Performance Analysis of Convolutional Neural Network Models on Numerical Value Prediction of Modified MNIST Dataset

Christopher Zheng[1], Qingchuan Ma[2] and Haoxuan Shi[3]
GROUP **58**

***Abstract*— Digit recognition is a heated application of modern day computer vision. In this work, we explore various machine learning models and deep learning models to predict the numerically greatest values in images where three arbitrary digits inhabit. We compare and contrast both the prediction accuracy and training time of these models. Besides, we also examine the impacts of different image preprocessing techniques. As a discovery, raw image data without preprocessing in fact leads to the best result given a well-tuned convolutional neural network architecture. Our best model is an ensemble model that merges multiple convolutional neural networks and makes predictions based on a hard-vote scheme. This model is proved to be accurate and robust as it achieves 97.066% accuracy on Kaggle.**

I. INTRODUCTION

Digit recognition is a classic problem in the field of image classification, the goal of which is to locate digits that appear in an image and categorize it as one of the digits from zero to nine based on its interpreted value or shape. This technique is highly applicable in real-world scenarios such as automatic cheque deposits and document identification.

A. Task Description

In this work, dealing with a modified MNIST dataset, we aim to perform digit recognition using machine learning and deep learning models. Our goal is to construct a model that learns from gray-scale images, each of which contains three handwritten digits and its corresponding label, and then predicts the value of the numerically largest number for every incoming image with three digits.

B. Important Findings

We implement three types of preprocessing strategies and the least complicated one demonstrates the best performance. We compare and contrast the performance of various models including traditional machine learning models (Naive Bayes [1] and k-Nearest Neighbors [2]) as well as several deep learning models (LeNet [3], VGG-16 [4] and its variant). We discover that, although machine learning algorithms are capable of dealing with digit recognition to some extent, deep learning approaches tend to produce more promising results despite the higher training costs.

Among all these models, our best-performed model is an ensemble model that combines all models' votes and decides final predictions. It achieves an accuracy score of 97.066% on Kaggle Leaderboard.

II. BACKGROUND AND RELATED WORK

A. Background

Keeping pace with the continuous rise of computational power of hardware, deep learning algorithms and models are becoming progressively popular and demonstrating promising performance. Common categories of neural networks include Feed-Forward Neural Network (FFNN, [5]), Recurrent Neural Network (RNN, [6]), and Convolutional Neural Network (CNN, [7]). FFNN is the first and ordinary structure of an artificial neural network. Besides, RNN belongs to a class of artificial neural networks where connections between nodes can form a directed graph along a sequence. RNNs can use their internal state, i.e. memory, to process sequences of inputs, and thus are applied to tasks such as speech recognition [8]. Long Short-Term Memory (LSTM), one of the most salient representative of RNNs, is widely used in time series trend prediction [9] and other scientific domains.

Convolutional Neural Networks (CNNs) are increasingly applicable and prevalent in both the industrial field and academia of computer vision. Particularly, CNNs are widely implemented to accomplish tasks including object localization, object detection, image segmentation, image captioning, autonomous driving, etc [10].

[1]Christopher Zheng (260760794, Corresponding Author): christopher.zheng@mail.mcgill.ca
[2]Qingchuan Ma (260740719): qingchuan.ma@mail.mcgill.ca
[3]Haoxuan Shi (260779480): haoxuan.shi@mail.mcgill.ca

B. Related Work

The success of AlexNet [11] draws the world's attention to CNNs. In the ImageNet [12] Large Scale Visual Recognition Challenge (a visual competition for object competition), Alex Krizhevesky et al. propose a quite large and deep network which achieves the state-of-the-art accuracy of 2012 against all the previous computer vision techniques and machine learning algorithms. This model, however, suffers relatively long training time and computation cost due to its many fully connected layers, and therefore, is not commonly utilized. Nonetheless, AlexNet triggers the mania of deep learning and its successors tend to outperform it on classification of MNIST, namely GoogleNet [13], VGGNet [4], ResNet[14], etc. Further, they are proved to be both accurate and robust on even larger datasets, for example, CIFAR-10 [15] and other more complicated tasks on ImageNet.

In terms of CNNs, the hyper-parameters can impact the performance and result of an arbitrary CNN model to a very large extent [16]. Take an example to elaborate, a study focuses on the influence of *batch size* on the accuracy of image recognition validated on MNIST and CIFAR-10, and concludes that the batch size parameter has a crucial effect on the recognition accuracy [17]. Apart from that, some research also suggests that the value of the learning rate can greatly influence the performance of a CNN model [11].

