Assessing four Neural Networks on Handwritten Digit Recognition Dataset (MNIST)

**Abstract**—Although the image recognition has been a research topic for many years, many researchers still have a keen interest in it[1]. In some papers[2][3][4], however, there is a tendency to compare models only on one or two datasets, either because of time restraints or because the model is tailored to a specific task. Accordingly, it is hard to understand how well a certain model generalizes across image recognition field[6]. In this paper, we compare four neural networks on MNIST dataset[5] with different division. Among of them, three are Convolutional Neural Networks (CNN)[7], Deep Residual Network (ResNet)[2] and Dense Convolutional Network (DenseNet)[3] respectively, and the other is our improvement on CNN baseline through introducing Capsule Network (CapsNet)[1] to image recognition area. We show that the previous models despite do a quite good job in this area, our retrofitting can be applied to get a better performance. The result obtained by CapsNet is an accuracy rate of 99.75%, and it is the best result published so far. Another inspiring result is that CapsNet only needs a small amount of data to get the excellent performance. Finally, we will apply CapsNet’s ability to generalize in other image recognition field in the future.

**Index Terms**—Neural Network, CNN, CapsNet, DenseNet, ResNet, MNIST.

1. **INTRODUCTION**

Motivated by the development of artificial intelligence, there has been a good amount of progress in image recogni- tion over the past 10 years, including the proposal of many new models and the creation of benchmark datasets.

In some papers, however, there is a tendency to compare models only on one or two datasets, either because of time restraints or because the model is tailored to a specific task. Accordingly, it is hard to understand how well a certain model generalizes across image recognition field.

In this paper, our main contributions are, therefore, comparing four mainstream image recognition models on MNIST dataset with different division. Among of them, three are CNN, ResNet and DenseNet respectively. These three models are already proved to have good performance in image recognition, and we summarize the characteristics of these models in Section 2. In addition, we find that the standard CNN model still exists some drawbacks. Accord- ingly we use CNN as a baseline model, and improve it through applying CapsNet to optimize on this basis. It is the fourth model and is described in detail in Section 3. We use the MNIST dataset for the test because the recognition of handwritten digits is a topic of practical importance. This object has continued to produce much research effort in recent years for several reasons. First, standard benchmark datasets like MNIST exist that make it easy for us to obtain result. Second, many publications and techniques are avail- able that can be cited and built on, respectively. In order to make the model more generalizable in the field of image recognition, we randomly divided the MNIST dataset into 25%, 50%, 75%, and 100% to test.

Ultimately, we contribute to a better understanding of

the performance of different model architectures on MNIST dataset. Consequently, we detect that CapsNet is the best overall model, which outperforms the other models on all tasks and consistently beats the baseline. The results of experiment are presented in Section 4, and conclusion

follows in Section 5.

Multi- Multi-task learning(Caruana (1998)) enjoys the idea of pooling information that can be learned from data collected for multiple connected tasks. Multiple sources of knowledge will stem from multiple datasets, or even one dataset, for multiple tasks. during this work, we have a tendency to concentrate on the case of exploitation multiple datasets for multiple tasks. Namely, we use MNIST image datasetscollectedfordigitrecognition,fashionitemrecognition,andletterrecognition,respectively. data sharing in multi-task coaching is achieved in numerous formality. For neural-network based mostly deep learning, the sharing will happen at the input layer, the hidden layers, or the output layer. Input-layer multi-tasking combines heterogeneous computer file, hidden-layer multi-tasking shares multiple teams of hidden layer units, and output-layer multi-tasking pools multiple output teams of classes. The implementation of a multi-task learning system depends on the info and also the tasks at hand. Multi-task learning has been with success applied to several applications of machine learning, from linguistic communication processing (Collobert (2008)) and speech recognition (Deng et al. (2013)) to laptop vision (Ren et al. (2015)) and drug discovery (Ramsundar et al. (2015)). A recent review of multi-task learning in deep learning is found in (Ruder (2017))

