Contents lists available at Crown Academic Publishing

International Journal of Engineering and Artificial Intelligence
Journal home page: http://wwwijeair.com

Empirical Design Framework for Development of Convolutional Neural Network Based Model

S. O Subairu[1], J. K Alhassan[1], S.M Abdulhamid[1], J.A Ojeniyi[1]

[1]Cyber Security Science, Federal University of Technology Minna, Nigeria

Corresponding Author: Islam4life@futminna.edu.ng

Original article

Received 17July 2020, Accepted 10 August 2020, Available online 1 September 2020

ABSTRACT

Convolutional Neural Network (CNN) has been described by most researchers as the best when it comes to image classification problems. This Neural Network is made up of high sensitive hyperparameters, such that if not properly design could lead to model misclassification and such returns high false positive (FP) and high false negative(FN). In other to solve this problem, this research proposed and developed design frameworks that mitigate this identified problem when it comes to image classification model using a Convolutional Neural Network.

Keywords: Convolutional Neural Network, Hyperparameters, Model, False Positive, False Negative

1. Introduction

Image classification, which can be defined as the task of categorizing images into one of several predefined classes, is a fundamental problem in computer vision. This is the framework to other tasks in computer vision as seen in localization, segmentation and detection(Durai Murugan et al, 2019)

There are many variations in the CNN architecture, but usually it has three basic types of layers: Convolution, pooling and fully-connected layers. The convolution layer usually has many convolution kernels which often learn feature representations from the inputs volume to produce a feature maps. The result obtained by convolving the input volume together with learned kernel is called a feature map, and next, an element-wise non linear activation function is performed on the convolved result(Namatēvs, 2018). CNN designed in such a way to learn features automatically and adaptively through backpropagation by using several building layers, such as convolution, pooling, and fully connected layers.

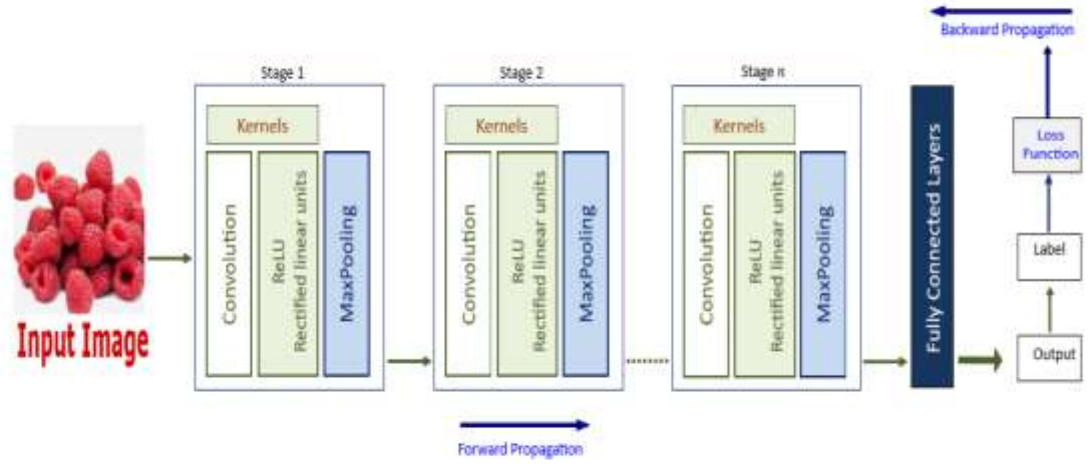

Figure 1: Convolutional Neural Network (Naranjo-Torres et al., 2020)

.

Right from 1980, solving visual task has always been approached using Convolutional Neural network (CNN). Though they were not too popular as little application were available until recently dating from year 2000, which witnesses a great development as in computational power of system and large volume of labeled and processed data. This development coupled with better enhanced algorithm contributed to the advancement in the usage of this neural network (Kalange & Mutalikdesai, 2019).

2. Literature Review

(Tabian, Fu, & Khodaei, 2019)With recent high computational power of system, artificial intelligence system has become easy to build using deep learning approach unlike in the past.The system infrastructure is hinged on levels of perceptions, while each layer of the computation is characterized in terms of its relation to concepts as essential layer consists of simple concepts. Drawing a graph will show how these concepts are built over each other; graph is deep, with so many layers. Thus, this approach is called deep learning covering several aspects of machine learning.

(Gu et al., 2018)Topology of CNN has three main concepts, which are local receptive fields, shared weights and spatial or temporal sampling. Thus CNNs are mainly made up of different kinds of layers called as follows; Convolutional layers, whereas each Convolutional layer is made of small kernels that allow extracting high-level features in an effective way. The last Convolutional layer is fed to fully connected layers.

