Sub-Sahara African Academic Research Publications
December, 2020, Editions

JPAS
Journal of Pure and Applied Science
Vol..17, No.9, ISSN 2318-9315

MINIMIZATION OF TRAINING TIME OF A CONVOLUTIONAL NEURAL NETWORK BY ADDING K-NEAREASET NEIGHBOR AS CLASSIFIER

**PROF. SOULEY BOUKARI; **FATIMA AHMED ABUBAKAR; **ATIKA AHMAD JIBRIN; **ISHAQ MUHAMMED; **AMATULLAH YAHYA ALIYU; & **TURAKI ABDULKADIR MAIGARI

**Computer Science, Abubakar Tafawa Balewa University Bauchi*
***Department of Computer Science, the Federal Polytechnic Bauchi.*

ABSTRACT

Image classification is an essential task in machine learning and computer vision however it is still challenging. Although, over the past years, Convolutional Neural Networks (CNNs) have greatly boosted the performance of a series of image classification tasks, recent works have shown that the softmax commonly used at the classification layer has exhibited some limitations. These limitations include low prediction performance. This research work aims at replacing the fully connected and the softmax layers with k-nearest neighbor so as to hybridize the model. This enabled us to overcome the limitation of the softmax as well to use the KNN to classify large dataset. CNN was adopted for feature extraction and the KNN for classification. Comparison of our model was done with two existing models- the original CNN and the CNN-SVM. Each model was executed ten times on four datasets and obtained as mean for three models as 358.92s for the CNN, CNNSVM 46.02s and our model CNNKNN 39.42s as training time respectively.

Keywords: *Minimization, Training, Convolutional, Neural, Network.*

INTRODUCTION

Image classification forms an important part of image processing. The objective of image classification is the automatic allocation of image to thematic classes (Lillesand et al., 2004). Today, the internet is full of

abundance of images and videos, which is promoting the development of different kinds of applications and algorithms that can perform all kinds of tasks on them.

Artificial intelligence is a computer science field that is used for the creation of intelligent machines that behave like humans. Artificial Intelligence has been defined by Meek et al., (2016) as the capacity of a computer to perform operations analogous to learning and decision making in humans.

Machine learning (ML) is a branch of artificial intelligence in which machines learn from examples, data, and experience. Machine learning emphases on the capability of machines to accept a set of data and then learn for themselves and also to alter algorithms as they acquire more knowledge about the data they are processing. Machine learning systems can carry out complex and difficult processes by learning from data, rather than following pre-programmed rules (Sharma & Kumar, 2017).

Deep learning can be considered a subcategory in machine learning that can also be referred to as deep neural networks which describes the many layers involved. A neural network consists of a single layer of data, whereas a deep neural network has two or more. Deep learning networks need to observe large quantities of items in order to be trained. Instead of being programmed with the edges that define items, the systems learn from exposure to millions of data points. An example is the Google Brain learning which can recognize cats after being exposed over ten million images.

The convolutional neural network (CNN) has recently achieved great success in many computer vision tasks. CNN was partially inspired by neuroscience and thus shares many of the brain's properties. Training a CNN can be achieved by back-propagating the classification error, which requires a reasonable amount of training data based on the size of the network.

Image classification can be defined as the task of categorizing images into one of several predefined classes, is a fundamental problem in computer vision. According to Karpathy (2016), image classification forms the basis for other computer vision tasks such as localization, detection and segmentation. Image classification is an important task in the field of machine learning and image processing, which is widely used in many

fields, such as computer vision, network image retrieval and military automation target identification. K-Nearest Neighbor (KNN) algorithm is one of the typical and simplest method used in image classification (Dang et al., 2018)

K-Neareast Neighbor (KNN) is a simple algorithm that stores all available cases and classifies new cases based on a similarity measure. Nearest Neighbor classification of objects is based on their similarity to the training sets and the statistics defined. KNN's basic idea is that if the majority of the k nearest samples of an image in the feature space belong to a certain category, the image also belongs to this category. KNN consists of two main procedures: similarity computing and k nearest samples searching. Since KNN requires no learning and training phases and avoids overfitting of parameters, it has a good accuracy in dealing with classification tasks with more samples and less classes. The k nearest neighbor (KNN) classifier is based on the Euclidean distance between a test sample and the specified training samples as described in (Weinberger et al., 2006).

CNNs, as standard feature extractors, have been continuously used to improve computer vision in terms of accuracy. This implies shunning the traditional hand-crafted feature extraction techniques in computer vision problems. The features learned from CNNs are generated using a general-purpose learning procedure (Lecun et al., 2015). Combining both hand-crafted features and machine learned feature is increasingly becoming a hot spot (Li et al., 2016).

The common architecture of CNNs contains many layers to recurrently extract suitable image features and feed them to the softmax function (also known as multinomial logistic regression) for classification (Szegedy et al., 2015). Agarap, (2017) replaced softmax with Support Vector Machine (SVM) in order to overcome the limitation of softmax classifier which according to Tang, (2013), often displays a low prediction performance.

KNN needs to compute the distance (or similarity) of all training samples for each test sample in the process of selecting k nearest neighbors. The high cost (i.e. linear time complexity over the sample size) prohibits the traditional KNN method to be used in big data (Zhu et al., 2014). According to Nugraheeni & Mutijarsa, (2016), at every stage of supervised learning K-NN can perform better than the SVM.

The aim of this work is to propose a hybrid model for image classification with reduced complexity and improve precision that can be easily trained and can adapt to different data and tasks for image classification. We apply convolutional neural networks (CNN) for feature extraction to reduce image dimensions for faster and accurate classification. Then incorporate K-Nearest Neighbor (KNN) as the classifier for the images based on the extracted features. We further evaluate our model using MNIST, Fashion_MNIST, CIFAR10 and CIFAR100 data sets against the typical convolutional neural network with softmax (Lecun et al., 2015) and the work of Agarap, (2017).