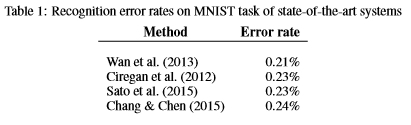
1. **RELATED WORKS**

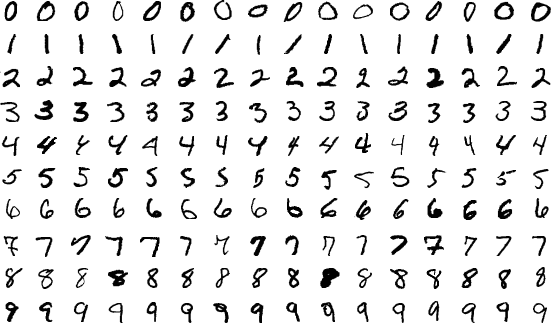
This section describes MNIST dataset which will be used in the experiments and then discusses the characteristics of the three neural network models.

#### Dataset

The MNIST dataset (LeCun et al. (1998)) consists of a coaching set of sixty thousand images, and a check set of ten thousand images. MNIST is commonly noted because the drosophila of machine learning, because it is a perfect testbed for brand new machine learning theories or ways on real-world knowledge. Table one lists the progressive performance on MNIST dataset. a couple of samples of MNIST dataset area unit shown in Table a pair of, alongside with samples of the opposite MNIST-like datasets.

The MNIST dataset is from the National Institute of Stan- dards and Technology (NIST). The training set consists of handwritten numbers from 250 different people, of which 50% are high school students and 50% are from the Census Bureau. The test set is also the same proportion of hand- written digital data. MNIST dataset totally contains 60,000 images in the training set and 10,000 patterns in the testing set, each of size 2828 pixels with 256 gray levels[8]. The dataset can be downloaded online and some examples from the MNIST corpus are shown in Fig. [1.](#_bookmark0)





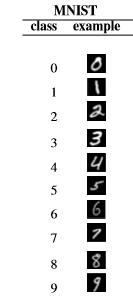
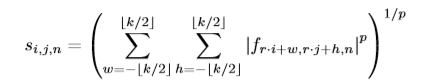


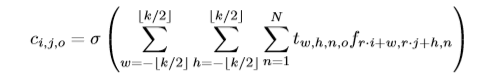
Fig. 1. Example images from the MNIST dataset, including 60,000 images in the training set and 10,000 patterns in the testing set.

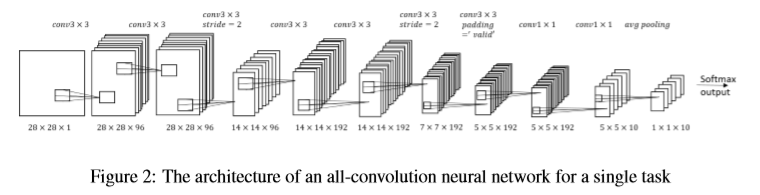
#### ALL-CONVOLUTION NEURAL NETWORKS

A convolutional neural networks(CNN) (LeCunetal.(1998))usually consists of alternating convolution layers and pooling layers. In distinction, all-convolution neural network (Springenberg et al. (2014)) replace pooling layers by convolution layers. A feature map created at a convolution layer may be pictured by a three-dimensional tensor of order W ×H ×N, wherever W and H are the dimension and height, N is that the variety of channels. A pooling operation with a sizeof k×k and a stride of r applied to a three-dimensional tensor leads to another tensor s with



where the ﬁrst a pair of subscripts index position within the map, and therefore the third subscript indexed the channel. If a convolution operation with constant stride were applied to f, we'd have a tensor c with 

  
  
where t is that the convolution kernel tensor, and σ(·) is AN activation operate. Thus, a pooling operation are often seen as a convolution operation with uniform kernel tensor and with L^p -norm because the activation function.

