Natural Image Processing

Jose Daniel Mendez - 001032715

Abstract

A research on the performance of Convolutional Neural Networks (CNNs) in Natural Image Processing, will be presented,emphasizing the effects of hyperparameter tuning in the prediction capability of the algorithm, based on a model composed by: 4 fully connected 2D convolutional layers,followed by 2 fully connected dense layers.Tests were performed using a Dataset of 16000 images from 8 different classes.

1. Introduction

Image classification is a widely studied field in Artificial Intelligence space,it could be described as the act of giving machines the ability to process optical information, in other words vision, where the goal is to enable machines to perceive the world as humans do.A CNN was chosen as the subject for this research, because their capability, to segment images and assign weights to this segments, based on perceived importance, thus being able to differentiate between this characteristics, while requiring less pre-processing, compared to other classication algoritms as Artificial Neural Networks (ANN) which require more parameters.

1.1. Convolutional Neural Networks

The concepts behind CNNs were pioneered by LeCun et al. (1989), they are a type of Neural Network specialized in processing data which portraits a grid-like topology, this data featuring spatial correlation between neighborhood data points, for instance time series's represent a 1D grid and Image data, could be defined as 2D grid of pixels, they are called "Convolutional" because they employ **Convolution** instead of general matrix multiplication, a convolution is the averaging of different measurements of the input data,where the each measurement/ segment of the data is given a weight based on its relevance.

1.2. Background

" Convolutional networks are perhaps the greatest success story of biologically inspired artificial intelligence"(Goodfellow, Bengio, and Courville 2016) , they became the facto-standard approach for image classification when Krizhevsky, Sutskever, and Hinton (2012), exposed their breakthrough model now known as AlexNet, it classified the entire ImageNet Fall 2011 dataset of $15M$ images and $22K$ categories using 7CNNs with a marging of error of 15.3% , which surpassed the performance of the second best model by remarkable margin of 10.9% in the "**Top 5 accuracy**", this model featured CNN architecture of 8 layers: 5 convolutional and 3 fully-connected dense layers, with ReLU applied to the output of all layers.

Since then there have been many new models, which improved upon this technology, currently the benchmark for ImageNet classification is the CoAtNet-7 model which has a **9.12%** error rate for the "**Top 1 accuracy**", which improved upon AlexNet by a marging of **27.58%** (AlexNet's **Top 1 accuracy** error rate is 36.7%), this architecture is combines ConvNets and Transformers, the key component of the model is the Transformer block with prenormalization relative attention variant,(see Eqn.1) stacked vertically.

$$y_i^{pre} = \Sigma \frac{exp(x_i^\top x_j + w_{i-j})}{\Sigma_{\kappa \in G} exp(exp(x_i^\top x_\kappa + w_{i-\kappa})} \tag{1}$$

1.3. Data Augmentation

"The best way to make a machine learning model generalize better is to train it on more data"(Goodfellow, Bengio, and Courville 2016), this an idea generally accepted, yet in reality with limitation in the amount of data people can gather, a practice called Data Augmentation has been adopted, as it name implies it consist in artificially enlarging the dataset by performing some transformations, this practice has been proven to enhance the results of Image Classification tasks, Perez and Wang (2017) reports improvement of up to 6% when adding an augmentation function.

1.4. Problem Domain

Image Classification concerning 8 classes **{ Aeroplane, Car, Cat, Dog, Flower, Fruit, Motorbike, Person}**, implementing a CNN.

2. Methods

2.1. Model

2.1.1. Conv2D Layer

Creates a convulution kernel that is convolved with the layer input to produce a tensor of outputs (Keras 2021).The present model have 4 consecutive Conv2D layers.

2.1.2. Flatten Layer

Converts a Matrix to a 1 Dimensional array.It reshapes the data into necessary shape for the next layer.The model has 1 Flatten layer between the Conv2D and Dense layers.

2.1.3. Dense Layer

Also called **"Fully Connected Layer"** as the neurons of the layer are connected to each node of the preceding layer, it changes the dimension of the preceding layer, to facilitate the model's capability of defining relationships between the values in the dataset. The present model has 2 Dense Layers with units 512 and 64 respectively.

2.1.4. Dropout Layer

(Srivastava et al. 2014) Is a technique to prevent overfitting it does this by dropping out units (hidden and visible) in a neural network. The model has dropout layer with 15% drop rate between the dense layers.

2.2. Loss Function

Sparse Categorical Crossentropy, it measures the performance of a model prediction value 0 and 1, the Cross-entropy loss increases as the prediction diverges from the actual label(Murphy66 2021),sparse categorical is a type which takes 1D inputs.It is mathematically defined by:

$$-\sum_{c=1}^{M} y_{o,c} log(p_{o,c}) \quad (2)$$

- Where the number of classes $M > 2$.
- y binary indicator(0 or 1) if class label c is the correct observation for observation o.
- p predicted probability observation o of class c.

