FUSIONET: A Scalable Framework for Image Classification.

Molokwu C. Reginald
Department of Computer Science
Chukwuemeka Odumegwu Ojukwu University
Uli, Anambra, Nigeria
live.reginald@gmail.com

2nd Molokwu C. Bonaventure
School of Computer Science
University of Windsor
Ontario, Canada
molokwub@uwindsor.ca

3rd Molokwu C. Victor
School of Energy, Geoscience, Infrastructure, and Society
Harriot-Watt University
Edinburgh EH14 4AS, United Kingdom
victormolokwu@hw.ac.uk

4th Okeke C. Ogochukwu
Department of Computer Science
Chukwuemeka Odumegwu Ojukwu University
Uli, Anambra, Nigeria
okekeogo@coou.edu.ng

***Abstract*——Convolutional Neural Networks have become *state-of-the-art* methods for image classification in recent times. CNNs have proven to be very productive in identifying objects, human faces, powering machine vision in robots as well as self-driving cars. At this point, they perform better than human subjects on a large number of image datasets. A large portion of these datasets depends on the idea of solid classes. Hence, Image classification has become an exciting and appealing domain in Artificial Intelligence (AI) research. In this paper, we have proposed a unique framework, FUSIONET, to aid in image classification. Our proposition utilizes the combination of 2 novel models in parallel (MainNET, a *3 x 3*, architecture and AuxNET, a *1 x 1* architecture) Successively; these relatively feature maps, extracted from the above combination are fed as input features to a downstream classifier for classification tasks about the images in question. Herein FUSIONET, has been trained, tested, and evaluated on real-world datasets, achieving *state-of-the-art* on the popular CINIC-10 dataset.**

Index Terms*—Nueral Networks, Image classification, Pattern recognition, *state-of-the-art

I. Introduction

Krig 2014 Convolutional Neural Networks, shortened as Convnets or CNNs, is a kind of Multilayer Perceptron (MLP). A synopsis of CNN inspirations is given by LeCun et al. 1998, who is viewed as one of the pioneers in this field. Convolution is the essential operation used to demonstrate an artificial neuron to both learn and distinguish the highlights, utilizing a weight network convolved against an input window of pixels. Convolutions used as a correlation template or feature identifier. The product of each convolutional channel is gathered into an output picture alluded to as a feature ma, wich is sent along as input to the following layer. One output image is made for each channel, and there are typically many channels per layer. Each convolutional channel goes about as both an feature detector and a channel. The convolutional channels are formed together into include sets at each degree of the progression, and each channel is tuned during training. Convolution Neural Networks are utilized for numerous ranging from classification of families of objects, recognition of specific objects within a class, localization of objects to find coordinates, and segmentation of region of pixels into classes, and general regression analysis. The Convnet is a feed-forward system, with inputs moving from through processing to output classification. Be that as it may, the training phase for feature learning operates backwards, computing the error between expected results and computed results, and distributing the errors back to their source to correct the filter weights (features) is a method referred to as backpropagation. Nonlinearity or activation function is accepted to help take care of issues of input saturation, maybe brought about by numeric overflow perhaps due to poor lighting or very strong lighting. Pooling is referred to as another name for subsampling in Convolution Neural Network parlance. Pooling is typically used as the last step, or one of the last steps, in each convolutional layer Fukushima and Miyake 1982. The expressed objective for pooling is commonly to include some translational invariance, or to just subsample the picture littler to decrease figure what's more, boundaries. Notwithstanding, the subsampling strategies utilized in Neural Networks are different than standard computer graphics sampling methods such as linear interpolation and other anti-aliasing methods. In Neural Networks, FC layers go about as a scaffold between the filtering layers and a classifier. FC models are flawlessly teachable in Convolution Neural Network models, since the equivalent convolutional neural weight model is utilized. Fully Connected layers are inclined to overfitting in this manner, for high boundary checks, different moderation procedures are utilized during training to diminish and regularize the boundaries, for example, haphazardly dropping inputs with a given rate. A fully connected layer is a general function approximator, capable of utilizing a linear classifier, or a logistic regression function for binary classification, perhaps using Hamming distance for matching. By weighting each input to every neuron in the FC layer, the combination of inputs and weights forms a linear classifier. Typically, a couple FC layers stacked together are adequate to rough the ideal function. One key guideline of Convolution Neural Network idea is the utilization of imitated convolutional layers in the engineering, Typically

the convolutional layers are recreated, or, more than likely changed marginally from layer to layer. For instance, a convolutional layer may alternatively incorporate numeric conditioning of the input data, convolutional filtering, followed by a nonlinear transform of the convolutional result, pooling and subsampling, and local region post-processing of the data. nearby locale post-preparing of the information. Parameters are the program code for the CNN. The design is set up to program itself through tuning the learning parameters and the training data, bringing about component or feature learning. The parameters are comparable to the neural DNA code in the cerebrum. When breaking down engineering multifaceted nature, we might be intrigued in analyzing parameters to compare design alternatives. Knowing that each parameter implies corresponding memory and compute operations, parameter analysis is handful for estimating training time and run-time performance.