The design of all-convolution neural network for one task is shown in Figure . The multi-task learning classiﬁer has constant design as a single-task classiﬁer except that the dimension of the output layer is proportional to the quantity of tasks. The target label is increased accordingly  
   
by zero-padding. As mentioned before, we have a tendency to mix the coaching knowledge along, in order that the coaching label dimension is redoubled from [1 \* 10] to [1 \* 20] by zero-padding.  Multi-task learning with totally different combination of datasets area unit evaluated, particularly three bi-task learning and one tri-task learning.

#### CNN

In machine learning, CNN is a feed-forward artificial neu- ral networks, most commonly applied to analyzing visual imagery. For CNN, the earliest date can be traced back to the 1986 BP algorithm[9]. Then in 1989 LeCun used it in multi-layer neural networks[10]. Until 1998, LeCun pro- posed the LeNet-5 model, and the neural network prototype was completed. CNN consists of one or more convolutional layers and the top fully connected layer, and it also includes associated weights and a pooling layer. This structure allows the convolutional neural network to take advantage of the two dimensional structure of the input data, so it can give very good results in image recognition[11]. So we try to apply it to the MNIST dataset for testing.

#### ResNet

Deep convolutional neural networks have led to a series of breakthroughs for image classification. However, when deeper networks are able to start converging, a degradation problem[12] has been exposed: with the network depth in- creasing, accuracy gets saturated and then degrades rapidly. Therefore, ResNet is presented in 2017. It can reduce the train error while deepening the depth of the network, and solve the problem of gradient dispersion[13], improving network performance, which is shown in the Eq. [1.](#_bookmark1) Most importantly, ResNet can not only be very deep, but also has a very simple structure. It is a very small single module piled up, its unit module block as shown in Fig. [2.](#_bookmark2)

#### DenseNet

In the field of image recognition, CNN has become the most popular method. A milestone in the history of CNN is the emergence of the ResNet model[14]. ResNet can train deeper CNN models to achieve higher accuracy. The basic idea of the DenseNet model is the same as that of ResNet, but it establishes a dense connection between all previous and subsequent layers[15]. Its other major feature is feature reuse through the connection of features on the channel. Therefore we also tested its performance on the MNIST dataset.

1. **EXPERIMENTAL SETUP**

We compare four models, three of which are mentioned in Section 2. The other is our retrofitting and improvement based on CNN model. It is described in detail in Section 4.2 and 4.3. We use the MNIST datasets mentioned in Section

2.1 to test these models.

We tested all the models using a workstation built from commodity hardware: dual GeForce GTX 1080 graphics cards, an i7-6800K CPU, and 64 GB of RAM. Our implemen- tation is in TensorFlow and we use the Adam optimizer with TensorFlow default parameters, including the exponentially decaying learning rate, to minimize the sum of the margin losses.

#### Baseline

Our model is based on a standard CNN with three con- volutional layers, which is demonstrated to achieve a low test error rate on MNIST. The channels of three layers are

*×*

*xl* = *Hl*(*x*

*l−*1

) + *x*

*l−*1

(1)

256, 256, 128 respectively. Each layer has 5 5 kernels and stride of one. Followed by the last convolutional layers

are two fully connected layers of size 328, 192. After that

In the Eq. [1,](#_bookmark1) *l* represents layer, *xl* represents the output of the *l* layer, *Hl* represents a nonlinear transformation. For ResNet, the output of the *l* layer is the output of the *l* 1 layer plus the nonlinear transformation of the output of the *l −* 1 layer.

*−*

# x

|  |  |
| --- | --- |
| weight layer | |
|  | ReLU |

*F(x) x*

weight layer

## identity

*F(x) + x* **+**ReLU

Fig. 2. Unit module block, where x means the input and F(x) means the output of the weight layer, the final output is the sum of F(x) and x.