Beside the currently popular deep learning methods, certain traditional machine learning algorithms are also implemented and applied to some computer vision tasks. For instance, proposed by Simon Bernard [18] in 2007, Random Forests [19] can be used to achieve a fairly excellent recognition accuracy score (approximately 93%) in MNIST digit classification, although this accuracy is still not comparable to what deep learning can produce.

III. Dataset and Setup

A. Dataset

In this work, the provided modified MNIST dataset for training consists of 50,000 gray-scale images, each of which contains three handwritten digits and is of 128 by 128 in terms of dimensionality. As shown in Fig. 1, the data distribution is relatively imbalanced because larger numbers tend to dominate in evaluation and thus are more possible to be the label values. To offset the impact of this issue, we intentionally augment the dataset by repeating the data entries whose labels are minor in number and over-sample them so that

Fig. 1: The original distribution of the quantities of number values in the provided training set.

Fig. 2: The ideal distribution of the quantities of number values after oversampling and data augmentation.

ideally the data distribution can be as in Fig. 2. To avoid overfitting, we split the training and testing sets with respect to a ratio of 0.2, i.e., 80% image entries for training and 20% for validation. Labels indicating the largest numeric values among the sets of three digits for all images are stored and provided in a separate *CSV* file.

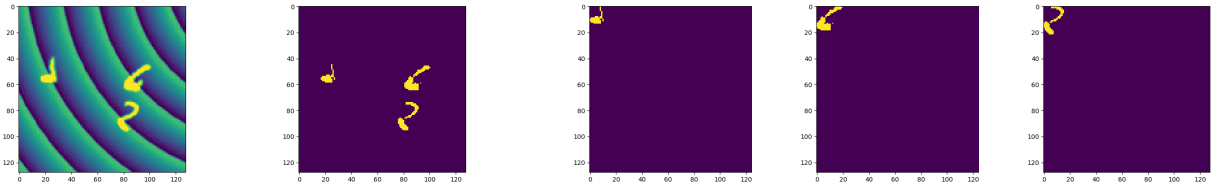
We are also given a dataset of 10,000 images that are used as test set, and our Kaggle competition model is tested on this set of images. Labels for test set samples are not given.

B. Preprocessing

We attempt the three following image data preprocessing methods respectively and examine their impacts on the overall result. We choose only one of these methods to prepare the data for the best deep learning model.

1) Zero preprocessing: As the name suggests, this strategy is to directly utilize the raw images' gray-scale values for later training. An example of running this type of preprocessing is shown in Fig. 3(a).

2) Simple preprocessing: To make the digits and the background appear to be more distinct, we convert the gray-scale values of background pixels all to zero and cast all digits' corresponding values to 255 (the maximum pixel value). An example of running this type of preprocessing is shown in Fig. 3(b).



(a) Zero preprocessing (b) Simple preprocessing (f) Complete preprocessing

Fig. 3: Sample output of three preprocessing methods.

3) Complete preprocessing: This preprocessing method is to separately consider individual digits. We deploy the K-Means clustering algorithm [20] to group the pixels into three clusters which represent the three numbers. Subsequently, we crop the three clusters out of the original picture and normalize them with respect to a uniform scale (128 by 128) so that ideally each picture comprises only one handwritten digit. Complete preprocessing does require that we extract three digits out of test images as well so that our model may determine the values of the three and predict the ones with the largest numerical values. An example of running this preprocessing is shown in Fig. 3(c).

IV. Proposed Approach

We aim to properly combine multiple deep learning models into an *ensemble* model to achieve better classification performance. To start with, we briefly introduce two traditional machine learning models for comparison and our *base models*, i.e. the models that we utilize as the building blocks of the final predictor. In this section, for all the models, we present their best hyper-parameters tuned using Grid Search [21] or 5-fold cross validation and their individual classification performance.

A. Traditional Machine Learning Base Models

As discussed in Section II-B, certain conventional machine learning algorithms can still be applicable to various tasks despite the widely acknowledge superiority of deep learning. We present the base models originating from traditional machine learning techniques that we compare against deep learning methods on this digit recognition task.