Image classification is an important aspect in the field of machine learning and image processing and can be widely applied in many fields, such as computer vision, network image retrieval and military automation target identification. K-Nearest Neighbor (KNN) is robust to noisy training data and can be effective in training large data sets. It can also be used for categorical data or the mixture of both categorical and continuation data. When implemented successfully, this work should be able to present an improved CNN model that can be used to classify images.

The rest of the work is organized as follows: Section two looked at the various literature on related topics. Section three presented the methodology of the work. In section four, experiments were carried out and the results obtained were explained. Summary, conclusion and recommendation were contained in section five.

LITERATURE REVIEW

Wagner et al., (2013) compared the performance of unsupervised feature learning and transfer learning against simple patch initialization and random weight initialization within the same setup. They showed that pre-training helps to train CNNs from few samples and that the correct choice of the initialization scheme can push the network's performance by up to 41% compared to random initialization. Their results show that pre-training systematically improves generalization capabilities when handling datasets with few samples. They concluded that the choice of a pre-training method depends highly on the dataset used. They used the traditional filter selection and softmax as their classifier.

In their work, Palaniappan et al., (2014) evaluated and compared the performance of the support vector machine (SVM) and K-nearest neighbor (KNN) classifiers in diagnosis respiratory pathologies using respiratory sounds from R.A.L.E database. They measured the performances of the classifiers using the confusion matrix technique. They found that the K-NN classifier was better than the SVM classifier for the discrimination of pulmonary acoustic signals from pathological and normal subjects obtained from the RALE database.

Lavin & Gray (2016) derived a new class of fast algorithms for convolutional neural networks using Winograd's minimal filtering algorithms. The algorithms were derived for network layers with 3x3 as filter size for image recognition tasks. Their algorithms reduced arithmetic complexity up to 4 times compared with direct convolution, while using small block sizes with limited transform overhead and high computational intensity. Their work basically concentrated on the convolution layer and did not consider other layers of the CNN.

Abouelnaga et al.,(2016) studied the performance of different classifiers on the CIFAR-10 dataset, and build an ensemble of classifiers to reach a better performance. They were able to show that on CIFAR-10, K-Nearest Neighbors (KNN) and Convolutional Neural Network (CNN), on some classes, are mutually exclusive, and as such produced higher accuracy when combined. They applied the concept of Principal Component Analysis (PCA) to reduce overfitting in KNN, and then combined it with a CNN to increase its accuracy. Their work both KNN and CNN were used for feature extraction.

Xue et al., (2016) investigated a series of data augmentation techniques to progressively improve the prediction invariance of image scaling and rotation. They used SVM classifier as an alternative to softmax to enhance generalization ability. The recognition rate was up to 92.74% on the patch level with data augmentation and classifier boosting. Their results showed that the combined CNN-SVM model beats models of traditional features with SVM as well as the original CNN with softmax.

To deal with the problem associated with softmax classifier, (Zhou et al., 2017)proposed a new technique of combining Biomimetic Pattern Recognition (BPR) with CNNs for image classification. BPR performs class

recognition by a union of geometrical cover sets in a high-dimensional feature space and therefore can overcome some disadvantages of traditional pattern recognition. They evaluated the method using three image datasets: MNIST, AR, and CIFAR10.

Agarap, (2017), presented an architecture which combines a convolutional neural network (CNN) and a linear SVM for image classification. They employed the use of a simple 2-Convolutional Layer with max-pooling. They tested their architecture using Fashion-MNIST datasets and found that the CNN-softmax outperformed the CNN-SVM. They confessed that there was no enough preprocessing of the data in their study and they need to improve on their model to achieve more accurate results,

In their paper, Nguyen et al. (2017) proposed a presentation attack detection (PAD) method called Spoof Detection for near-infrared (NIR) camera-based finger-vein recognition system using convolutional neural network (CNN) to enhance the detection ability of previous handcrafted methods. They derived a suitable feature extractor for the PAD using CNN. They processed the extracted image features in order to enhance the presentation attack finger-vein image detection ability of the CNN method using principal component analysis method (PCA) for dimensionality reduction of feature space and support vector machine (SVM) for classification. Through extensive experimental results, they endorsed that their proposed method is suitable for presentation attack finger-vein image detection and it can deliver superior detection results when compared with other CNNs methods.

Ren et al., (2018) presented the use of fully convolutional architectures in the notable object detection systems such as Fast/Faster RCNN to replace the fully connected layer of CNNs. They derived a general formula specifically to accurately design the input size of the various fully convolutional networks in which the convolutional layer and pooling layer are concatenated with their strides and have proposed an efficient architecture of skip connection to accelerate the training process. They compared their model with Fast RCNN and the accuracy increased by about 2%. They tested their method using a very small data set. Again, using CNN as a classifier might not be feasible because fully connected layer is able to

generalize the feature extracted into the output space and also the output need to be scaler.

In their work, Zhang et al. (2018) proposed a k-Tree method to learn different optimal k values for different test and new samples, by involving a training stage in the KNN classification. In the training stage, k-Tree method first learns optimal k values for all training samples by a new sparse reconstruction model, and then constructed a decision tree using training samples and the learned optimal k values. In the test stage, the k-Tree obtained as output the optimal k value for each tested sample, before the KNN classification was carried out using the learned optimal k value and all training samples. Their model had similar running cost but higher classification accuracy, compared with traditional KNN methods but less running cost. It also realized similar classification accuracy, compared with the newly KNN methods, which assign different k values to different test samples.