So we try to apply it to the MNIST dataset for testing.

is a 10 class softmax with cross entropy loss. However, the baseline model has two shortcomings. First, training a powerful CNN model requires a large number of training data. Second, in the pooling layer, CNN loses some of the information, which leads to the ignorance of interrelation- ships between different component[16].

Thus, for small changes in input, the output of CNN will be almost constant, which may result in a higher error rate.

#### Retrofitting

In order to overcome these shortcomings of CNN, we try to introduce CapsNet to optimize the baseline. Fig. [3](#_bookmark4) shows the structure of the CapsNet. CapsNet uses capsules instead of neurons. The input and output of the capsule are high- dimensional vectors, where the module length represents the probability of occurrence of an object, and the direction represents the position, color, size and other information. The output of the low-level capsules is used to generate a prediction through transformation matrices, which are then linearly integrated and passed into high-level capsules according to certain weights. The method of updating the weights is a dynamic routing algorithm, which compares the output of high-level capsules with the prediction of low- level capsules, and increases the input weights of low-level capsules with higher similarity until convergence.

Through the capsule, we retain the information on the

details of the picture. In this way, on the basis of accurately

identifying the image, small changes in the image input will cause small changes in the output. It has a human-like understanding of the three-dimensional space. In addition, with less information loss, it only needs a small amount of data to achieve amazing results compared to CNN.