1) Naive Bayes: The Multinomial [1] Naive Bayes classifier takes input that represents the pixel values are generated with respect to a multinomial distribution. We flatten the matrix representation of each picture so that the classifier receives 16384 (128^2) features. Based on past literature, this model is commonly selected for classification [22]. The α value of the Lidstone smoothing is 1 (Laplace smoothing).

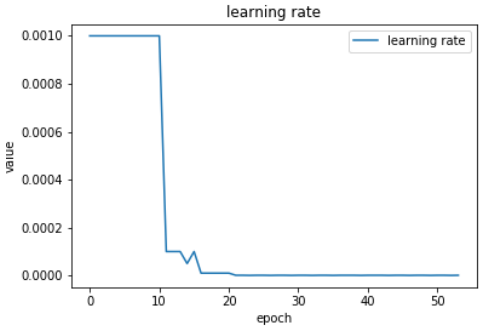


Fig. 4: Learning rate (initialized to be 10^{-3}).

2) k-Nearest Neighbors (kNN): kNN [2] is an instance-based method. It stores all the training examples and when given a test sample, the model finds the k training sample points (in this case, k = 5) that are *closest* to this test example. The predicted result of this test sample is the majority class of the five nearest training samples' labels.

B. Convolutional Neural Network Base Models

CNN is the main focus of this work and we experiment on several CNN architectures with various aforementioned preprocessing methods. All of their configuration units are detailed in Table I. For all the CNN models that we practice, we take advantage of Keras Library [14]. Peculiarly, we use the criterion of Categorical Cross Entropy and Adam Optimizer [23], and we also place three callbacks, i.e., *ReducedLROnPlateau*, *EarlyStopping*, and *LearningRateScheduler*, to dynamically mutate the learning rate during training process to search for a global optimum (shown in Fig. 4) and intelligently stop the process when the validation accuracy demonstrates a tendency of decay.

1) LeNet-5: LeNet-5 [3] is one of the best-known CNN architectures and has a fairly simple structure which renders it quite training efficient. LeNet-5 comprises 5 layers, including 2 convolutional layers, 2 sub-sampling layers and 1 fully connected layer. The result of this CNN, however, is not promising due to, we suspect, the overly simplistic layer design of the model for this digit recognition task which is quite complicated. Please note that we do not embed any dropout nor batch normalization in this architecture.

2) VGG-16 Modified: We modify the original VGG-16 model, a quite simple structure (3 by 3 convolutional layers), so that it suits our scenario. We underscore some major changes as follows.

- In the modified version, we use two convolutional layers with depth 32, two with depth 64, four with depth 64, and 2 with depth 128. We delete the layers with depth of 256.
- Besides, we utilize only one fully connected layer rather than two as in the VGG-16 model.
- The dimension of the only fully connected layer is 1024 and is smaller than those of the layers in the original VGG-16.
- Batch normalization and dropout are applied immediately following the convolutional layers.

3) VGG-16 Modified 2: We observe several common structures of CNNs and develop this following architecture based on the previously modified VGG-16. It possesses six convolutional layers with depth 64 and two with depth 128. Remarkably, it has two following fully connected layers with 1024 units for each.

C. Ensemble

We follow the *Stacking* ensemble method [24]. Our meta classifier controls the voting procedure and is responsible for generating final predictions. The meta classifier is constructed based on a *hard-vote* scheme, i.e., given an image, each model has one vote and the mode of all votes is selected as the final prediction.

V. Results

Since our best model is an ensemble model, we take one of the CNN base models, our modified version of VGG-16 (IV-B.2), to illustrate how our model develops with training. As we can see in Fig. 5, as the epoch proceeds, both the training set accuracy and validation accuracy boost drastically initially, then slow down with a lower increasing rate and eventually converge after the tenth epoch. In contrast, the value of the cross entropy loss displays a mirrored behavior in Fig.6.

TABLE I: The configuration table of the three convolutional neural networks in our model.

