provided in Section III. The system hyper parameters are described in Section IV. The system model and results are provided in Section V and VI, with the conclusion in Section VII.

II. BACKGROUND

A. Advancements in Computer Vision

Computer vision technologies for object recognition have rapidly evolved and new methodologies and techniques with improved results have been proposed [4], [5]. The current re-emergence of neural networks for computer vision applications is attributed to [5], [18] and there have been improvements. Although the biological systems, particularly vision process are not fully understood, the current focus yields promising results. ImageNet Large Scale Recognition Challenge (ILSVRC) [19] has been running since 2009 that provides a common platform for comparing computer vision algorithms for object detection and classification [19].

B. Neural Network and why they work

Artificial Neural Networks (ANNs) are modelled on the human nervous system [3]. ANNs originated in neuroscience in 1943 [20]. ANNs need computational power and large volumes of data to be trained before they can be successfully used. It can learn from any mathematical relation between the input and output [3]. For supervised learning, we need to train the ANN by providing a large number of labelled true and false examples, before good results can be obtained.

A deep neural network utilizes many intermediate or hidden layers. The depth increases interconnections and complexity of nodes. The initial layers work on low level features, lines, circles etc., whereas the deeper layers work to higher or complex features, until the whole image gets recognized [3]. Such systems can perform at the same or better levels than humans [3].

C. Advanced Neural Architectures

Of particular interest to us, and which also provide good results are Convolutional Neural Networks (CNN), and Recurrent Neural Network (RNN). CNNs are more suitable for spatial data and take a fixed size input and generate fixed size outputs whereas RNNs can handle different input/output sizes and use time-series data. Although RNNs have been successfully applied to hand writing recognition, some variations were also applied to images. However CNNs, due to their very nature, are more suitable for object recognition exploiting the spatial dimensions of height and width.

The basic ANNs have been refined and combined to yield many different algorithms each inspired by a different philosophy. More complex models have been proposed by combining both CNN and RNN [21], [22], [23], [24].

III. CNN MODEL AND LAYERS

CNNs are better suitable for image classification than any other neural network. They have fewer connections and parameters compared to standard feedforward networks and are thus easier to train [4]. These were brought to prominence through LeNet [18] which used CNNs for the first time for character recognition. Improved CNN architectures have provided state-of-the-art object recognition results on large datasets [21]. CNN is a type of feed-forward artificial neural network which requires minimal preprocessing. The connectivity pattern of CNN neurons is inspired by the animal visual cortex [6]. The use of Graphical Processing Unit (GPU) has enabled the training and testing of CNNs for large datasets [21]. The first GPU implementation of CNN [4] to win the ImageNet competition was in 2012 and its winning top-5 error rate was 15.3% whereas the second best was 26.2%. A CNN comprises of many layers that are used to create an architecture for a sequential processing pipeline.

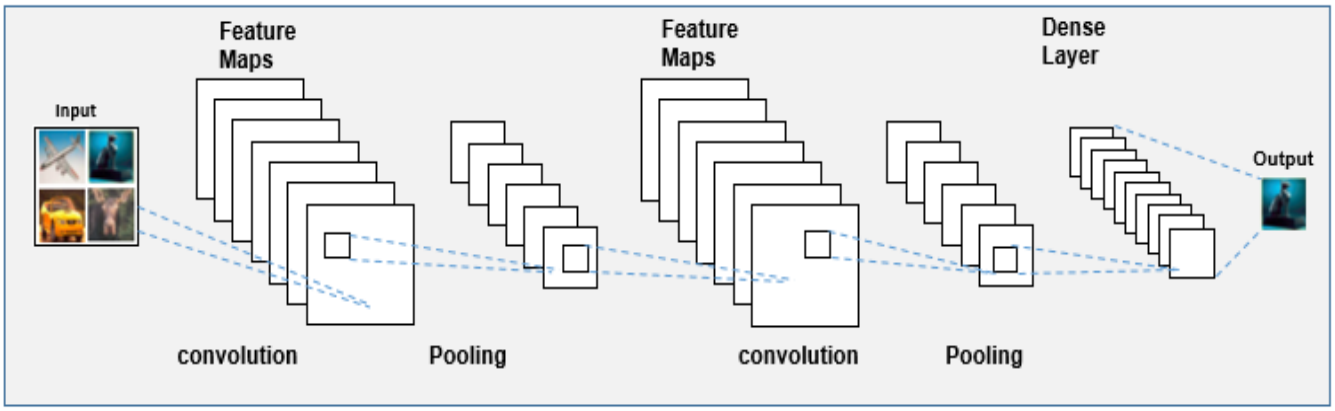

Fig. 2. A Generalized Convolutional Neural Network (CNN) architecture.