II. Review of Related Works

Molokwu 2019A significant hassle in machine vision is to devise effective and efficient means of transferring humans' informal knowledge (like sense of image recognition, understanding of image location, etc.) into machines and computers such that these machines can act and behave exactly like humans. However, the occurrence of objects concerning image representations in the real-world is usually associated with various features of variation or factors of influence that constitute distractions or noise in the image representations. Hence, it tends to be very difficult to disentangle these abstract factors of influence from the principal object or observed entity. To that effect, these remain open problems, and challenges to CNN and modern AI. LeCun et al. 1998 proposed a CNN architecture (LeNET) for handwritten and machine-printed character recognition, this architecture, adopted a pioneering 7-level convolutional network. One objective for LeNet was to diminish computational overhead, so cautious consideration was paid to diminishing parameters. Weight sharing was utilized to take into consideration one component at a time to be looked for. The channel loads and inclination were common for all neurons all the while in a similar layer, so virtual neurons are actualized to share weights instead exhaustively implementing all neurons with their copy of the weights—shared weights, and virtual neurons are an innovation successfully demonstrated In LeNet. While this appears to be characteristic, LeCun et al. 1998 was among the first to utilize this strategy with regards to a customary convolutional Network. Continuation of his design, Krizhevsky, Sutskever, and Hinton 2012 assembled a system (AlexNet), that had a fundamentally the same as design as [2] however was more top to bottom, having more channels per layer, and with stacked squares of convolution activity. The LeNet design is the reason for a few current CNNs, including the AlexNet CNN variety created by Krizhevsky, Sutskever, and Hinton 2012, which was the first CNN to understand the capability of Convnets on huge image datasets. AlexNet turned into the reason for ensuing Convolution Neural Networks utilized for image classification. Zeiler and Fergus 2014 improved the AlexNet design, alluded to as ZFNet, which was in this way popularized in a startup called Clarifai ZFNet success is due to Zeiler and Fergus 2014 method to visualize learned features corresponding to the pixel regions they match in the input image, referred to as deconvolutional networks, to develop intuition about how to enhance the learning hyperparameters to improve weak features. Likewise, ZFNet decreased the quantity of hyperparameters, and extended the convolutional filter layer profundity, improving the exactness by a few rate focuses. In 2014, Simonyan and Zisserman 2014, built up the primary forms of VGGNet utilizing up to 19 layers of *3 x 3* convolutional pieces as opposed to a course of action of bigger parts, with insignificant different tasks, to be specific - pooling and ReLu, with fantastic outcomes. Krig 2014The focal idea was to utilize stacked convolutions to decrease the parameter tally and increment performance. A heap of three 3 *3 x 3* convolutions covers the equivalent responsive field and approximates a *7 x 7* convolution; in any case, utilizing stacked convolutions utilizes far less parameters, coming about in noteworthy performance advantage at training time and run time. Be that as it may, it might be contended that bigger filters for example, *11 x 11* or *15 x 15* are priceless for a few applications requiring more detail to depict features, where *3 x 3* stacked decreases may not work due to the constrained receptive field. VGGNet has propelled some eminent Convolution Neural Networks, for example, He et al. 2016, Wu et al. 2015. The InceptionNet design presented by Szegedy et al. 2016 in 2015 with design so close as the natural eye yet extremely perplexing. The system utilized a CNN roused by LeCun et al. 1998 yet executed a novel component, which characterizes a total of various estimated convolution channels at a similar layer, giving a multi-scale convolution, feature space measurement decrease utilizing the *1 x 1* convolution model presented by Lin, Chen, and Yan 2013. The InceptionNet design *10 x* parameter count reduction over the Krizhevsky architecture, and *25 %* less compute. The Inception engineering included a 22 layer profound CNN with 4 million parameters. Upadhyay 2019 At last, at the ILSVRC 2016, the presentation Residual Neural Network (ResNet) by He et al. 2016 novel a structure with "skip connections," such skip connections are otherwise called gated units or on the other hand gated repetitive units. They have significant closeness to ongoing fruitful components applied in Recurrent Neural System. With this strategy, they had the option to prepare a Convolution Neural Network with 152 layers. Researchers are on the chase to additionally grow the precision and speed of the underlying algorithms, consequently making the field of Image classification a functioning region of explorative research. Lu et al. 2020a presented a Neural Architectural Transfer model that was intended to Lu et al. 2020b productively create task-explicit custom models that are competitive even under multiple conflicting objectives using evolutionary search algorithms. In such manner, we have proposed a system that essentially expands adaptability and precision and accomplishes best in class otherwise known as *state-of-the-art* on the famous CINIC-10 huge scope dataset for image classification, and yet accomplishes brilliant outcomes when applied to other