Zhang et al. (2018) further proposed an improved version of k-Tree method called k*Tree method to speed up the test stage by extra storing the information of the training samples in the leaf nodes of kTree, such as the training samples located in the leaf nodes, their KNNs, and the nearest neighbor of these KNNs. The model was able to conduct KNN classification using a subset of the training samples in the leaf nodes rather than all training samples used in the newly KNN methods. This actually reduced running cost of test stage.

(Dang et al., 2018) used the powerful parallel computing ability of quantum computers to optimize the efficiency of image classification. Their scheme was based on quantum K Nearest-Neighbor algorithm. The complexity of the quantum algorithm was found to be superior to the classical algorithms. Moreover, the measurement step is executed only once to ensure the validity of the scheme. Their experimental results showed that, the classification accuracy is 83.1% on Graz-01 dataset and 78% on Caltech-101 dataset, which is close to existing classical algorithms and they concluded that their quantum scheme has a good classification performance and improved the efficiency.

In their work, Wang et al., (2019) proposed a SAR image recognition algorithm based on the CNN-ELM algorithm is proposed by combining the

CNN and the ELM algorithm. They conducted their experiment on the Moving and Stationary Target Acquisition and Recognition (MSTAR) database which contains 10 kinds of target images. The experiment result showed that their algorithm realized the sparsity of the network, improve the overfitting problem, and speed up the convergence speed of the network. They also mentioned that the running time of the experiment was very short and when compared with other experiment on the same database, it indicated that their experiment generated a higher recognition rate. Their experiment was carried out on moving images unlike ours which uses stationary images.

Lu et al., (2019) applied convolutional neural network to the detection of power network icing that could effectively classify and recognize power network icing image. They proposed a hybrid classification model combining convolution neural network and support vector machine. Their simulation results showed that it was feasible to use convolution neural network to classify the detection images of ice-covered power grid. They compared their model with the original CNN and reported that their model had better image classification effect.

Based on the literature studied, we discovered that most researchers tried to improve the performance of the CNN by adding SVM as classifier neglecting other types of classifiers. As such, this work tried to explore the potentiality of the KNN by adding it to the CNN, Furthermore, the KNN suffers dimensionality curse, which according to (Pal et al., 2016) is a major setback of the KNN. We proposed our model in order to improve the performance of the CNN as well as eliminate the dimensionality curse limitation of the KNN.

METHODOLOGY

The Proposed System

In this work, we are implementing the work earlier proposed by Abubakar & Boukari, (2018). We adopted the concept of replacing the softmax classifier with K-Nearest Neighbor (K-NN). We first use the CNN for feature extraction to produce filtered feature maps. We then replace the SVM of Agarap, (2017) with the K-NN at the fully connected layer for classification.

Architecture of the Proposed System

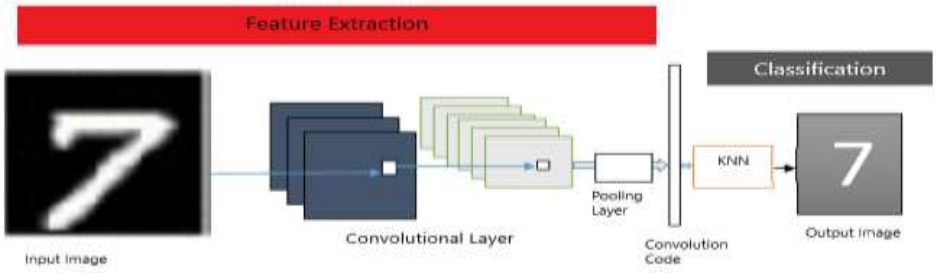

Figure 1: The Proposed Model Architecture.

A CNN architecture with alternating convolutional and max-pooling layers is used here because it propagates the maximum value within a receptive field to the next layer (Ranzato et al., 2007). The nodes in the output layer match one-character class. After using CNN for feature extraction, the extracted features are then fed into K-NN for classification.
To extract the CNN based feature, network training is used first, which is composed of two main procedures, namely, forward propagation and back-propagation (Yang, et al., 2015).

Forward Propagation

Suppose that we have some $M \times N$ input image, where M is the width and N the height. We train a convolutional filter ω of $m \times n$, our convolutional layer output will be of size$(M - m + 1) \times (N - n + 1)$.
To compute the pre-nonlinearity input to some unit x_{ij}^{l} in a layer, we need to sum up the contributions (weighted by the filter components) from the previous layer cells given by: $x_{ij}^{l} = \sum_{a=0}^{m-1} \sum_{b=0}^{n-1} \omega_{ab}\, y_{(i+a)(j+b)}^{l-1} \ \ldots (1)$
This is just a convolution. The convolutional layer then applies its nonlinearity:

$$y_{ij}^{l} = \sigma\left(x_{ij}^{l}\right) = \max(0, x) = \begin{cases} x, & if\ x \geq 0 \\ 0, & if\ x < 0 \end{cases} \ldots (2)$$

where $\sigma(x_{ij}^{l})$ is define by the ReLU activation function which is a linear identity for all positive values and zero for all negative values (Nwankpa et al., 2018).
Max-pooling is applied for down-sampling. The max-pooling layers are relatively simple, and do no learning themselves. They fundamentally

accept a $k \times k$ region and produce a single value, which is the maximum in the region. As such, given an input $M \times N$, max-pooling will output a $\frac{M}{k} \times \frac{N}{k}$, thus a $k \times k$ region is reduced to a single value.