*ui*

*Wij*

*u*ˆ*j|i*

*u*ˆ1*|*1

*Cij*

*C*11 *C*21

*Sj*

*W*11

*u*ˆ

*S*1

1*|*2

*u*1

*W*12

*u*ˆ

2*|*1

*C*12

*W*13 *W*21

*W*22

*C*22 *S*2

*u*ˆ

2*|*2

*u*2

*u*ˆ3*|*1

*u*ˆ3*|*2 *u*ˆ4*|*1 *u*ˆ4*|*2

*C*13

*C*23

*W*14 *W*23

*S*3

*W*24

*C*14

*C*

*S*4

24

Squashing

Squashing

Squashing

Squashing

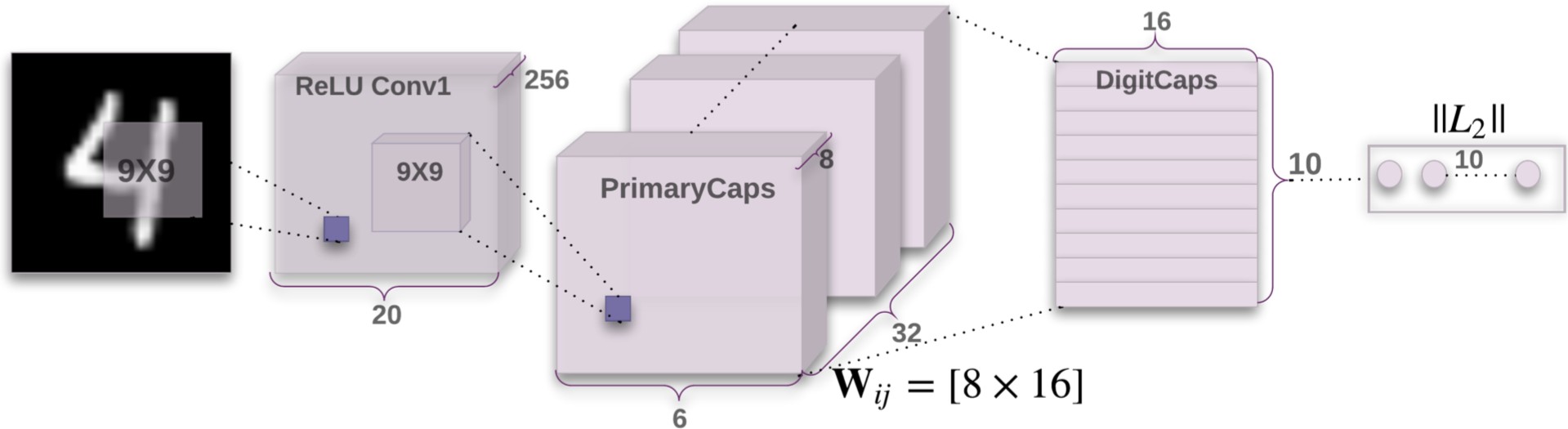


Fig. 4. A simple CapsNet with three layers. The convolutional layer extracts image features, PrimaryCaps integration the features, and Dig- itCaps output the prediction.

to train. DenseNet performs better than CNN on all divided datasets. It also improves the results of ResNet across all datasets but 50% dataset. That is related to DenseNet’s parameter settings. Inspiringly, CapsNet is the best overall model, which outperforms the other models on all tasks and consistently beats the baseline. In addition, we can observe from the Table [1](#_bookmark5) that CapsNet trained with half datasets reach approximately equal accuracy with complete CNN. We attribute this to CapsNet’s ability to generalize in image recognition. This is in line with other research[14][16][17], which suggests that this model is very robust across tasks as well as datasets.

Fig. 3. Structure of CapsNet, where *ui*

weight matrix,

is the input layer, *Wij*

is the

TABLE 1

Results of experiment on divided datasets.

*u*ˆ*j i* is the *ui*’s prediction to *Sj* , *Cij* is the weight and *Sj*

*|*

is the output layer. Squashing is the activate function as shown in Eq. [2.](#_bookmark6)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Accuracy(%) MNIST  Models | 25% | 50% | 75% | 100% |
| CNN | 80.73 | 86.73 | 91.23 | 98.32 |
| ResNet | 79.46 | 90.55 | 93.78 | 99.16 |
| DenseNet | 82.57 | 89.24 | 94.20 | 99.37 |
| CapsNet | 87.68 | 97.12 | 98.79 | 99.75 |

*vj* =

*||sj||*2 *sj*

(2)

1 + *||sj||*2 *||sj||*

#### CapsNet Architecture

The architecture is showed in Fig. [4,](#_bookmark3) it consists of one convolutional layer and two capsule layers[18][19]. The convolutional layer 1 has 256, 9 9 convolution kernels with a stride of 1 and ReLU activation. This layer extracts the basic features of the image, and then uses them as the inputs of the primary capsules layer (PrimaryCaps). The PrimaryCaps contains 32 capsules, which receives the basic features detected by the convolution layer, creating a combination of features. The 32 primary capsules in this layer are essentially similar to the convolutional layer[20]. Each has 8, 9 9 256 convolution kernels with a stride of 2. The output of PrimaryCaps is 6632 eight-dimensional vector. The last layer is digital capsules layer (DigitCaps), it has 10 digital capsules and each of which represents the prediction of number. Every capsule receives input from all capsules in the PrimaryCaps, and finally outputs the result.

*×*

*× ×*

1. **RESULT**

We randomly divided the MNIST dataset into 25%, 50%, 75%, and 100%. Table [1](#_bookmark5) shows the results for the four models across all divided datasets, and we visualize them in Fig. [5.](#_bookmark7) Obviously, CNN continues to be a strong baseline: Though it never provides the best result on a dataset, it gives better results than ResNet on 25% MNIST. Because the ResNet’s network structure requires a larger number of data

100

95

CNN

ResNet DenseNet CapsNet

90

accuracy(%)

85

80

75

25% 50% 75% 100%

MNIST Datasets

Fig. 5. The results for the four models across all divided datasets.

1. **CONCLUSION**

The goal of this paper has been to discover which models perform better across divided MNIST datasets. We com- pared four models on MNIST dataset with different divi- sion, and showed that CapsNet perform best across datasets. Additionally, we also observe surprisingly that CapsNet requires only a small amount of data to achieve excellent performance. Finally, we will apply CapsNet’s ability to generalize in other image recognition field in the future.