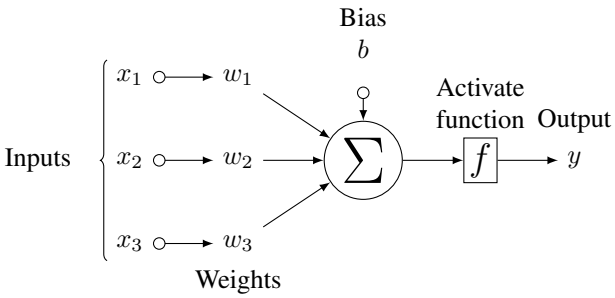
image classification datasets.

III. Our Contribution

In this paper, we propose a *state-of-the-art* architecture for image classafication (FUSIONET) from large to small scale datasets. We analyze and test our proposition on different datasets and benchmark. In this regard, we built FUSIONET on the basis of 3 significant features

- FUSIONET is the only architecture to combine 2 distinct novel models in parallel for image classification with significant results. From our experiments, we show that this is plausible, as different low level feature maps are extracted from each model. This features are then combined to give the classifier a well oriented information of the underlying image.
- FUSIONET employs uniform kernel size of *3 x 3* for MainNET and *1 x 1* for AuxNET because literature wise proposes that they cover as much details as large kernel size.
- FUSIONET aims at minimizing parameters count, taking note that deep neural networks are prone to significant large parameter count for efficient and effective classification.

IV. Proposed Methodology and Framework



A. Definition of Problem

Definition 1 Image Classification: Hashmi, Kumar, and Keskar 2020Image classification is referred as a process of directly mapping floating integers to symbols

$$f(x) : x \Rightarrow \Delta; x \in R^n, \Delta = \{c1, c2, ..., c_L\} \qquad (1)$$

Number of bands = n;
Number of classes = L
f(.) is a function that assigns a pixel vector x to a single class in the set of classes Δ

Definition 2 Convolution Neural Network: A Convolutional Neural Network (otherwise known as CNN or ConvNet) LeCun et al. 1998Molokwu et al. 2020 is a popular deep learning architecture and artificial neural network which is primarily used to extract and learn higher-order features in the dataset via convolutions. Convolutional Neural Networks are biologically inspired, thus they are modeled based on the working principles of the visual cortex present in animals. CNNs are well adapted for learning features of image input due to its significant arrangement structure of neurons (or processing units) present in its respective processing layers. Each layer in an ordinary CNN is associated with different layers in a conceivably extraordinary input/output topology. For instance, the input layers in convolutional systems are ordinarily collected into nxn kernel designs containing 2D formats or designs, and every piece is portion associated with a (virtual) separate artificial neuron for convolutional processing.

Definition 3 Convolution Operation: The convolution operation is the fundamental operation of any Convolution Nueral Network. It is responsible for extracting latent features and representations from the input image. A convolution is defined as a mathematical operation rule for merging two sets of data.

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} * \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 3 & 1 \\ 1 & 4 & 3 \\ 1 & 3 & 1 \\ 1 & 3 & 1 \\ 3 & 1 & 0 \end{pmatrix}$$

I $\quad$ K $\quad$ $I * K$

$$A_{xy} = (K * I)_{xy} = \sum_{0}^{i} \sum_{0}^{j} K_{ij}.I_{x-i,y-j} \qquad (2)$$

Formally, the above equation 2. represents a 2-dimensional convolution operation. In virtually all Deep Learning libraries, the above 2-dimensional convolution operation is implemented as cross-correlation (which is the same as convolution but without flipping the kernel); thus, it is computed as

$$A_{xy} = (K * I)_{xy} = \sum_{0}^{i} \sum_{0}^{j} K_{ij}.I_{x+i,y+j} \qquad (3)$$

Such that: A_{xy} represents a cell/matrix position in the Activation Map (or Convolved Feature or Feature Map); K_{ij} represents a cell/matrix position in the Kernel (or Filter or Feature Detector); and $I_{(x+i,y+j)}$ represents a cell/matrix position in the Input matrix. As the index, K_{ij}, of the kernel slides from left-to-right and top-to-bottom; the index, $I_{(x+i,y+j)}$, of the input matrix increases respectively - from left-to-right and top-to-bottom. Each resultant, A_{xy}, which is a convolution of the respective kernel and input indices is used to populate the activation/feature map (or convolved feature).