Backward Propagation

We start by defining some error function, E with respect to each neuron output $\frac{\partial E}{\partial y_{ij}^{l}}$ applying chain rule. The chain rules states that the derivative of $f(g(x))$ is $f'\big(g(x)\big).g'(x)$.

We can then obtain the equation: $\frac{\partial E}{\partial \omega_{ab}} = \sum_{i=0}^{M-m} \sum_{j=0}^{N-n} \frac{\partial E}{\partial x_{ij}^{l}} \frac{\partial x_{ij}^{l}}{\partial \omega_{ab}}$ ... (3)

From equation (1) $\frac{\partial E}{\partial \omega_{ab}} = \sum_{i=0}^{M-m} \sum_{j=0}^{N-n} \frac{\partial E}{\partial x_{ij}^{l}} y_{(i+a)(j+b)}^{l-1}$ ... (4)

In this case, we must sum over all xℓij expressions in which ωab occurs.

In order to compute the gradient, we need to know the values of $\frac{\partial E}{\partial x_{ij}^{l}}$ and can be computed using equation (2) to yield:

$$\frac{\partial E}{\partial x_{ij}^{l}} = \frac{\partial E}{\partial y_{ij}^{l}} \frac{\partial y_{ij}^{l}}{\partial x_{ij}^{l}} = \frac{\partial E}{\partial y_{ij}^{l}} \frac{\partial}{\partial x_{ij}^{l}} \Big(\sigma\big(x_{ij}^{l}\big)\Big) = \frac{\partial E}{\partial y_{ij}^{l}} \Big(\sigma'\big(x_{ij}^{l}\big)\Big) \text{ ... (5)}$$

Where $\sigma'\big(x_{ij}^{l}\big)$ is the partial derivative of $\sigma\big(x_{ij}^{l}\big)$ with respect to x_{ij}^{l}.

There is need to also propagate errors back to the previous layers.

$$\frac{\partial E}{\partial y_{ij}^{l-1}} = \sum_{a=0}^{m-1} \sum_{b=0}^{n-1} \frac{\partial E}{\partial x_{(i-a)(j-b)}^{l}} \frac{\partial x_{(i-a)(j-b)}^{l}}{\partial y_{ij}^{l-1}}$$

$$= \sum_{a=0}^{m-1} \sum_{b=0}^{n-1} \frac{\partial E}{\partial x_{(i-a)(j-b)}^{l}} \omega_{ab} \text{ ... (6)}$$

where $\frac{\partial x_{(i-a)(j-b)}^{l}}{\partial y_{ij}^{l-1}} = \omega_{ab}$ from equation (3).

Classification using the K-Nearest Neighbor (KNN)

The classification rules of KNN are generated by the training samples themselves without any additional data. KNN classification algorithm predicts the test sample's category according to the K training samples which are the nearest neighbors to the test sample, and judge it to that category which has the largest category probability. The process of KNN algorithm to classify an image X according to Lihua et al., (2006) is:

Suppose that there are j training categories as $C1, C2, \dots, Cj$, and the sum of the training samples is N. After feature extraction, the image becomes k-dimension feature vector, we make image X to be the same vector form $(X1, X2, \dots, Xm)$ as the training samples. We then calculate the similarities between all training samples and image X, taking the ith image $di = (di1, di2, \dots, dim)$, the similarity $SIM(X, di)$ as

$$SIM(X, d_i) = \frac{\sum_{j=1}^{m} X_j . d_{ij}}{\sqrt{\sum_{j=1}^{m} X_j} . \sqrt{\sum_{j=1}^{m} d_{ij}}} \dots (7)$$

We choose k samples which are larger from N similarities of $SIM(X, di)$, $(i\ 1,2, \dots, N)$ and treat them as a KNN collection of X. Then, calculate the probability of X belong to each category respectively with the following formula.

$$P(X, C_j) = \sum_{d_j \epsilon KNN} SIM(X, d_i) . y(d_i, C_j) \dots (8)$$

Where, $y(d_i, C_j)$ is a category attribute function that satisfies:

$$y(d_i, C_j) = \begin{cases} 1, d_i \in C_j \\ 0, d_i \notin C_j \end{cases} \dots (9)$$

Image X is classified based on the category which has the largest $P(X, Cj)$.

System Design

In designing our model, the convolutional layers and pooling layers of the CNN are fixed while the full-connected layer of the CNN is replaced by the KNN. The topology of KNN is also shown as in Figure 3. In other words, the feature images trained by CNN are considered as the input of KNN, and the classification results are obtained through KNN. The role of KNN is to act as a classifier in CNNKNN model.

Datasets

The datasets to be used in this work are given in Table 1 below:

Table 1: Datasets

Dataset	Dataset Training Sample	Testing Sample	Class	Sample per Class	Properties
MNIST	60,000	10,000	10	6,000	28x28 grayscale
Fashion_MNIST	60,000	10,000	10	7,000	28x28 grayscale
CIFAR10	50,000	10,000	10	5,000	32x32 coloured

CIFAR100	50,000	10,000	100	500	32x32 coloured

Performance Metrics

The performance of the existing system and the proposed system was evaluated using the following performance metrics:

Training time

The training measures how long it took the model to go through and learn the training set. It is a determinant of how fast it takes to train a particular model on a dataset. It is calculated as the difference between the finish time and the start time. Mathematically, it can be calculated as:

$$training\ time = finish_{time} - start_{time} \quad \ldots (13)$$

This work was implemented using Python, Keras API with tensor flow backend environment in Anaconda 3. Other libraries include Numpy, Pandas, and Scikit Learn, which are most used open-source libraries intended for machine learning analysis.

RESULT AND DISCUSSION

System Testing

Preprocessing of the data before feature extraction is important. This includes resizing, reshaping, scaling of images and splitting of the dataset into batches. The images in the datasets are required to be in the same dimension because the same filter shall be used to extract features from them by sliding over in convolutional layer.

