**1. Introduction**

The human visual system is one of the wonders of the world. Most of the people can recognize handwriting digits or words easily and effortlessly. Humans have a primary visual cortex, in each hemisphere of our brain, also known as V1, contains 140 million neurons, with tens billions of connections between them. And yet human vision involves not just V1, but an entire series of visual cortices doing progressively more complex image processing. We carry in our heads a supercomputer, tuned by evolution over hundreds of millions of years, and superbly adapted to understand the visual world. Recognizing handwritten digits isn't easy. Humans are good at making sense of what our eyes show us. There is a goof based on the idea that any 3-year-old child can recognize a photo of a bird, but figuring out how to make a computer recognize objects has puzzled the very best computer scientists for over 50 years. Since 1998, human have finally found approaches to object recognition. That sounds like a bunch of made up words from a William Gibson Sci-Fi novel, but the ideas are totally understandable if you break them down one by one.

Neural networks approach the problem in a brand new way. The idea is to take a large number of handwritten digits, known as training examples, and then develop a system which can learn from those training examples. In other words, the neural network uses the examples to automatically infer rules for recognizing handwritten digits. Furthermore, by increasing the number of training examples, the network can learn more about handwriting, and so improve its accuracy. So we could build a better handwriting recognizer by using thousands or even millions or billions of training examples.

In this article we'll introducing two computer program implementing neural network and convolution neural network that learns to recognize handwritten digits. Both two are short program can recognize digits with an accuracy over 98 percent, without human intervention.

We're focusing on handwriting recognition because it's an excellent prototype problem for learning about neural networks in general. As a prototype it hits a sweet spot: it's challenging - it's no small feat to recognize handwritten digits - but it's not so difficult as to require an extremely complicated solution, or tremendous computational power. Furthermore, it's a great way to develop more advanced techniques, such as deep learning. And so throughout the book we'll return repeatedly to the problem of handwriting recognition. Later in the book, we'll discuss how these ideas may be applied to other problems in computer vision, and also in speech, natural language processing, and other domains.

We'll develop many key ideas about neural networks, including two important types of artificial neuron (the perceptron and the sigmoid neuron), and the standard learning algorithm for neural networks, known as stochastic gradient descent. Throughout, we focus on explaining whythings are done the way they are, and on building your neural networks intuition. Also deep learning will be discussed in this article.

**2. Related work**

Some researchers have achieved "near-human performance" on the MNIST database, using a committee of neural networks; in the same paper, the authors achieve performance double that of humans on other recognition tasks. The highest error rate been recorded is the research in 1998, using Pairwise Linear Classifier, has an error rate as 7.6 percent.

In 2004, a best-case error rate of 0.42 percent was achieved on the database by researchers using a new classifier called the LIRA, which is a neural classifier with three neuron layers based on Rosenblatt's perceptron principles.

Some researchers have tested artificial intelligence systems using the database put under random distortions. The systems in these cases are usually neural networks and the distortions used tend to be either [affine distortions](https://en.wikipedia.org/wiki/Affine_transformation) or [elastic distortions](https://en.wikipedia.org/wiki/Elastic_deformation). Sometimes, these systems can be very successful; one such system achieved an error rate on the database of 0.39 percent.

In 2011, an error rate of 0.27 percent, improving on the previous best result, was reported by researchers using a similar system of neural networks. In 2013, an approach based on regularization of neural networks using DropConnect has been claimed to achieve a 0.21 percent error rate. Recently, the single convolutional neural network best performance was 0.31 percent error rate. Currently, the best performance of a single convolutional neural network trained in 74 epochs on the expanded training data is 0.27 percent error rate. Also, the Parallel Computing Center (Khmelnitskiy, Ukraine) obtained an ensemble of only 5 convolutional neural networks which performs on MNIST at 0.21 percent error rate in 2016.