Definition 4 Backpropagation: Backpropagation works by taking the consequences of the forward pass through the Neural Network, finding the mis-classification between the current anticipated outcome and the right outcome, and conveying the mistake relatively in reverse through the system, to limit the mis-classification at each layer by altering the feature weight.

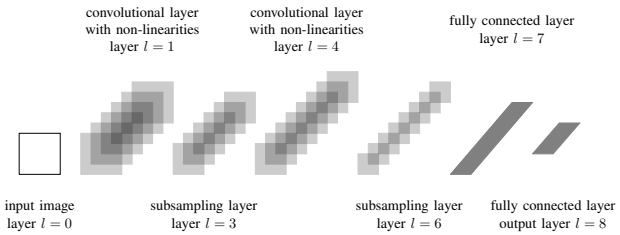
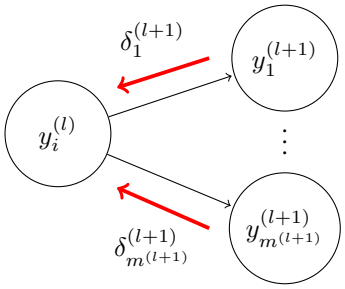


Fig. 1. Architecture of a traditional convolutional neural network. as introduced by LeCun et al. 1998



Backpropagation strategies look like a tremendous component averaging process. Numerous techniques are utilized, and some contain improvements for different objectives, for example, increasingly quick convergence.

Algorithm 1: Backpropagation learning algorithm

for d in data **do**
  FORWARDS PASS
    Compute responses *f()* at each neuron for each feature to feed forward
    Store response derivatives *f'()* at each neuron for each feature for backprop.
    Compute classification scores and errors
  BACKWARDS PASS
    Compute the derivatives of the error function with respect to the output layer activities
    for layer in layers **do**
      Distribute errors backwards to each contributing neuron
      Adjust weights using errors and stored response derivatives
      Compute residuals for current layer, distribute backwards, repeat
    end for
    Continue until stopping criteria reached (threshold, iterations, elapsed time)
end for=0

B. Proposed Methodology

Our proposition, FUSIONET, is comprisingly of (2) distinct ConvNet models *(MainNet, AuxNet)* utilizing a two-dimensional (2D) convolution operations fused together with the input *x* (Image vector) and (1) classification layer.

1) MainNET: The MainNET is a stack of (19) layers of small 3 x 3 2D Convolution block for accurate distinctions. Sequel to the 2D Convolution block is BatchNormalization which is a technique to provide any layer in a Neural Network with inputs that are zero mean/unit variance.

In this regard, Molokwu et al. 2020 Let B denote a mini-batch of size m training set, The empirical mean and variance of B is ascribed as $\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i$, and $\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2$

The non-linearity activation activity is a Parametric rectified linear unit (PReLU) work which presents non-linearity after the BatchNormalization activity. The essential artificial neural processor model utilized in many Deep Nueral Networkss comprises of two sections: (1) a convolution work: *A = f(inputs x weights)*, and (2) a nonlinear activation work: *f = f(A)* to propagate or squash the information, which delivers a scalar worth. Enactment capacities, are utilized to (1) bring nonlinearity into the neural work, (2) forestall immersion of values, and (3) guarantee that the neural output is differentiable to support back propagation techniques utilizing gradient descent. One key objective of nonlinear actuation capacities is to extend the absolutely straight convolution activity into a nonlinear arrangement space, which is accepted by numerous individuals to improve results. Furthermore, the nonlinearity may bring about quicker convergence during backpropagation preparing to move the gradient all the more rapidly out of level spots towards the nearby minima. In such manner, the corrected component/initiation map is figured by means of: $f(x) = \begin{cases} x & \text{if } x > 0, \\ ax & \text{otherwise.} \end{cases}$

Pooling is a technique for joining a few neural outputs comprising of the scalars from the activation or on the other hand transfer function into a pool, or group, and making another output from the pool. The pooling activity goes about as a specialist answerable for decreasing the information width of each redressed activation map while holding its fundamental properties. subsequently, the Max-Pooling capacity is characterized with the end goal that the resultant pooled (or downsampled) feature map is produced by means of: $p_i \in P = h(r_i \in R) = maxPool(R)$.

2) AuxNET: The AuxNet acts as a (17) layers of small 1 x 1 Convolution block otherwise recognized as a linear transformation of the input, followed by BatchNormalization, a non linear activation function and pooling.