The filter used in CNNs have an attribute size, which is in relation to the number of weights used to extract the features of an area of the image. In the proposed hybrid model, we are used 5x5 kernel sizes. The first convolutional layer was defined to have 32 kernels and the second one has 64. The third layer has 128 nodes in a fully connected architecture. The fourth layer is composed of 10 nodes also fully connected. The last layer is a softmax operation.

The first step in order to build our model is to train the CNN using backward propagation algorithm. The ADAM optimizer (Kingma & Ba, 2015) was used in order to optimize the model that makes use of a cross-entropy loss function. After training the CNN, in order to construct the

model, the softmax layer was subsequently dropped so that the activations of the 128 nodes fully connected previous layer are used as the features extracted from a given image from the datasets. As such, instead of feeding the KNN classifier with the rows of large pixels values, the model is built by feeding the KNN classifier with the 128 higher level features extracted from the trained CNN.

The same process was followed in the implementation of the two existing models which were used to test the performance of our model.

Experimental Results

The experiments were carried out using four datasets on our model: the MNIST, the fashion MNIST, the CIFAR10 and the CIFAR100 datasets. On each dataset, ten experiments were conducted by training CNN (Lecun et al., 2015), CNN with SVM (Agarap, 2017) and CNN with KNN (our proposed model) and testing the models with the test sets from each dataset. We executed each experiment on the models ten times (epochs).

In the experiment, the original convolutional neural network, CNN, was used to extract the feature from the images and the proposed hybrid system that consists of using a KNN classifier on the 128 high-level features extracted by the conventional neural network is called CNNKNN. We executed each of the mentioned models ten times.

The results for the experimental results of the existing and proposed models are described below:

Comparison of the Models

In this section, we compared the means of the results of the three models on all the dataset. The table below depicts the mean of the values obtained:

Table 2: Mean Training Times(s) on all Datasets

Model/ Dataset	MNIST	Fashion_MNIST	CIFAR10	CIFAR100
CNN (Lecun et al., 2015)	399.77	346.58	345.61	343.74
CNNSVM (Agarap, 2017)	61.82	31.84	44.63	45.77
CNNKNN (our model)	40.67	30.39	42.63	43.90

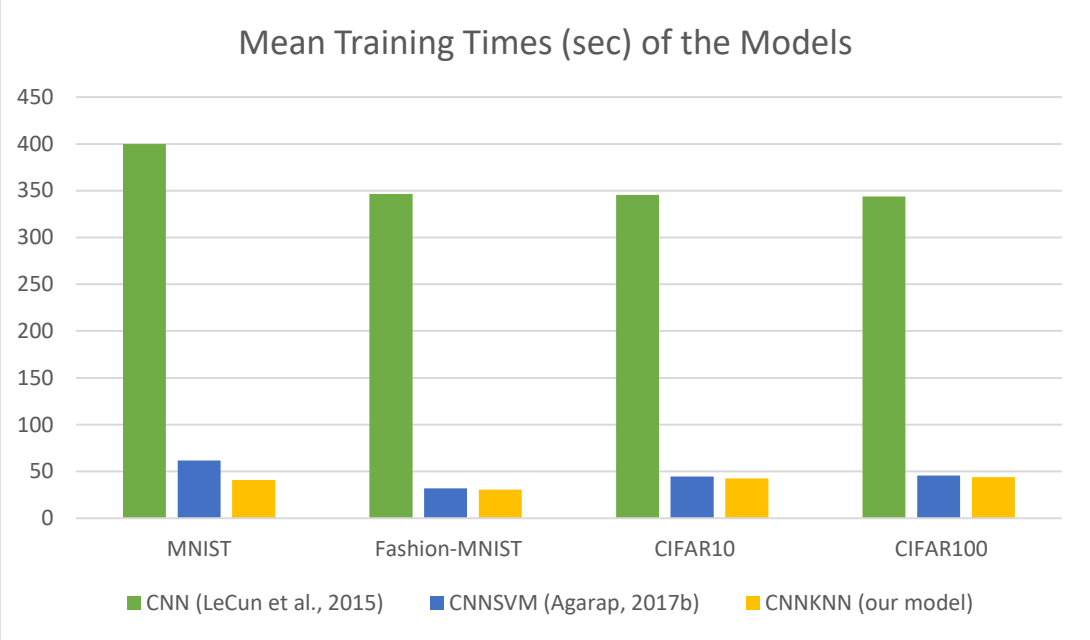

Figure 2: Training Time Performance of the Models on all Datasets.

Our model outperformed the others by obtaining the minimum training time, which implied that our model was fast in going through all the training images in all the datasets.

SUMMARY AND CONCLUSION

Summary

In this thesis, we proposed a model of classifying images based on the hybridization of convolutional neural network and k-nearest neighbor. We developed an improved faster classification model where the CNN was used for feature extraction and the 128 extracted features were classified using the KNN instead of the usual softmax. The model was using four different datasets, namely: MNIST, which consist of handwritten images of numbers; fashion_MNIST consisting of images of different types of clothing; CIFAR10 and CIFAR100 consisting respectively of images of different kinds. Furthermore, comparison was made with two existing models, the original CNN with softmax classifier and an improved CNNSVM model which is a combination of both CNN and SVM. The mean training times of the three models on the four dataset were 358.92s for CNN, 46.02s for CNNSVM and our model 39.42s.

Conclusion

This work had shown that a hybrid system composed of a CNN with KNN reduced the training time when compared with the original CNN. We were also able to show that K-nearest neighbor (KNN) can be used on large dataset as SVM.