ConvNet Configuration		
Input: 128 by 128 Gray-Scale Image		
LENET-5	VGG-16 MODIFIED	VGG-16 MODIFIED 2
CONV5-6	CONV3-32 CONV3-32	CONV3-64 CONV3-64
AVGPOOL	MAXPOOL	MAXPOOL
CONV5-16	CONV3-64 CONV3-64	CONV3-64 CONV3-64
AVGPOOL	MAXPOOL	MAXPOOL
CONV5-120	CONV3-128 CONV3-128	CONV3-64 CONV3-64
	MAXPOOL	MAXPOOL
	CONV3-128 CONV3-128	CONV3-128 CONV3-128
	MAXPOOL	MAXPOOL
FC-84	FC-1024	FC-1024 FC-1024
SOFTMAX		

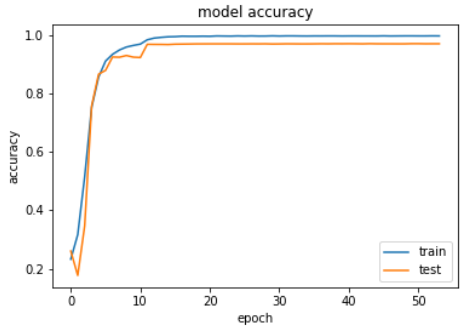


Fig. 5: Training set accuracy and validation set accuracy of VGG-16 modified.

The learning rate is a crucial hyper-parameter. Separately, the learning rate for this above base model is preset to 10^{-3} and our *LearningRateScheduler* mentioned in IV-B controls the learning rate to achieve the global optimum as shown in Fig. 4. Again, take the VGG-16 modified model as an example. In Fig. 7, it is clear that the value of the learning rate can to a large extent influence the performance of a model.

Table II displays the final result of all the models we train. (Please note that this list is not exhaustive and we only present the best result and statistics for each model due to the limit of space.) Remarkably, all deep learning architectures surpass selected traditional machine learning models to a very large extent. Besides, as described in Section IV, our ensemble model combines predictions of three deep learning models, i.e., LeNet-5, VGG-16 Modified, and VGG-16 Modified 2, the validation accuracy scores of which

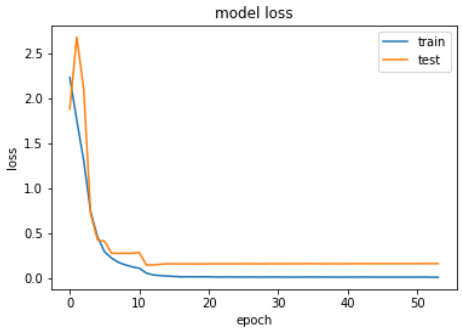


Fig. 6: Cross entropy loss of training set and validation set of VGG-16 modified.

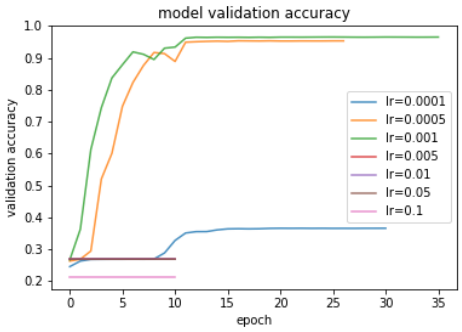


Fig. 7: Validation accuracy of VGG-16 modified with different learning rates. (Some curves die out half way through because of *EarlyStopping*.)

are 68%, 96% and 96%, respectively. The ultimate emsemble boosts the accuracy by one more percent to 97% despite the long training time which is negligible since the training is offline[1].

VI. Discussion

Admittedly, certain parts of this work are not exhausted nor complete, and some procedures are not fully scientifically sound. As the future work, we aim to explore more types of structures of neural networks. For example, we may study about more CNN models such as ResNet and other categories of recurrent neural networks such as LSTM, and examine their performance on digit recognition. Currently, we are unable to implement such models in this project due to the limited computing resources. Next, we also expect to further research on other types of ensemble methods such as Bagging and Boosting. Eventually, we might replicate the models and evaluate and analyze their performance on other mainstream deep learning applications including speech recognition, objection localization, etc.

[1]Implementation specs: models are implemented in Python 3.7.3 and our experiments are conducted on Google Colaboratory with 25.5 GB RAM and GPU.

TABLE II: Final result: performance comparison of our models after fine-tuning.