(Namatēvs, 2018)CNN model often has a structure made up of input layer, Convolutional layers in alternating form, pooling layers or subsampling and non-linear layers. The last often has a small number of fully-connected layers, but the final layer is usually a sofmax classifier. Each Convolutional layer has more than one stage, as a result, each stage of the Convolutional layer can be set alone and every computational step of processing of it can be ruled in its own rights.
In order to reduce the computational time which is usually high, Convolutional layers are interspersed with sub-sampling layers

(Yaseen, 2018)Reduction in the parameters to be learned by CNN leads to less connections and model training becomes easier as compare with traditional neural networks. The latter uses a matrix product AB that is produced by multiplying two matrices n x m, where n is matrix of parameters and m is a parameter that is describing the main interaction between each input unit and the output unit. Major advantage of using CNNs when compare to the traditional fully-connected neural networks is the reduction in number of parameters to be learned.

(Sharma, Jain, & Mishra, 2018)Initially classification tasks were solved by adopting two levels of approach; handcrafted features were first extracted from images using feature descriptors, and these served as input to a trainable classifier. The major hindrance of this approach was that the accuracy of the classification task was profoundly dependent on the design of the feature extraction stage, and this usually proved to be a formidable task

(Naranjo-Torres et al., 2020) Deep learning models recently uses multiple layers of nonlinear computational processing, majorly to extract features, transformation, and for pattern detection, analysis and classification. This approach has shown considerable edge to overcome challenges encountered in the manual feature extraction.

3. Methodology Framework for CNN Model Design

In order to effectively design a CNN based model that will mitigate misclassification, the followings approach should be taken in the course of building various parameters of its network:

3.1 Basic Layers of Convolutional Neural Network

The basic layers of Convolutional neural network used for the model design are convolution layer, pooling layer and fully connected layer. These layers are stalked to form CNN model architecture with other hyperparameters. The breakdowns of this architecture with other parameters are as follows:

I. The Input Volume

The input volume of our design model (m*n*q); which stores the input image pixels attributes or values, where (m) is the input image width; (n) is the input image height and (q) represent the channels.

Input Volume (W_{in}) = m * n * q

II. Convolutional Layer

This layer computes the output neuron which connected to a point in local region of the input volume, thus a dot product computation between the weight and the local region connected to in the input volume is performed. This then result in output volume such as:

$$\text{Output Volume } (W_{out}) = m * n * k$$

Where (m) is the output volume width, (n) is the output volume height and (k) is the number of filter uses during computation.

III. RELU Activation

At this point in the model design architecture, application of activation function as shown in Figure 2, in elementwise is performed; where a threshold value at zero is max (0, x). The size of output volume remains the same as;

Output Volume (W_{out}) = m * n * k

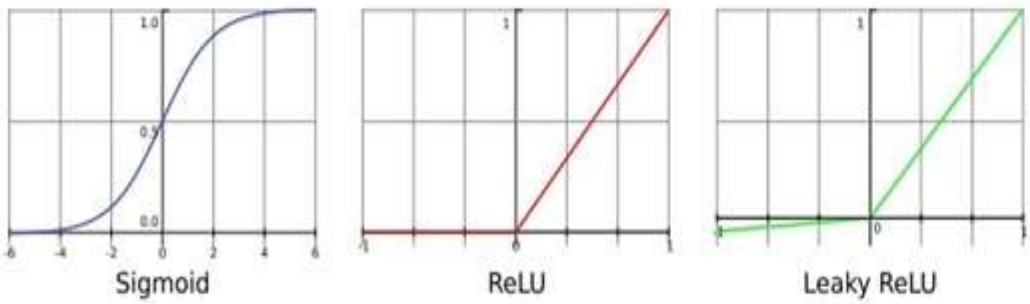

Figure 2: Types of Activations Used in CNN (Kuo, 2016)

IV. Pooling Layer

Down sampling of output neuron from previous Convolutional layer occurred at this layer along spatial dimension (width (m), height (n)); this resulted in:

$$W_{out} = m/x \times n/x$$

Where (x) is the stride value of the pooling function.

V. Fully Connected Layer

This is the last layer in the CNN architecture; it computes the input volume class scores after undergoing series of convolution, activations and pooling computation. The FC layer gives an output volume of:

$$m * n * c$$

Where m = n = width and height and c is the classes of classification.

VI. Hyperparameters

The following hyperparameters were often used in the design of CNN based model, after a thorough studied based on earlier works, presented are selection factors for better model performance.

VII. Receptive Field Size

Though in practice it is a common norm to select small size receptive field size such as (3*3) for small input volume images and (5*5, or 7*7 and more) for large input volume images. The following factors as shown in Table 1 ought to be considered for proper selection of receptive field size in the model design.

Table 1: Factors for Filter Size Selection

Smaller Filter Sizes	Larger Filter Sizes
Lower number of weights but more layers	Higher number of weights but lesser layers
Computationally efficient	Computationally expensive
With more layers, it learns complex, more non-linear features	With fewer layers, it learns simpler non linear features.
With more layers, it necessitates the need for larger memory.	It will use less memory for back propagation.
Smaller receptive field as it looks at very few pixels at once	Larger receptive field per layer.
Highly local features extracted without much image overview	Quite generic features extracted spread across the image
Therefore captures smaller, complex features in the image	Therefore captures the basic components in the image
Amount of information extracted will be vast, useful in later layers	Amount of information extracted are considerably lesser
Slow reduction in the image dimension can make the network deep	Fast reduction in the image dimension makes the network shallow
Better weight sharing	Poorer weight sharing

VIII. Stride Width

The default stride width of one is better used in the design of CNN model; this is to avoid using padding to handle the receptive field falling off the edges of the input volume. For larger images, it could be increase to two or more.