The model CNNKNN demonstrated good performances on all the datasets. The use of feature extraction process helped in producing fast and accurate image classification. This indicated that extraction of characteristics is an excellent initial approach, with a very good compromise between performance and complexity.

As expected, the CNN model presents a good performance, but the proposed CNNKNN model is better. Again, the experiments suggest that learning high-level feature indeed improves the speed of a posterior classifier.

Since our model was faster in training the datasets. It was observed that CNNKNN improved the accuracy of the CNN. Therefore, our proposed hybrid system is shown to be a better classifier than the original CNN model.

Future work

This work could be extended to accommodate data of different types such as video, audio and text so as to enable KNN be more versatile when applied these datasets.

REFERENCES

Abouelnaga, Y., Ali, O. S., Rady, H., & Moustafa, M. (2017). CIFAR-10: KNN-Based Ensemble of Classifiers. Proceedings - 2016 International Conference on Computational Science and Computational Intelligence, CSCI 2016, 1192–1195. https://doi.org/10.1109/CSCI.2016.0225

Abubakar, F. A., & Boukari, S. (2018). A Convolutional Neural Network with K-Neareast Neighbor for Image Classification. IJARCCE, 7(12), 1–7. https://doi.org/10.17148/ijarcce.2018.71201

Agarap, A. F. (2017). An Architecture Combining Convolutional Neural Network (CNN) and Support Vector Machine (SVM) for Image Classification. 1992. http://arxiv.org/abs/1712.03541

Agarap, A. F. (2018). A Neural Network Architecture Combining Gated Recurrent Unit (GRU) and Support Vector Machine (SVM) for Intrusion Detection in Network Traffic Data. ACM International Conference Proceeding Series, 26–30. https://doi.org/https://doi.org/10.1145/3195106.3195117

Aggarwal, C. C., Hinneburg, A., & Keim, D. A. (2001). On the surprising behavior of distance metrics in high dimensional space. Database Theory — ICDT 2001, 1973, 420–434.

Arel, I.; Rose, D. C.; Karnowski, T. P. (2010). Deep Machine Learning-A New Frontier in Artificial Intelligent Research. IEEE Computational Intelligence Magazine, 5(4), 13–18.

Beyer, K., Goldstein, J., Ramakrishnan, R., & Shaft, U. (1999). When is 'nearest neighbor'meaningful? DatabaseTheory—ICDT'99, 1540, 217–235.

Biglarian, A., Hajizadeh, E., Kazemnejad, A., & Zali, M. (2011). Application of artificial neural network in predicting the survival rate of gastric cancer patients. Iranian Journal of Public Health, 40(2), 80–86. http://ovidsp.ovid.com/ovidweb.cgi?T=JS&CSC=Y&NEWS=N&PAGE=fulltext&D=prem&AN=23113076 http://digitaal.uba.uva.nl:9003/uva-linker?sid=OVID:medline&id=pmid:23113076&id=doi:&issn=2251-6085&isbn=&volume=40&issue=2&spage=80&pages=80-6&date=2011&title=Iranian+J

Chan, Y. -b., & Hall, P. (2009). Robust nearest-neighbor methods for classifying high-dimensional data. The Annals of Statistics, 37(6A), 3186–3203.

Chao, J. ., & Park, C. G. (2017). Additional feature CNN based automatic Target Recognition in SAR Image. Fourth Asian Coference on Defence Technology, 1–4.

Ciregan, D., Meier, U., & Schmidhuber, J. (2012). Multi-column deep neural networks for image classification. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 3642–3649. https://doi.org/10.1109/CVPR.2012.6248110

Ciresan, D., Meier, U., Masci, J., Gambardella, L. M., & Schmidhuber, J. (2011). Flexible, High Performance Convolutional Neural Networks for Image Classification. International Joint Conference on Artificial Intelligence (IJCAI) 2011, 1237–1242. https://doi.org/10.5591/978-1-57735-516-8/ijcai11-210

Dang, Y., Jiang, N., Hu, H., Ji, Z., & Zhang, W. (2018). Image classification based on quantum K-Nearest-Neighbor algorithm. Quantum Information Processing, 17(9), 1–18. https://doi.org/10.1007/s11128-018-2004-9

Ding, C., & Tao, D. (2018). Trunk-branch ensemble convolutional. IEEE Transactions on Pattern Analysis and Machine Intelligence, 40(4), 1002–1014.

Du, K.; Deng, Y.; Wang, R.; Zhao, T.; Li, N. (2016). SAR ATR Based on Displacement- and Rotation-Insensitive CNN. Remote Sensing Letters, 7(9), 895–904.

Fern, X. Z., & Brodley, C. E. (2003). Random projection for high dimensional data clustering: a cluster ensemble approach. Proceedings of the 20th International Conference on Machine Learning (ICML '03), 186–193.

Ghatak, A. (2017). Machine Learning with R. Springer. https://doi.org/https://doi.org/10.1007/978-981-10-6808-9

Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., Liu, T., Wang, X., Wang, L., Wang, G., Cai, J., & Chen, T. (2015). Recent Advances in Convolutional Neural Networks. 1–38. https://doi.org/10.3389/fpsyg.2013.00124

Hackeling, G. (2014). Mastering Machine Learning with scikit-learn.

Hafemann, L. G.; Sabourin, R.; Oliveira, L. E. S. D. (2016). Analyzing features learned for offline signature verification using deep CNNs. 23rd International Conference on Pattern Recognition (ICPR).