3) Classifier: This is the last layer of our proposed FUSIONET design, and it succeeds the AuxNET layers, with (5) squares of Multi-layer perceptron. Classification is commonly the last phase of the FC vision framework, where the nearness and nonappearance of features is utilized to decide. The classifier matches distinguished examples against learned examples to distinguish the class of the input and make a score to show certainty. The pooled feature maps, created by both the MainNET and AuxNET which contains high level features removed from the constituent images in the dataset are concatenated alongside the input (x). Thus, the the classification uses these extricated "high level features" for recognizing images, in view of the individual classes. In this regard, Molokwu et al. 2020,B. C. Molokwu and Kobti 2019 a MLP function is denoted as a mathematical function, f_c that zips some set of input values, *P*, to their respective output labels, *Y* . In other words, $Y = f_c(P, \theta)$ and θ

denotes a set of parameters.The MLP function models the estimations of θ that will bring about the best end, Y , approximation for the input set, P. The MLP classifier output is a likelihood dissemination which shows the probability of a component portrayal having a place with a specific output class.

Algorithm 2: Image Classification Algorithm defined via Algorithm 2:

0: **procedure** IMAGECLASSIFICATION
Input: $\{V, E, \mathbb{Y}_{gTruth}\} \equiv$ {Images, Labels, Ground-Truth Entities}
Output: $\{\mathbb{Y}_{pred}\} \equiv$ {Predicted Entities}

Preprocessing:
$V \leftarrow$ Data *Augmentation* {(Rotation, Fliping, Cropping)}
$f_c, W \leftarrow$ *Initialize* {Construct classifier model, Weights}
Training:
$i \leftarrow 0$
while $i < 200$ **do**
$out \leftarrow f_t \in F = (K \cdot I)_t$ // Conv. Operation 1;
$out \leftarrow p_t \in out = h(R) = maxPool_(out)$;
$out \leftarrow f_t \in out = (K \cdot I)_t$ // Conv. Operation 2;
$out \leftarrow p_t \in out = h(R) = maxPool_(out)$;
$f_c|\theta : out \rightarrow \mathbb{Y}_{gTruth}$ // MLP Operation;
return:
$\{\mathbb{Y}_{pred} = f_c(\mathbb{Y}_{gTruth}, \theta)$
end procedure=0

V. DATASETS

With regard to Image classification herein, five (5) real-world benchmark image classification data sets were employed for experimentation and evaluation, ranging from large scale to medium and small datasets viz: CINIC-10 Darlow et al. 2018, (CINIC-10, CIFAR-100 Krizhevsky, Hinton, et al. 2009) SVHN Netzer et al. 2011, STL-10 Coates, Ng, and Lee 2011 .

TABLE I
BENCHMARK DATASET FOR EVALUATION.

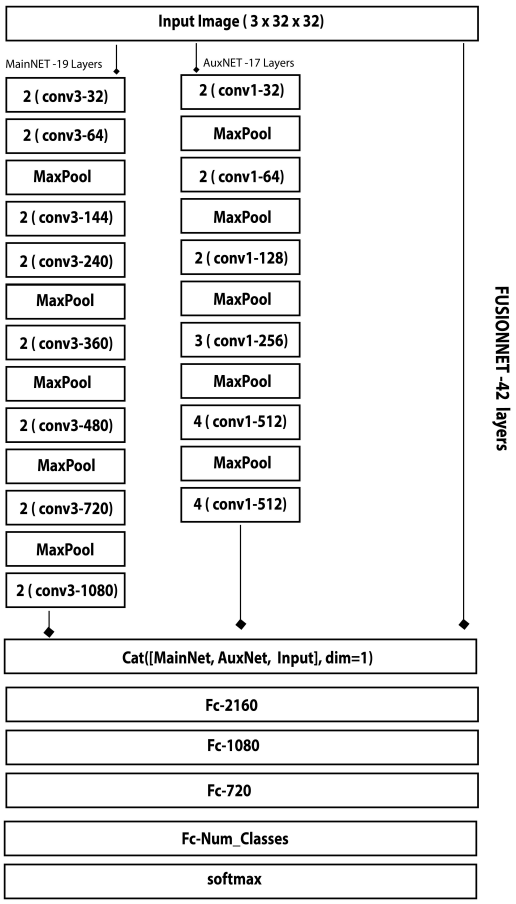
Dataset	Type	Training set	Testing set	Classes
CINIC-10		120,000	90,000	10
CIFAR-10		50,000	10,000	10
CIFAR-100	Multi-class	50,000	10,000	100
SVHN		73,000	60,000	10
STL-10		5,000	8,000	10

TABLE II
CONFIGURATION OF EXPERIMENTATION HYPERPARAMETERS.