IX. Number of Filters

Filters are the features detector; generally fewer filters are used at the input layer and increasingly more filter used at deeper layers.

X. Padding

Padding could be set to zero in the design; thus this zero padding concept has been utilized in the design to make the model when implemented more generalize in terms of input image size. Convolution design and implementation could either be valid convolution or same convolution. For same convolution border around input image is used such that the image at input and output are same size. In case of same convolution, the padding width should satisfy this equation:

$$P = \frac{\mathrm{f} - 1}{2}$$

Where P is the padding and f is the filter dimension; usually odd.

XI. Feature Map

The feature map that will be generated after applying a kernel or filter on the input image will be represented mathematically as:

$$\mathrm{G}\,[\mathrm{m}, \mathrm{n}] = (\mathrm{W}*\mathrm{f})\,[\mathrm{m}, \mathrm{n}] = \sum_{\mathrm{j}} \quad \sum_{\mathrm{k}} [\mathrm{j}, \mathrm{k}] \mathrm{f}[\mathrm{m} - \mathrm{j}, \mathrm{n} - \mathrm{k}]$$

Where W is the input image, f is the kernel or filter and (m, n) are rows and columns.

XII. Output Volume

The dimension of output volume matrix considering padding, stride and filter is thus represented mathematically as:

$$W_{out} = \frac{\mathrm{Win} - f + 2p}{s} + 1$$

Where W_{in} is the input image matrix dimension, f is the filter, p is the padding and s is the stride.

4. Model Design Pattern Architecture

It is a common practice to pattern the layers in convolutional neural network architecture, thus the model design pattern as follows:

INPUT – [CONV-RELU-POOL]*^{N} – FC- RELU – FC – SIGMOID ACTIVATION

Where (N) could be any number depending on Convolutional layer to be implemented based on problem to be solve , however a single CONV layer exist in between every Pool layer.

5. Analysis and Result

The presented design framework for image classification problem using a Convolutional neural network based model has really performed well on the preliminary result obtained from its implementation as shown in Figure 3 and Figure 4.

```
In [95]: tn, fp, fn, tp = confusion_matrix([0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0

In [96]: (tn, fp, fn, tp)
Out[96]: (99, 1, 0, 100)
```

Figure 3: Model Based on Design CNN Framework Evaluation

Figure 4: Graphical Evaluation of Model Based on Design CNN Framework

6. Conclusion

The results after proper implementation of proposed CNN design framework shows an insignificant misclassification as the false positive rate was extremely low, likewise the false negative. Thus, we can conclude that the proposed design framework for Convolutional Neural Network based model for image classification has achieved its aim, hence following this framework in CNN design will solve problem of misclassification.

References

Durai Murugan, N., Chokkalingam, S. P., & Belhaouari, S. B. (2019). Analysis of deep learning models using convolution neural network techniques. *International Journal of Engineering and Advanced Technology*, *8*(3 Special Issue), 568–573.

Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., … Chen, T. (2018). Recent advances in convolutional neural networks. *Pattern Recognition*, *77*, 354–377. https://doi.org/10.1016/j.patcog.2017.10.013

Kalange, P., & Mutalikdesai, M. (2019). Deep learning through convolutional neural networks. *International Journal of Recent Technology and Engineering*, *8*(3), 1463–1465. https://doi.org/10.35940/ijrte.B3770.098319

Kuo, C.-C. J. (2016). *Understanding Convolutional Neural Networks with A Mathematical Model*. 1–21. https://doi.org/10.1016/j.jvcir.2016.11.003

Namatēvs, I. (2018). Deep Convolutional Neural Networks: Structure, Feature Extraction and Training. *Information Technology and Management Science*, *20*(1), 40–47. https://doi.org/10.1515/itms-2017-0007

Naranjo-Torres, J., Mora, M., Hernández-García, R., Barrientos, R. J., Fredes, C., & Valenzuela, A. (2020). A review of convolutional neural network applied to fruit image processing. *Applied Sciences (Switzerland)*, *10*(10). https://doi.org/10.3390/app10103443

Sharma, N., Jain, V., & Mishra, A. (2018). An Analysis of Convolutional Neural Networks for Image Classification. *Procedia Computer Science*, *132*(Iccids), 377–384. https://doi.org/10.1016/j.procs.2018.05.198

Tabian, I., Fu, H., & Khodaei, Z. S. (2019). A convolutional neural network for impact detection and characterization of complex composite structures. *Sensors (Switzerland)*, *19*(22), 1–25. https://doi.org/10.3390/s19224933

Yaseen, A. F. (2018). A Survey on the Layers of Convolutional Neural Networks. *International Journal of Computer Science and Mobile Computing*, *7*(12), 191–196.