A. *Generalized CNN Architecture*

Although a neural network can learn the image features autonomously, the model's parameters need to be defined for better results. A generalized CNN architecture is shown in Fig 2. We provide details of the layer types and hyper parameters needed to identify the depth of the network, and other parameters. The layers between the input and output layers are termed as hidden layers and a CNN can be designed to intersperse the different types of layers. The depth of the CNN is equivalent to its hidden layers.

B. *Convolution Layer*

The convolution layers apply a filtering process to the image to extract the relevant features. In 2D convolution, a 2D mask or filter is moved over the image, starting at the top left and ending at the bottom right. At each position the sum of the products of the image and mask parameters is obtained and replaces the image pixel value at the mask center. The important parameters at this stage are the filter size, dimension and stride.

C. *Activation Layer*

After the convolutional layer, an activation layer is added to introduce a non-linearity to the otherwise linear operations in convolution. The common non-linear functions are ReLU, Sigmoid and Tanh. Comparisons between the different activation functions found ReLU to perform better overall [14]. Recently however, some variants of ReLU have also been proposed, such as Leaky ReLU (LeakyReLU) [25] and Parametric RELU (PReLU) [26].

D. *Pooling layers*

The purpose of the pooling layer is to reduce the spatial dimensions of the image through down sampling. This results in significant reduction in the number of trainable parameters. A representative value is chosen, for example, to represent four (2x2 grid) values, resulting in halving the image resolution. The pooling layers can be interspersed in a neural network for the required effect. The choice of value can be governed by selecting the maximum, average pooling or L2-norm from the selected subset (normally 2x2 block in the image), however, max pooling is generally the preferred approach.

Combining and generalizing of the pooling functions is proposed in [27] but it slightly increased the computational complexity in training and required more model parameters.

E. *Batch Normalization Layer*

The training of a neural network model is slowed down, as the changes to layers' inputs are effected by previous layer parameters changes [16]. Batch normalization proposed by [16] overcomes this by normalizing each training mini-batch. They also propose dispensing with the Dropout layer as batch normalization aids regularization. Their results show improvements for ImageNet classification performing better than a human.

F. *Dropout Layer*

Dropout [30] is added to help prevent overfitting by randomly setting some of the input units to 0 for each update during training.

G. *Dense Layers*

These are used at the end of a classification task to combine the results of the previous layers. These are fully connected layers.

IV. HYPER PARAMETERS

CNN are complex networks and comprise of many hyper parameters. Hyper parameters are basically parameters about parameters, describing the model parameter values. The right choice of hyper parameters has a profound effect on the result of a network.

It is generally considered difficult to design an optimum model on the first attempt and it needs some trial and experimentation to find optimum values. Hyper parameter optimization or tuning is a process applied to tune the model by tweaking the parameters for the best results. The model may be

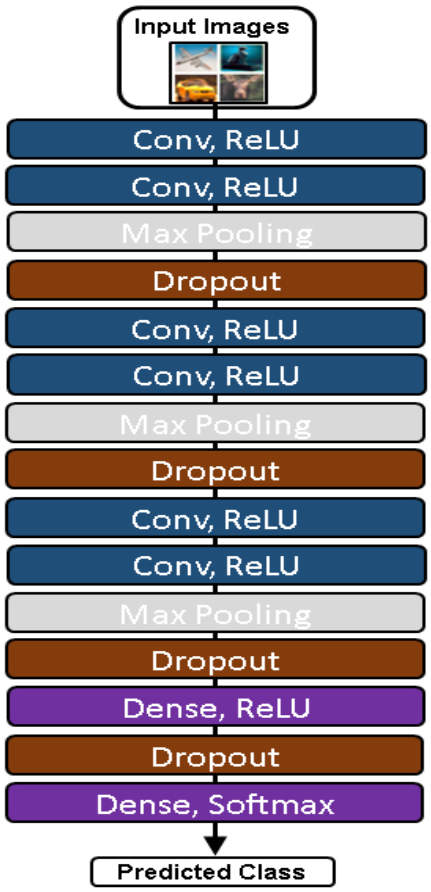

Fig. 3. System architecture.