In this paper, we have a tendency to pass the parameters of trained multi-task models to single-task models. analysis on MNIST- like datasets show that using multi-task learning will improve image recognition accuracy. The a lot of data we have a tendency to use, the higher results we have a tendency to get. This agrees with applied math learning theory that victimization a lot of information reduces the generalization gap, so rising check set performance, albeit the data comes from a distinct domain.

**REFERENCES**

1. Sabour, Sara, Nicholas Frosst, and Geoffrey E. Hinton. ”Dynamic Routing Between Capsules.” n*eural information processing systems* (2017): 3859-3869.
2. He, Kaiming, et al. ”Deep Residual Learning for Image Recogni- tion.” *computer vision and pattern recognition* (2016): 770-778.
3. Huang, Gao, et al. ”Densely Connected Convolutional Networks.”

*computer vision and pattern recognition* (2017): 2261-2269. [4]Bengio, Yoshua, Patrice Simard, and Paolo Frasconi. ”Learning

long-term dependencies with gradient descent is difficult.” *IEEE transactions on neural networks 5.2* (1994): 157-166.

1. Keysers, Daniel. ”Comparison and Combination of State-of-the-art Techniques for Handwritten Character Recognition: Topping the MNIST Benchmark.” *arXiv: Computer Vision and Pattern Recognition* (2007).
2. Cortes, Corinna, et al. ”AdaNet: Adaptive Structural Learning of Artificial Neural Networks.” *international conference on machine learning* (2017): 874-883.
3. Gidaris, Spyros, and Nikos Komodakis. ”Object Detection via a Multi-region and Semantic Segmentation-Aware CNN Model.”*international conference on computer vision* (2015): 1134-1142.
4. MLA Chatfield, Ken, et al. ”The devil is in the details: an evaluation of recent feature encoding methods.” *british machine vision conference* (2011): 1-12.
5. Ba, Jimmy, Volodymyr Mnih, and Koray Kavukcuoglu. ”Multiple Object Recognition with Visual Attention.” *international conference on learning representations* (2015).
6. Goodfellow, Ian J., et al. ”Multi-digit Number Recognition from Street View Imagery using Deep Convolutional Neural Networks.” *international conference on learning representations* (2014).
7. Hinton, Geoffrey E., Zoubin Ghahramani, and Yee Whye Teh. ”Learning to Parse Images.” *neural information processing systems* (2000): 463-469.
8. Deng, Jia, et al. ”ImageNet: A large-scale hierarchical image database.”*computer vision and pattern recognition* (2009): 248-255.
9. Goodfellow, Ian J., et al. ”Maxout Networks.” *international confer- ence on machine learning* (2013).
10. Abadi, Martin, et al. ”TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems.” *arXiv: Distributed, Parallel, and Cluster Computing* (2015).
11. Ren, Shaoqing, et al. ”Object Detection Networks on Convolu- tional Feature Maps.” *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.7 (2017): 1476-1481.
12. Chang, Jiaren, and Yongsheng Chen. ”Batch-normalized Maxout Network in Network.” *arXiv: Computer Vision and Pattern Recogni- tion* (2015).
13. Glorot, Xavier, Antoine Bordes, and Yoshua Bengio. ”Deep Sparse Rectifier Neural Networks.” *international conference on artificial intel- ligence and statistics* (2011): 315-323.
14. He, Kaiming, and Jian Sun. ”Convolutional neural networks at constrained time cost.” *computer vision and pattern recognition* (2015): 5353-5360.
15. Ren, Shaoqing, et al. ”Faster R-CNN: towards real-time object de- tection with region proposal networks.” *neural information processing systems* (2015): 91-99.
16. Hinton, Geoffrey F.. ”Shape representation in parallel systems.”

*international joint conference on artificial intelligence* (1981): 1088-1096.