He, K., & Sun, J. (2015). Convolutional neural networks at constrained time cost. IEEE Conference on Computer Vision and Pattern Recognition (CVPR'15), 5353–5360. https://doi.org/10.1109/CVPR.2015.7299173

Huang, F.-J., & LeCun, Y. (2006). Large-scale learning with SVM and convolutional nets for generic object categorization. Proceedings. Computer Vision and Pattern Recognition Conference (CVPR'06), 284–291). https://doi.org/https://doi.org/10.1109/CVPR.2006.164

Huang, G.-B., Zhu, Q.-Y., & Siew, C.-K. (2004). Extreme learning machine: a new learning scheme of feedforward neural networks. IEEE International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541), 985–990.

Kang, S., Lee, J., Bong, K., Kim, Y., & Yoo, H.-J. (2018). Low-Power Scalable 3-d Face Frontialization Processor for CNN-Based Face Recognition in Mobile Devices. IEEE Journal on Emerging and SelectedTopics in Circuit and Systems, 8(4), 873–883.

Karpathy, A. (2016). Convolutional Neural Networks for Visual Recognition. Stanford University, 1–21. http://cs231n.github.io/classification/

Kingma, D. P., & Ba, J. L. (2015). Adam: A method for stochastic optimization. 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings, 12.

Krizhevsky, A., & Hinton, G. (2009). Learning Multiple Layers of Features from Tiny Images. cs.toronto.edu

Krizhevsky, Alex, Sutskever, I., & Geoffrey E., H. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Advances in Neural Information Processing Systems 25 (NIPS2012), 1–9. https://doi.org/10.1109/5.726791

Lavin, A., & Gray, S. (2016). Fast Algorithms for Convolutional Neural Networks. Proc. IEEE CVPR, 4013–4021. https://doi.org/10.1109/CVPR.2016.435

Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436–444. https://doi.org/10.1038/nature14539

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278–2323. https://doi.org/10.1109/5.726791

Leng, J., Tao, L., Gang, B., Dong, Q., & Han, D. (2016). Cube-CNNSVM: a novel hyperspectral image classification method. In IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI) (pp. 1027–1034). https://doi.org/10.1109/ICTAI.2016.0155

Li, J., Zhang, D., Zhang, J., & Zhang, J. (2017). Facial Expression Recognition with Faster R-CNN. Procedia - Procedia Computer Science, 107(Icict), 135–140. https://doi.org/10.1016/j.procs.2017.03.069

Li, W., Manivannan, S., Akbar, S., Zhang, J., Trucco, E., & McKenna, S. J. (2016). Gland segmentation in colon histology images using hand-crafted features and convolutional neural networks. International Symposium on Biomedical Imaging, 2016-June, 1405– 1408. https://doi.org/https://doi.org/10.1109/ISBI.2016.7493530

Lihua, Y., Qi, D., & Yanjun, G. (2006). Study on KNN Text Categorization Algorithm. Micro Computer Information, 21, 269–271.

Lillesand, T. M., Kiefer, R. W., & Chipman, J. W. (2004). Remote Sensing and Image Interpretation (5th ed.). Wiley.

Lu, C.-Y., H.Min, Gui, J., Zhu, L., & Lei, Y.-K. (2013). Face recognition via weighted sparse representation. Journal of Visual Communication and Image Representation, 24(2), 111–116.

Lu, J., Ye, Y., Xu, X., & Li, Q. (2019). Application research of convolution neural network in image classification of icing monitoring in power grid. Eurasip Journal on Image and Video Processing, 2019(1), 762–763. https://doi.org/10.1186/s13640-019-0439-2

Mao, X.; Hijazi, S.; Casas, R.; Kaul, P.; Rowen, C. (2018). Hierarchical CNN for traffic Sign Recognition. IEEE Intelligent Vehicle Symposium (IV), 130–135. https://doi.org/10.1109/IVS.2016.7535376

Meek, T., Barham, H., Beltaif, N., Kaadoor, A., & Akhter, T. (2016). Managing the Ethical and Risk Implications of Rapid Advances in Artificial Intelligence: A Literature Review. Engineering and Technology Management Faculty Publications and Presentations, 2017, 682–693.

Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. International Conference on Machine Learning (ICML), 807–814.

Nguyen, D. T., Yoon, H. S., Pham, T. D., & Park, K. R. (2017). Spoof detection for finger-vein recognition system using NIR camera. Sensors (Switzerland), 17(10). https://doi.org/10.3390/s17102261

Niu, X.-X., & Suen, C. Y. (2012). A novel hybrid CNN-SVM classifier for recognizing handwritten digits. Pattern Recognition, 45(4), 1318–1325.

Nugraheeni, R. A., & Mutijarsa, K. (2017). Comparative Analysis of Machine learning KNN, SVM and Random Forest Algorithm for Facial Expression Classification. International Seminar on Application for Technology of Information and Communication, 163–168. https://doi.org/https://doi.org/10.1109/ISEMANTIC.2016.7873831

Nwankpa, C. E., Ijomah, W., Gachagan, A., & Marshall, S. (2018). Activation Functions : Comparison of Trends in Practice and Research for Deep Learning. ArXiv:1811.03378v1, 1–20.

Pal, A. K., Mondal, P. K., & Ghosh, A. K. (2016). High dimensional nearest neighbor classification based on mean absolute differences of inter-point distances. Pattern Recognition Letters, 74, 1–8.

Palaniappan, R., Sundaraj, K., & Sundaraj, S. (2014). A Comparative study of SVM and KNN Machine Learning Algorithms for the Respiratory Diagnosis of Pathologies using Pulmonory Acoustic Signals. BMC Bioinformatics.