TABLE I. SELECTED HYPER PARAMETERS AND THEIR VALUE RANGES

Hyper parameter	*Value*
Optimizers	SGD, RMSProp, Adam
Activation	ReLU, LeakyReLU, PReLU
Learning Rate	0.00001, 0.0001, 0.001, 0.01, 0.1
Batch size	32, 64, 128, 256, 512
DropOut	0.25, 0.5
Regularization	Batch Normalization, Dropout
Pooling	Max (2,2)
Convolution Layers	6

susceptible to degradation by small changes to its parameters. For example, removing one layer from the five convolutional layer model degraded the performance [4]. Thus a lot of mutual interdependencies might exist amongst the optimum hyper parameters.

Grid search based methods worked well in earlier machine learning models with limited parameters. Grid based methods performed an exhaustive search over the entire parameter space without regard to the importance or effect of a single parameter. Nowadays it is generally agreed in the research community that the random search for optimum parameters provides better results compared to grid search. This could initially be in larger steps across the selected range of values and then in finer steps closer to values yielding improved results. A brief summary of the important hyper parameters is provided in the following sub-sections.

A. Architecture-Type and number of hidden layers

The layers between the input and output layers are termed as hidden layers (Fig 1b and 2). The number of hidden layers define the depth of the network. The depth of the proposed layers has been consistently increasing and in general perform better than a shallow network. However alternate architectures with less depth have also been proposed [28].

B. Optimizers

The selected optimizers for investigation are RMSProp, Stochastic Gradient Descent (SGD) and Adam. These work well with a batch size of 32 to 512 [29]. The authors [29] have studied using SGD with larger batch sizes for deep learning applications. An important and common parameter in all these optimizers is the learning rate. The value of the learning rate is between 0 and 1. Instead of a linear scale, it is appropriate to use a logarithmic scale.

C. Activation Function

The activation functions have been compared in [14] and ReLU provided the best results. More complex variants of ReLU have been proposed recently, that is, LeakyReLU [25] and PReLU [26]. The implementations are avaialble in Keras as "Advanced Activations". For multi-class classification Softmax is used as the last layer.

D. Dropout Regularization

Training the model prepares it to perform well on the unseen data during testing [6]. However, a complex model can learn the training data perfectly but may not generalize and perform poorly on unseen examples, a phenomenon termed as overfitting. Regularization is used to avoid overfitting. One regularization technique is Dropout [15]. We use 0.25 Dropout layers to avoid overfitting. Data Augmentation is another possibility where existing data is augmented by generating new images from existing data, that is, an image generated by modifying an existing one through simple operations such as flip and rotation.

E. Convolution Layer

There are many parameters that can be changed. The important ones here are the number of kernels applied to each layer, the height and width of each convolutional kernel, zero padding and stride. Without zero padding the size of the convolved image will reduce. Stride means the amount of

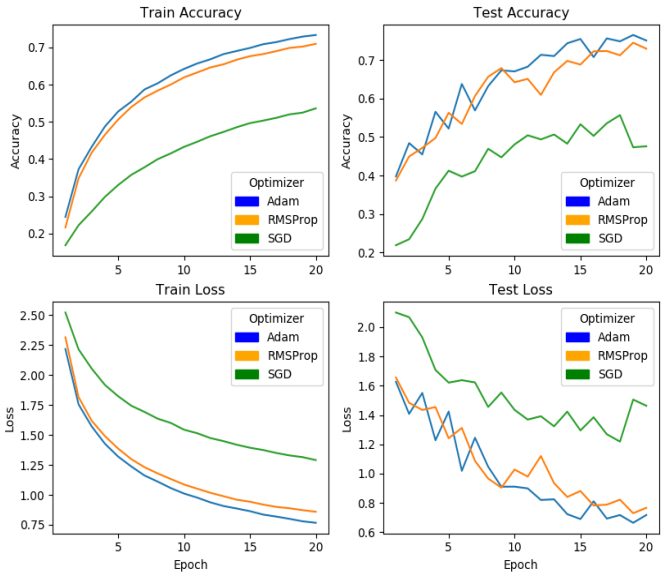

Fig. 4. Accuracy and Loss results with change of Optimizers.