2.3. Optimizers

2.3.1. Notation

1. α: learning rate.
2. $\beta 1, \beta 2 \in [0, 1)$: Exponential decay rates for the moment estimates.
3. $f(\theta)$: Sthocahastic objective function with parameters θ.
4. θ_0: Initial parameter vector.
5. w: weight parameter to update where the subscript t indexes the weight at time step t.
6. $\partial L/\partial w$ gradient of L, the loss function to minimise,$w.r.t$ to w.

2.3.2. Adam

Adam is an algorithm for first-order gradient-based optimization of stochastic objective functions, based on adaptive estimates of lower-order moments(Kingma and Ba 2014), it combines the Adaptive Gradient Algorithm(AdaGrad) and the Root Mean Squared Propagation(RMSProp).It acts upon the gradient component by using m,the exponential moving average of gradients and the learning rate component by dividing α by square root of v,the exponential moving average of squared gradients(Karim 2018).Described by :

$$wt + 1 = w_t - \frac{\alpha}{\sqrt{\hat{v}_t + e}} . \hat{m}_t \quad (3)$$

where,

$\hat{m}_t = \frac{m_t}{1-\beta_1^t}$
$\hat{v}_t = \frac{v_t}{1-\beta_2^t}$

2.3.3. Nadam

(Dozat 2016) It introduces nesterov momentun to update the gradient one step ahead by replacing the previous $\hat{m}$ in the adam algorithm by :

$$wt + 1 = w_t - \frac{\alpha}{\sqrt{\hat{v}_t + e}} (\beta 1 \hat{m}_t + \frac{1 - \beta 1}{1 - \beta_1^t} . \frac{\partial L}{\partial_t}) \quad (4)$$

where,

$\hat{m}_t = \frac{m_t}{1-\beta_1^t}$
$\hat{v}_t = \frac{v_t}{1-\beta_2^t}$

and

$m_t = \beta_1 m_{t-1} + (1 - \beta_1) \frac{\partial L}{\partial w_t}$
$v_t = \beta_2 v_{t-1} + (1 - \beta_2) [\frac{\partial L}{\partial w_t}]^2$

2.4. Dataset

- Size: The dataset is comprised of 16000 images, resulting 1.36GB of occupied space, pertaining to 8 categories (See Problem Domain), where 2000 images have been allocated to each class.
- Split: The has a split of 85:10:5 for Train,Test and Validation respectively.

- Source : Created Parting from the Natural Images Dataset from Roy et al. (2018).
- Picture Size : The maximum dimension across the set of images is 300x300p and the minimum is 100x100p.

2.5. Software

The model was created using python as its programming language it has the following dependencies:

- Tensorflow 1.15.2
- Keras 2.3.1
- Numpy 1.19.5
- Cv2 4.1.2

3. Experiments

The model was fine tuned by changing its hyperparameters specifically the dropout rate, and optimizer.

3.1. Experimental settings

The model has 4 convolutional layers and 2 dense layers, with a dropout of 15% model one uses Adam optimizer and model 2 uses Nadam, the number of epochs is 37.

3.2. Evaluation criteria

- Achieve the most homogenous plot, with the best convergence between training and validation
- Improve the average Accuracy of the model.

3.3. Results

Model one (See Fig.1 and Table.1)performs the with an improve of 8% over model 2 (See Fig.2 and Table.2) , and clear better convergence in the plots.

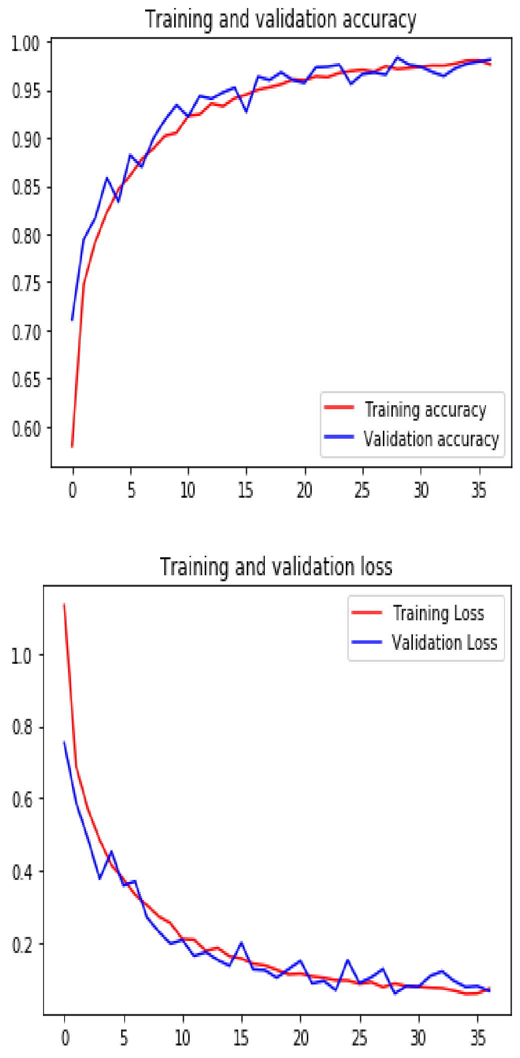

Figure 1. Training Metrics

Table 1. Model 1

Class name	Precision	Recall	F1-Score
Aeroplane	98%	99%	98%
Car	97%	99%	98%
Cat	96%	95%	96%
Dog	96%	96%	96%
Flower	98%	97%	98%
Fruit	98%	98%	98%
Motorbike	100%	99%	99%
Person	100%	99%	100%
Average	98%	98%	98%