Model	Train Acc	Test Acc	Training Time (h)	Epoch of Con.	Prep.
LeNet-5	0.72	0.68	0.94	5	Simple
VGG-16 Modified	0.99	0.96	1.18	31	Zero
VGG-16 Modified 2	0.99	0.96	1.58	40	Zero
Ensemble	/	**0.97**	3.70	/	Zero
Naive Bayes	0.29	0.29	0.51	/	Zero
kNN	0.27	0.27	0.79	/	Zero

VII. Conclusion

In this work of digit recognition, we examine a variety of machine learning models and deep learning models to predict the numerically greatest values in images where three arbitrary digits inhabit. We evaluate these models according to their recognition accuracy and model training time. Besides, we also research on the impacts of the three image preprocessing techniques we propose. As an important finding, raw image data without preprocessing in fact leads to the best result given a well-tuned convolutional neural network architecture. Our ensemble model that merges multiple convolutional neural networks and makes predictions based on a hard-vote scheme is the best model that we construct. This model has well-proved accuracy and robustness as it achieves 97.066% accuracy on Kaggle.

VIII. Statement of Contributions

- **Define the question:** Done as a group.
- **Develop the methodology:** Done as a group.
- **Data preparation and preprocessing:** Qingchuan
- **Data analysis:** Christopher
- **Implement the solution code:**
  - Image preprocessing: Qingchuan
  - Architecture of LeNet-5: Christopher and Haoxuan.
  - Architecture of VGG-16 and its variants: Christopher and Haoxuan.
  - CNN Comparison experiments: Christopher
  - The meta classifier: Christopher.
  - Naive Bayes and kNN: Qingchuan.
- **Writing up the report:** Christopher

REFERENCES

[1] A. M. Kibriya, E. Frank, B. Pfahringer, and G. Holmes, "Multinomial naive bayes for text categorization revisited," in *Australasian Joint Conference on Artificial Intelligence*, pp. 488–499, Springer, 2004.

[2] J. M. Keller, M. R. Gray, and J. A. Givens, "A fuzzy k-nearest neighbor algorithm," *IEEE transactions on systems, man, and cybernetics*, no. 4, pp. 580–585, 1985.

[3] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *Advances in neural information processing systems*, pp. 1135–1143, 2015.

[4] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[5] D. Svozil, V. Kvasnicka, and J. Pospichal, "Introduction to multi-layer feed-forward neural networks," *Chemometrics and intelligent laboratory systems*, vol. 39, no. 1, pp. 43–62, 1997.

[6] T. Mikolov, M. Karafiát, L. Burget, J. Černockỳ, and S. Khudanpur, "Recurrent neural network based language model," in *Eleventh annual conference of the international speech communication association*, 2010.

[7] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, "Face recognition: A convolutional neural-network approach," *IEEE transactions on neural networks*, vol. 8, no. 1, pp. 98–113, 1997.

[8] Y. Miao, M. Gowayyed, and F. Metze, "Eesen: End-to-end speech recognition using deep rnn models and wfst-based decoding," in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pp. 167–174, IEEE, 2015.

[9] F. A. Gers, D. Eck, and J. Schmidhuber, "Applying lstm to time series predictable through time-window approaches," in *Neural Nets WIRN Vietri-01*, pp. 193–200, Springer, 2002.

[10] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[12] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.

[13] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[15] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," tech. rep., Citeseer, 2009.

[16] F. Radenović, G. Tolias, and O. Chum, "Fine-tuning cnn image retrieval with no human annotation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 7, pp. 1655–1668, 2018.

[17] P. M. Radiuk, "Impact of training set batch size on the performance of convolutional neural networks for diverse datasets," *Information Technology and Management Science*, vol. 20, no. 1, pp. 20–24, 2017.

[18] S. Bernard, S. Adam, and L. Heutte, "Using random forests for handwritten digit recognition," in *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, vol. 2, pp. 1043–1047, IEEE, 2007.

[19] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[20] A. Likas, N. Vlassis, and J. J. Verbeek, "The global k-means clustering algorithm," *Pattern recognition*, vol. 36, no. 2, pp. 451–461, 2003.

[21] C.-W. Hsu, C.-C. Chang, C.-J. Lin, *et al.*, "A practical guide to support vector classification," 2003.

[22] J. D. Rennie, L. Shih, J. Teevan, and D. R. Karger, "Tackling the poor assumptions of naive bayes text classifiers," in *Proceedings of the 20th international conference on machine learning (ICML-03)*, pp. 616–623, 2003.

[23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[24] F. Divina, A. Gilson, F. Goméz-Vela, M. García Torres, and J. Torres, "Stacking ensemble learning for short-term electricity consumption forecasting," *Energies*, vol. 11, no. 4, p. 949, 2018.