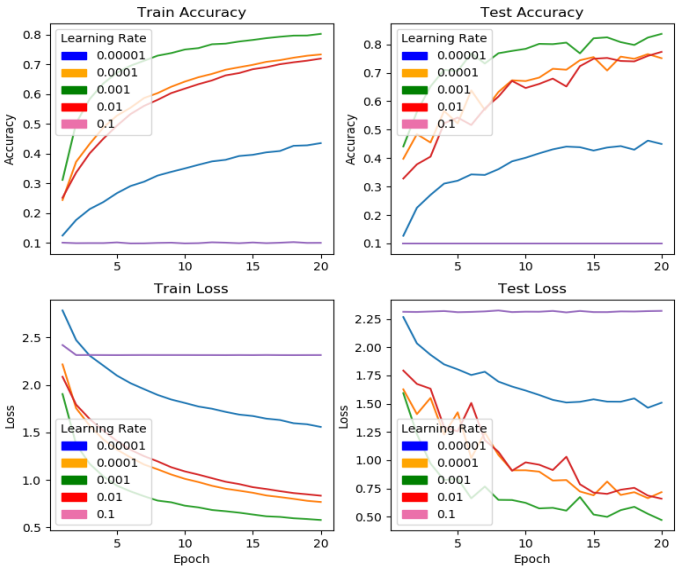

Fig. 5. Accuracy and Loss results with change of Learning Rate.

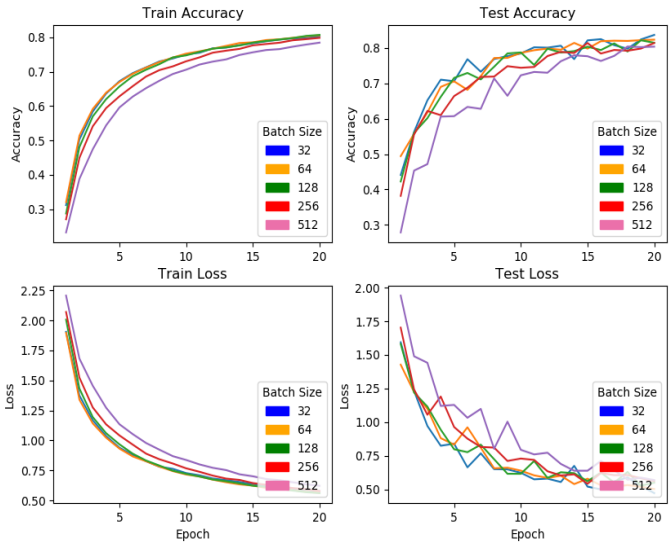

Fig. 6. Accuracy and Loss results with change of Batch Size.

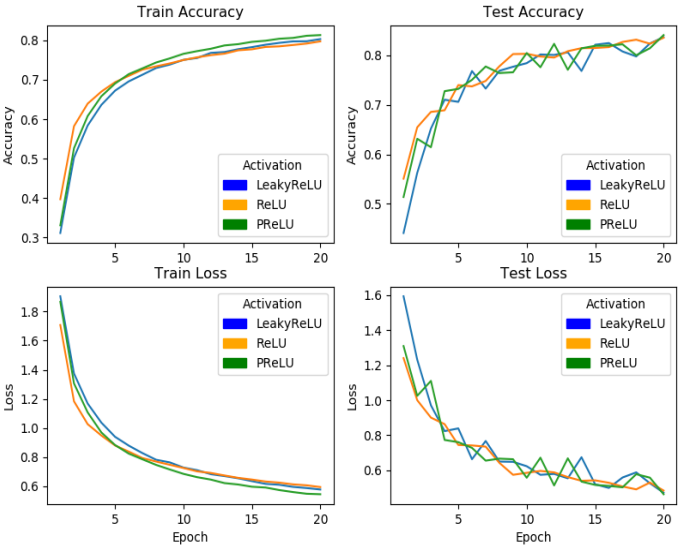

Fig. 7. Accuracy and Loss results with Change of Activation Units.

movement of the kernel after calculating one value. If it is more than one then again the convolved image will reduce in size.

F. Dimensions of pooling matrices in Pooling Layers

Generally a 2x2 size for the pooling is used that down samples the image into half. A larger pooling matrix size would increase the down sampling rate.

G. Number of Epochs and batch size

An epoch consists of one pass of data over the batches. In general a higher value for epoch will result in better results. During each epoch the data is processed in batches.

V. System Model

Keras [30] was used with Tensorflow [31] (as a backend), a popular library for image classification. Keras is the perfect choice for investigating choice of hyper parameters as it allows for a quick change of parameters to observe the corresponding effect. The Amazon Web Services (AWS) p2.xlarge instance type having 61 GB RAM, 4 vCPUs and a single GPU was used to run the experiments.