Table 2. Model 2

Class name	Precision	Recall	F1-Score
Aeroplane	86%	90%	88%
Car	93%	94%	93%
Cat	75%	80%	78%
Dog	80%	75%	78%
Flower	89%	91%	90%
Fruit	96%	90%	93%
Motorbike	98%	99%	99%
Person	96%	99%	98%
Average	90%	90%	90%

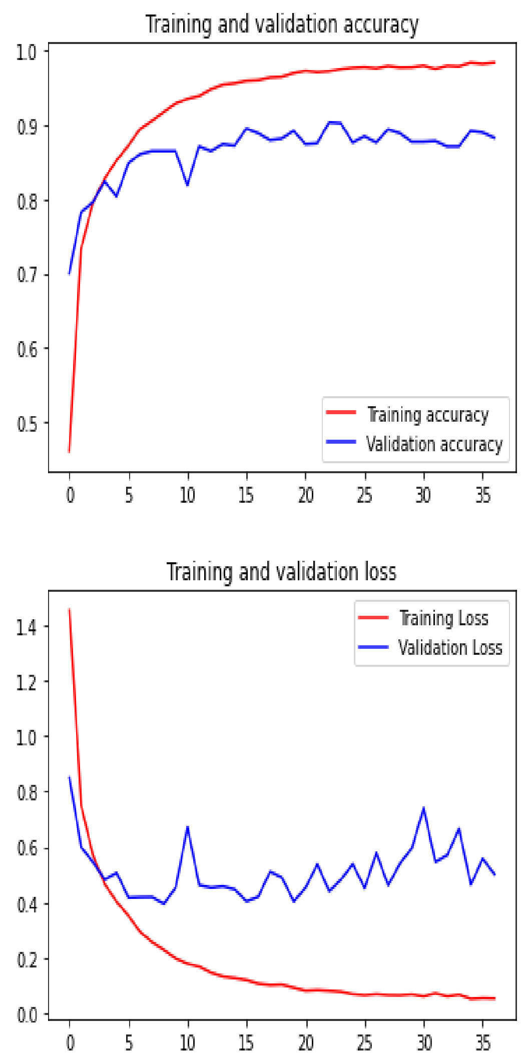

Figure 2. Training Metrics

3.4. Discussion

Image classification is a resource intensive task, where the need of computational power by the time taken to train a model, where the prediction results of these models will change drastically ,given the parameters used in the model, the presented model 1 and model 2 where the best results across various experiments, based in hyperparameter, it was observed that models fed with the current dataset over fitted after 50 epochs and the results did not improve after 37, as a results the final models use the later configuration,it is worth adding that data augmentation proved being a key factor in model accuracy as models trained without its presence had on average 10% decrease in average accuracy.

4. Conclusion

The Adam algorithm outperforms Nadam by a significant margin of accuracy, so it could be argued that Nadam is not necesarely a better version of Adam, for later experimentation it has been considered that many different optimizers as AdaGrad and RMSprop, which are the base of Adam and Nadam.

References

Dozat, Timothy (2016). “Incorporating nesterov momentum into adam”. In:

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep learning*. MIT press.

Karim, Raimi (2018). *10 Stochastic Gradient Descent Optimisation Algorithms + Cheat Sheet*. URL: `https://towardsdatascience.com/10-gradient-descent-optimisation-algorithms-86989510b5e9`.

Keras (2021). *tf.keras.layers.Conv2D*. URL: `https://www.tensorflow.org/api_docs/python/tf/keras/layers/Conv2D`.

Kingma, Diederik P and Jimmy Ba (2014). “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980*.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems* 25, pp. 1097–1105.

LeCun, Yann et al. (1989). “Generalization and network design strategies”. In: *Connectionism in perspective* 19, pp. 143–155.

Murphy66 (2021). *Loss Functions*. URL: `https://ml-cheatsheet.readthedocs.io/en/latest/loss_functions.html`.

Perez, Luis and Jason Wang (2017). “The effectiveness of data augmentation in image classification using deep learning”. In: *arXiv preprint arXiv:1712.04621*.

Roy, Prasun et al. (2018). “Effects of Degradations on Deep Neural Network Architectures”. In: *arXiv preprint arXiv:1807.10108*.

Srivastava, Nitish et al. (2014). “Dropout: a simple way to prevent neural networks from overfitting”. In: *The journal of machine learning research* 15.1, pp. 1929–1958.