Peng, L., Liu, X., Liu, M., Dong, L., Hui, M., & Zhao, Y. (2018). SAR Target Recognition and Posture Estimation using Spatial Pyramid Pooling within CNN. International Conference on Optical Instruments and Technology: Optoelectronic Imaging/Spectroscopy and Signal Processing Technology, 27. https://doi.org/https://doi.org/10.1117/12.2285913

Radovanovi´c, M., Nanopoulos, A., & Ivanovi´c, M. (2010). Hubs in space: popular nearest neighbors in high-dimensional data. Journal OfMachine Learning Research (JMLR), 11, 2487–2531.

Ranzato, M. A., Huang, F. J., Boureau, Y., & LeCun, Y. (2007). Unsupervised learning of invariant feature hierarchies with applications to object recognition. IEEE Conference on Computer Vision and Pattern Recognition, 1–8.

Ren, Y. (2018). Object Detection Based on Fast / Faster RCNN Employing Fully Convolutional Architectures. 2018. https://doi.org/10.1155/2018/3598316

S. Deegalla, & Bostr¨om, H. (2006). Reducing high-dimensional data by principal component analysis vs. random projection for nearest neighbor classification. Proceedings of the 5th International Conference OnMachine Learning and Applications, (ICMLA'06), 245–250.

Sharma, D., & Kumar, N. (2017). A Review on Machine Learning Algorithms , Tasks and Applications. International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 6, Issue 10, October 2017, ISSN: 2278 – 1323 A, 6(10), 1549–1552.

Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. ImageNet Challenge, 1–10. https://doi.org/10.1016/j.infsof.2008.09.005

Singhal, S., Passricha, V., Sharma, P., & Aggarwal, R. K. (2019). Muti-level Region-of-Interest CNNs for End to End Speech Recognition. Journal of Ambient Intelligent and Humanized Computing, 10, 4615–4624.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 07-12-June, 1–9. https://doi.org/10.1109/CVPR.2015.7298594

Talwar, A., & Kumar, Y. (2013). Machine Learning: An artificial intelligence methodology. International Journal of Engineering and Computer Science, 2, 3400–3404.

Tan, T., Qian, Y., Hu, H., Zhou, Y., Ding, W., & Yu, K. (2018). Adaptive Very Deep Convolutional Residual Network for Noise Robust Speech Recognition. IEEE/ACM Transactions on Audio, Speech and Language Processing, 26(8), 1393–1405.

Tang, Y. (2013). Deep Learning using Linear Support Vector Machines. Deeplearning.Net. http://deeplearning.net/wp-content/uploads/2013/03/dlsvm.pdf

Wagner, R., Thom, M., Schweiger, R., & Rothermel, A. (2013). Learning Convolutional Neural Networks From Few Samples. 1884–1890.

Wang, P., Zhang, X., & Hao, Y. (2019). A Method Combining CNN and ELM for Feature Extraction and Classification of SAR Image. 2019.

Weinberger, K. Q., Blitzer, J., & Saul, L. K. (2006). Distance metric learning for large margin nearest neighbor classification. In Advances in Neural Information Processing Systems, 1473–1480.

Xiao, H., Rasul, K., & Vollgraf, R. (2017). Fashion-MNIST : a Novel Image Dataset for Benchmarking Machine Learning Algorithms. ArXiv:1708.07747, 2, 1–6.

Xie, S., & Hu, H. (2017). acial Expression Recognition with FRR-CNN. Electronic Letters, 53(4), 235–237.

Xu, B., Wang, N., Chen, T., & Li, M. (2015). Empirical Evaluation of Rectified Activations in Convolution Network. ICML Deep Learning Workshop, 1–5.

Xue, D., Zhang, R., & Wang, H. F. Y. (2016). CNN-SVM for Microvascular Morphological Type Recognition with Data Augmentation. Journal of Medical and Biological Engineering, 36(6), 755–764. https://doi.org/10.1007/s40846-016-0182-4

Yang, J., Gao, J., Wang, G., & Zhang, S. (2015). Natural Scene Recognition Based on Superpixels and Deep Boltzmann Machines. ArXiv:1506.07271 [Cs], 100, 2369–2374. http://arxiv.org/abs/1506.07271%5Cnhttp://www.arxiv.org/pdf/1506.07271.pdf

Yu, D., Wang, H., Chen, P., & Wei, Z. (2014). Mixed pooling for convolutional neural networks. Lecture Notes in Computer Science (Including Subseries Lecture Notes in

Artificial Intelligence and Lecture Notes in Bioinformatics), 8818, 364–375. https://doi.org/10.1007/978-3-319-11740-9_34

Zeiler, M. D., & Fergus, R. (2014). Visualizing and Understanding Convolutional Networks BT - Computer Vision – ECCV 2014. In Computer Vision – ECCV 2014 (Vol. 8689, Issue Chapter 53, pp. 818–833). https://doi.org/10.1007/978-3-319-10590-1_53

Zhang, S., Li, X., Zong, M., Zhu, X., & Wang, R. (2018). Efficient kNN classification with different numbers of nearest neighbors. IEEE Transactions on Neural Networks and Learning Systems, 29(5), 1774–1785. https://doi.org/10.1109/TNNLS.2017.2673241

Zhao, H., & Liu, H. (2019). Multiple classifiers fusion and CNN feature extraction for handwritten digits recognition. Granular Computing. https://doi.org/10.1007/s41066-019-00158-6

Zhou, L., Li, Q., Huo, G., & Zhou, Y. (2017). Image Classification Using Biomimetic Pattern Recognition with Convolutional Neural Networks Features, Computational Intelligence and Neuroscience. 2017, 1-12. http://doi.org/10.1155/2017/3792805

Zhu, X., Zhang, L., & Z, H. (2014). A sparse embedding and least variance encoding approach to hashing. IEEE Trans. Image Process.