As explained above, although grid search could also have been used, we chose to do the manual hyper parameter tuning by changing one hyper parameter at a time and observing its effect. Due to the limited availability of computing resources, we kept the model as fixed (Fig 3) and investigated the effect

of the change of the selected hyper parameters on the results. Only one parameter at a time was altered and used the others with the corresponding optimal values. This provides a fair comparison across the choices for a hyper parameter and provides insights into the effect each hyper parameter has on improving the model.

An epoch size of 20 and Batch Normalization with Dropout were used.

A. *Image Datasets*

We use the most common benchmark dataset in machine learning classification problems, CIFAR-10 [4][26], which has 60,000 images of size 32x32 pixels across 10 categories (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck). It is widely used for deep learning research on image processing.

B. *System Architecture*

The architecture of neural network is shown in Fig 3. It comprises of 6 convolutional layers, each followed by an Activation layer. We chose the architecture to be both simple and yet support the proposed hyper parameter tuning. Important parameters with the values to be investigated are listed in Table I. We use both Batch Normalization and Dropout for regularization [16]. In addition Max pooling was used. The model provides promising results and can be effectively used to see the outcome of hyper parameter changes.

C. *Selected Hyper Parameters*

The width and depth of the model was kept constant. Batch normalization and Dropout regularization was used to speed up the training process. The following hyper parameters for investigation based on their importance, were selected:

- Optimizer (RMSprop, SGD, Adam)
- Learning rate (0.00001, 0.0001, 0.001, 0.01, 0.1)
- Number of epochs and batch size (batch 32, 64, 128, 256, 512)
- Activation (ReLU, LeakyReLU, PReLU)

VI. RESULTS

The model was executed for tuning each of the hyper parameters, as described in Section V (C). Continuing in the sequence listed, we explored and selected the best hyper parameter value and used it for subsequent explorations. For each selected parameter value we report the loss and accuracy for both training and test data.

A. *Optimizers*

The results are shown in Fig 4. The Learning rate as 0.001, batch size as 32, and ReLU were used. Adam and RMSProp gave better results compared to SGD for both accuracy and loss.

B. *Learning Rate*

The results for the effect of the learning rate are plotted in Fig 5 for the Adam optimizer and batch size of 32, which had better results overall compared to RMSProp (not shown). The best results for accuracy and loss were obtained for a learning rate of 0.001. The extreme values at 0.1 and 0.00001 gave poor performance.

C. *Number of Epochs & Batch Size*

The improvements of results with a higher epoch is understood, however, the effect of batch size has not been explored. We kept the number of fixed epochs as 20 but varied the batch size from 32 to 256. The results are shown in Fig 6. The performance at batch size of 32, 64 and 128 was similar. However, for size 32 we had a better loss performance.

D. *Activation Unit*

We compared ReLU with the newly added advanced activations Leaky ReLU (LeakyReLU), and Parametric ReLU (PReLU). The batch size was 32, and learning rate was 0.001. The results are shown in Fig 7. The performance of the ReLU and LeakyReLU is similar to PReLU having a relatively better performance both in terms of loss and accuracy.

VII. CONCLUSION

A lot of advancements have taken place in the application of cognitive image processing concepts as artificial neural networks making it a mainstream research area. The results of current state-of-the-art CNN have been very promising. The tuning of a model architecture is driven by intuition and experimentation resulting in the optimal values of the hyper parameters. We have investigated the effect of different model hyper parameters on the problem of image classification for a popular image dataset and have provided the effect on the results. The findings would be helpful for researchers in selecting an initial optimal set of hyper parameters that can quickly be tuned to yield better results. Advancements are expected in autonomic image classification with further understanding of biological visual cognition processes. Automatic image cognition will thus become more accurate, fast, encompassing semantic and contextual information, thus truly opening innovative areas for cognitive image processing.

REFERENCES

[1] E. Diamant, "Cognitive image processing: the time is right to recognize that the world does not rest more on turtles and elephants," arXiv:1411.0054v1 [cs.CY].

[2] D. Bubley, "IoT & Realtime Communications," IEEE newsletter, Mar 2016, https://iot.ieee.org/newsletter/march-2016/iot-realtime-communications.html

[3] IBM Report, Romeo Kienzler, "Developing cognitive IoT solutions for anomalydetection by using deep learning, Part 1: Introducing deep learning and long-short term memory networks:Detecting anomalies in IoT time-series data by using deep learning," 16 May 2017.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," In Advances in Neural Information Processing Systems (NIPS), vol. 1, pp. 1097–1105, 2012.