Batch Size: 128	Optimizer: SGD (Momentum=0.99)
Activation: PReLU	Epochs: 1000
Dropout: 0.5	Learning Rate: 0.003
Learning Decay: after 300 epochs	weight decay: L2 penalty
Stride: 1	L2 Multiplier: 5e-4
Maxpooling: (2, 2)	Number of Parameters: 64M

A. Proposed Architecture/Framework

The framework parameters are denoted as "Conv(receptive field size)-(number of channels)". The PReLU activation function and BatchNormalization are not shown for brevity.



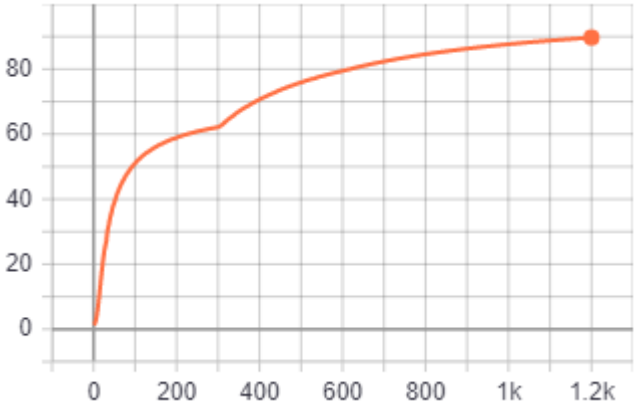
VI. TRAINING

On training, the inputs (x) to our ConvNets are fixed-size 32 × 32 RGB image. Preprocessing employed includes manually correction of wrong labelled inputs, Normalization, Flipping of random images horizontally and vertically, randomly cropping the centers of some images, erasing some parts of the images, changing the colors of random images and random perspective. The image is then passed through a block of convolutional layers, where we employed distinct filters with a small and tiny receptive field: 3 x 3, 1 x 1 for MainNET and AuxNET respectively.

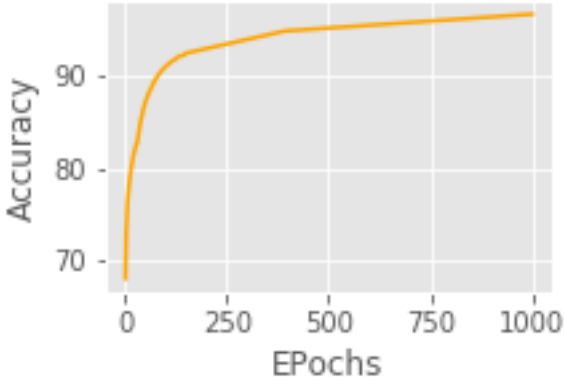
VII. Experiments, Results and Discussion

FUSIONET's experimentation setup was tuned in accordance with the hyperparameters shown in Table 2. Categorical Cross Entropy was employed as the cost/loss function; while the fitness/utility was measured with reference to accuracy at *Top1, Top5 and Flops*. Moreover, the accuracy have been computed against each benchmark data set with regard to the constituent classes (or categories) present in each data set.

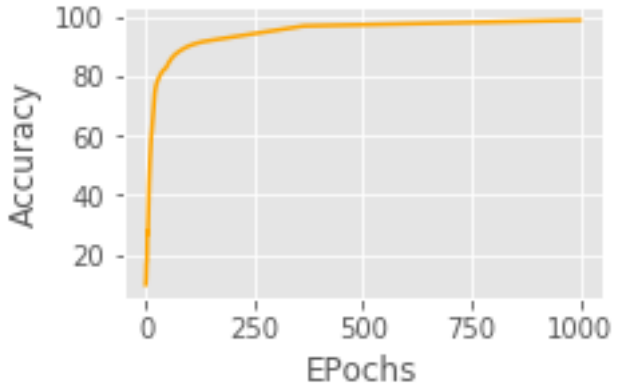
With regard to the image classification tasks herein, the performance of our FUSIONET model while benchmarking against five(5) popular baselines (NAT-M4, NAT-M3, SOPCNN, ResNeXt29x64d); and when evaluated against the validation/test samples for the benchmark data sets are as documented in Table 3, 4, 5, 6, 7



Epochs Vs Accuracy on CIFAR-100 dataset after 1200 Epochs



Epochs Vs Accuracy on CINIC-10 dataset after 1000 Epochs



Epochs Vs Accuracy on CIFAR-10 dataset after 1000 Epochs

TABLE III
IMAGE CLASSIFICATION OVER CINIC-10 DATA SET. RESULTS ARE BASED ON THE SET APART VALIDATION SAMPLES

CINIC-10				
Model	FLOPS	Top 1 accuracy	Top 5 accuracy	Error rate
NAT-M4	710M	94.80%	99.30%	5.20%
NAT-M3	**501M**	94.30%	98.60%	5.70%
ResNeXt29x64d		91.45%	98.40%	8.55%
VGG-16		87.77%	97.55%	12.23%
DenseNET-121		91.26%	98.53%	8.74%
ResNET-18		90.27%	98.01%	9.73%
FUSIONET	669M	**96.84%**	**99.88%**	**3.16%**

Table 3. Shows our FUSIONET achieving state-of-the-art on the CINIC-10 dataset for top-1 and top-5 accuracy. Improving NAT-M4 by 2.04% for top-1 and 0.58% for top-5. floating point operations per second (FLOPS) was benched at 669M with regards to NAT-M4 of 710M.