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. FeiFei, "Imagenet: A large-scale hierarchical image database," In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 248–255, 2009.

[6] Deep Learning Tutorial, Release 0.1, LISA Lab, University of Montreal, Sep 2015. http://deeplearning.net/tutorial/deeplearning.pdf.

[7] P. Auer, A. Billard, H. Bischof, I. Bloch, et al., "A Research Roadmap of Cognitive Vision," ECVision: The European Research Network for Cognitive Computer Vision Systems, IST Project IST-2001-35454, Aug 2005, www.ecvision.org.

[8] P. Courtney, P. Böttcher, "ECVision White paper on industrial applications of cognitive vision", European Research Network for Cognitive Computer Vision Systems, 2002.

[9] G.E.H. Ras, Cognitive Image Processing for Humanoid Soccer in Dynamic Environments, Bachelor thesis, Maastricht University.

[10] K. D. Shah, "Image Processing for Cognitive Dynamic Gaming Environments", MSc thesis, 2003.

[11] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, "Going deeper with convolutions," CoRR, abs/1409.4842, 2014.

[12] J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, G. Toderici, "Beyond Short Snippets: Deep Networks for Video Classification" arXiv:1503.08909v2.

[13] D. Shen, G. Wu, Heung-II Suk, "Deep Learning in Medical Image Analysis", Annu Rev Biomed Eng, Jun 2017, 19, pp. 221-248.

[14] A. Koutsoukas, K. J. Monaghan, X. Li, J. Huan, "Deep‑learning: investigating deep neural networks hyper‑parameters and comparison of performance to shallow methods for modeling bioactivity data", Journal of Cheminform (2017) 9:42.

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

[16] S. Ioffe, C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," *CoRR*, 2015.

[17] J. Bergstra, J., D. Yamins, D. D. Cox, "Making a Science of Model Search: Hyperparameter Optimizationin Hundreds of Dimensions for Vision Architectures," 30th International Conference on Machine Learning (ICML 2013), Atlanta, Gerorgia, June 16 – 21, 2013. In JMLR Workshop and Conference Proceedings 28 (1), pp. 115-123.

[18] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E.Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. Neural Computation, 1(4), pp. 541-551, 1989.

[19] Large Scale Visual Recognition Challenge (ILSVRC) http://www.image-net.org/challenges/LSVRC/

[20] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. Bulletin of Mathematical Biophysics, 1943.

[21] M. Liang, X. Hu, "Recurrent Convolutional Neural Network for Object Recognition", CVPR 2015.

[22] R. Socher, B. Huval, B. Bhat, C. D. Manning, A. Y. Ng, "Convolutional-Recursive Deep Learning for 3D Object Classification" NIPS'12 Proceedings of the 25th International Conference on Neural Information Processing Systems, vol. 1, pp. 656-664.

[23] [23] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, W. Xu, "CNN-RNN: A Unified Framework for Multi-label Image Classification", arXiv:1604.04573 [cs.CV]

[24] J. Wu, Y. Yu, C. Huang, K. Yu, "Deep multiple instance learning for image classification and auto-annotation," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, 2015, pp. 3460-3469.

[25] A. L. Maas, A. Y. Hannam, A. Y. Ng, "Rectier Nonlinearities Improve Neural Network Acoustic Models," Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP vol. 28.

[26] K. He, X. Zhang, S. Ren, J. Sun, "Delving Deep into Rectifiers:Surpassing Human-Level Performance on ImageNet Classification" 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, 2015, pp. 1026-1034.

[27] C. Y. Lee, P. W. Gallagher, Z. Tu, "Generalizing Pooling Functions in Convolutional Neural Networks: Mixed, Gated, and Tree" in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PP, no. 99, pp. 1-1.

[28] S. H. Hasanpour, M. H. Rouhani, M. Fayyaz, M. Sabokrou, "Lets keep it simple, Using simple architectures to outperform deeper and more complex architectures," arXiv:1608.06037.

[29] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, P. T. P. Tang, "On Large-batch Training for Deep Learning: Generalization Gap and Sharp Minima" ICLR 2017.

[30] Keras https://keras.io/

[31] M. Abadi, P. Barham, J. Chen, et al. "TensorFlow: a system for large-scale machine learning," OSDI'16, Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation, pp. 265-283.