TABLE IV
IMAGE CLASSIFICATION OVER CIFAR-10 DATA SET. RESULTS ARE BASED ON THE SET APART VALIDATION SAMPLES

CIFAR-10				
Model	FLOPS	Top 1 accuracy	Top 5 accuracy	Error rate
NAT-M4	468M	98.40%	99.60%	1.60%
NAT-M3	**392M**	97.20%	98.30%	2.80%
ResNeXt29x64d		96.71%	98.40%	3.29%
SOPCNN		94.29%	98.00%	5.71%
FUSIONET	440M	**98.54%**	**99.82%**	**1.46%**

TABLE V
IMAGE CLASSIFICATION OVER STL-10 DATA SET. RESULTS ARE BASED ON THE SET APART VALIDATION SAMPLES

STL-10				
Model	FLOPS	Top 1 accuracy	Top 5 accuracy	Error rate
NAT-M4	573M	**97.90%**	**98.55%**	**2.10%**
NAT-M3	**436M**	97.80%	98.48%	2.20%
ResNeXt29x64d		80.61%	96.40%	19.39%
SOPCNN		88.08%	97.02%	11.92%
FUSIONET	490M	92.61%	98.76%	7.39%

TABLE VI
IMAGE CLASSIFICATION OVER SVHN DATA SET. RESULTS ARE BASED ON THE SET APART VALIDATION SAMPLES

SVHN				
Model	FLOPS	Top 1 accuracy	Top 5 accuracy	Error rate
NAT-M4	610M	**98.51%**	**99.67%**	**1.49%**
NAT-M3	**494M**	97.80%	99.24%	2.20%
ResNeXt29x64d		98.50%	99.48%	1.50%
SOPCNN		98.50%	99.55%	1.50%
FUSIONET	520M	97.63%	99.31%	2.37%

TABLE VII
IMAGE CLASSIFICATION OVER CIFAR-100 DATA SET. RESULTS ARE BASED ON THE SET APART VALIDATION SAMPLES

CIFAR-100				
Model	FLOPS	Top 1 accuracy	Top 5 accuracy	Error rate
NAT-M4	796M	88.30%	**95.71%**	11.70%
NAT-M3	**492M**	87.70%	95.55%	12.30%
ResNeXt29x64d		83.44%	94.48%	16.56%
SOPCNN		72.96%	90.22%	27.04%
FUSIONET	670M	**89.72%**	**96.86%**	**10.28%**

In this manner, we have highlighted the model which performed best (considering *Top-1, Top-5* precision and FLOPS where conceivable), for each classification task using a bold font. We utilized a point-based reviewing standard to discover the fittest model for each image classification task. The model with the most noteworthy combined point altogether wins the fittest model for the predetermined assignment, etc in a plunging request of cumulatives. As needs be, as shown from our plain outcomes, our proposed structure (FUSIONET) has the most noteworthy wellness focuses. To prepare the CIFAR-100 dataset, we changed a few information like augmentation tasks, decreased the batchsize to 64 and expanded the learning rate moreover the training epochs. Every single other function and hyperparameters were left unaltered all through the preparation/training procedure.

VIII. LIMITATIONS, CONCLUSION AND FUTURE WORK

Molokwu et al. 2020 The benchmark models evaluated herein were implemented using their default parameters. In summary, FUSIONET's remarkable performance with respect to our benchmarking results can be argued to be a result of the following:

1) FUSIONET is constituted of the fusing of two (2) distinct models of *3 X 3 and 1 X 1* of stacked convolution operations, BatchNormalization, non-linear activation and pooling, running independently viz: MainNET and AuxNET.
2) The thorough data analysis and preprocessing techniques employed herein with respect to the benchmark data sets. We ensured that all underlying images of a given classification task were mapped to their individual classes, thereby minimizing the noise in the dataset, and normalized prior to training and/or testing.

In conclusion, we aspire to expand our experimentation scope to include more computational visual problems, datasets and benchmark models.

IX. ACKNOWLEDGEMENT

This research was supported by Bhevencious and PESOCS. We thank our partners from Chukwuemeka Odumwgwu Ojukwu University, who gave understanding and aptitude that incredibly helped the research process, despite the fact that they may not concur with the entirety of the translations/finishes of this paper. We express gratitude toward Rita Ifunanya for remarks that extraordinarily improved the manuscript.

REFERENCES

Coates, Adam, Andrew Ng, and Honglak Lee. 2011. "An analysis of single-layer networks in unsupervised feature learning." In *Proceedings of the fourteenth international conference on artificial intelligence and statistics,* 215–223.

Darlow, Luke N, Elliot J Crowley, Antreas Antoniou, and Amos J Storkey. 2018. "CINIC-10 is not ImageNet or CIFAR-10." *arXiv preprint arXiv:1810.03505.*

Fukushima, Kunihiko, and Sei Miyake. 1982. "Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition." In *Competition and cooperation in neural nets,* 267–285. Springer.

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

Hashmi, Mohammad Farukh, Aashish Kumar, and Avinash G Keskar. 2020. "Subjective and Objective Assessment for Variation of Plant Nitrogen Content to Air Pollutants Using Machine Intelligence: Subjective and Objective Assessment." In *Fuzzy Expert Systems and Applications in Agricultural Diagnosis,* 83–108. IGI Global.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification." In *Proceedings of the IEEE international conference on computer vision,* 1026–1034.

———. 2016. "Deep residual learning for image recognition." In *Proceedings of the IEEE conference on computer vision and pattern recognition,* 770–778.

Krig, Scott. 2014. *Computer vision metrics: Survey, taxonomy, and analysis.* Springer Nature.

Krizhevsky, Alex, Geoffrey Hinton, et al. 2009. "Learning multiple layers of features from tiny images."

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton. 2012. "Imagenet classification with deep convolutional neural networks." In *Advances in neural information processing systems,* 1097–1105.

LeCun, Yann, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. 1998. "Gradient-based learning applied to document recognition." *Proceedings of the IEEE* 86 (11): 2278–2324.

Lin, Min, Qiang Chen, and Shuicheng Yan. 2013. "Network in network." *arXiv preprint arXiv:1312.4400.*

Lu, Zhichao, Gautam Sreekumar, Erik Goodman, Wolfgang Banzhaf, Kalyanmoy Deb, and Vishnu Naresh Boddeti. 2020a. "Neural Architecture Transfer." *arXiv preprint arXiv:2005.05859.*

———. 2020b. "Neural Architecture Transfer." *arXiv preprint arXiv:2005.05859.*

Molokwu, Bonaventure C, and Ziad Kobti. 2019. "Event Prediction in Complex Social Graphs via Feature Learning of Vertex Embeddings." In *International Conference on Neural Information Processing,* 573–580. Springer.

Molokwu, Bonaventure C, Shaon Bhatta Shuvo, Narayan C Kar, and Ziad Kobti. 2020. "Node Classification in Complex Social Graphs via Knowledge-Graph Embeddings and Convolutional Neural Network." In *International Conference on Computational Science,* 183–198. Springer.

Molokwu, Bonaventure, and Ziad Kobti. 2019. "Spatial Event Prediction via Multivariate Time Series Analysis of Neighboring Social Units using Deep Neural Networks," 1–8. July. doi:10.1109/IJCNN.2019.8851730.

Molokwu, Reginald. 2019. "AN ADVANCED MOVIE RECOMMENDER ENGINE IMPLEMENTED IN PYTHON" (May). doi:10.13140/RG.2.2.31431.04000.

Netzer, Yuval, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. 2011. "Reading digits in natural images with unsupervised feature learning."

Ogolo, Naomi Amoni, Victor C Molokwu, Mike O Onyekonwu, et al. 2017. "Proposed technique for improved oil recovery from thin oil rim reservoirs with strong aquifers and large gas caps." In *SPE Nigeria Annual International Conference and Exhibition.* Society of Petroleum Engineers.

———. 2018. "Techniques for effective oil production from thin oil rim reservoirs." In *SPE Nigeria Annual International Conference and Exhibition.* Society of Petroleum Engineers.

Simonyan, Karen, and Andrew Zisserman. 2014. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556.*

Szegedy, Christian, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. "Rethinking the Inception Architecture for Computer Vision." *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR):* 2818–2826.

Upadhyay, Yash. 2019. *Computer Vision: A Study On Different CNN Architectures and their Applications,* March. https://medium.com/alumnaiacademy/introduction-to-computer-vision-4fc2a2ba9dc.

Wu, Ren, Shengen Yan, Yi Shan, Qingqing Dang, and Gang Sun. 2015. "Deep image: Scaling up image recognition." *arXiv preprint arXiv:1501.02876* 7 (8).

Zeiler, Matthew D, and Rob Fergus. 2014. "Visualizing and understanding convolutional networks." In *European conference on computer vision,* 818–833. Springer.