**3. Implementation**

i. data set

ii. Neural Network

iii. Convolution Neural Network

**4. Comparison**

**5. Future Research Directions**

From the approaches we discussed above. We can see that neural break down all the parts into pieces and determine whether this piece satisfied this situation. If yes, then out put the answer. And of-cause, in these two method we only use 2 hidden layers. We can always improve further and further through multiple layers. Ultimately, we'll be working with sub-networks that answer questions so simple they can easily be answered at the level of single pixels. Those questions might, for example, be about the presence or absence of very simple shapes at particular points in the image. Such questions can be answered by single neurons connected to the raw pixels in the image. The end result is a network which breaks down a very complicated question into very simple questions answerable at the level of single pixels. It does this through a series of many layers, with early layers answering very simple and specific questions about the input image, and later layers building up a hierarchy of ever more complex and abstract concepts. To train those neural, we could use learning algorithms so that the network can automatically learn the weights and biases - and thus, the hierarchy of concepts - from training data.

Researchers in the 1980s and 1990s tried using stochastic gradient descent and back propagation to train deep networks. Unfortunately, except for a few special architectures, they didn't have much luck. The networks would learn, but very slowly, and in practice often too slowly to be useful. Since 2006, a set of techniques has been developed that enable learning in deep neural nets. These deep learning techniques are based on stochastic gradient descent and back propagation, but also introduce new ideas. These techniques have enabled much deeper (and larger) networks to be trained - people now routinely train networks with 5 to 10 hidden layers. And, it turns out that these perform far better on many problems than shallow neural networks, i.e., networks with just a single hidden layer. The reason, of course, is the ability of deep nets to build up a complex hierarchy of concepts. It's a bit like the way conventional programming languages use modular design and ideas about abstraction to enable the creation of complex computer programs. Comparing a deep network to a shallow network is a bit like comparing a programming language with the ability to make function calls to a stripped down language with no ability to make such calls.

And furthermore, due to the simplicity of the number output, from 0-9, we can also expand the recognition problem to handwriting words. Like recognizing a-z, A-Z, so that we can recognize the written word. Which means the output will increased.

Besides the increment of output size, we can also improve the input size or minimize the input size to generate more accuracy or reduce the training time. Due to the data characteristic in MNIST database. The input has to be 28\*28=784. But if the size increased. The number of input will be dramatically increased. And we can use some methods like dichotomy to minimize the input data to a considerable size.

**6. Conclusion**

**7. Individual Contribution**

**8. Reference**

LeCun, Yann; Corinna Cortes; Christopher J.C. Burges. ["MNIST handwritten digit database, Yann LeCun, Corinna Cortes and Chris Burges"](http://yann.lecun.com/exdb/mnist/). Retrieved 17 August 2013.

Cires¸an, Dan; Ueli Meier; Jürgen Schmidhuber (2012). ["Multi-column deep neural networks for image classification"](http://repository.supsi.ch/5145/1/IDSIA-04-12.pdf) (PDF). 2012 IEEE Conference on Computer Vision and Pattern Recognition: 3642–3649. [arXiv](https://en.wikipedia.org/wiki/ArXiv):[1202.2745](https://arxiv.org/abs/1202.2745) Freely accessible. [doi](https://en.wikipedia.org/wiki/Digital_object_identifier):[10.1109/CVPR.2012.6248110](https://doi.org/10.1109%2FCVPR.2012.6248110). [ISBN](https://en.wikipedia.org/wiki/International_Standard_Book_Number) [978-1-4673-1228-8](https://en.wikipedia.org/wiki/Special:BookSources/978-1-4673-1228-8).

LeCun, Yann; Léon Bottou; Yoshua Bengio; Patrick Haffner (1998). ["Gradient-Based Learning Applied to Document Recognition"](http://yann.lecun.com/exdb/publis/pdf/lecun-98.pdf) (PDF). Proceedings of the IEEE. 86 (11): 2278–2324. [doi](https://en.wikipedia.org/wiki/Digital_object_identifier):[10.1109/5.726791](https://doi.org/10.1109%2F5.726791). Retrieved 18 August 2013.

Kussul, Ernst; Tatiana Baidyk (2004). ["Improved method of handwritten digit recognition tested on MNIST database"](https://vlabdownload.googlecode.com/files/Image_VisionComputing.pdf) (PDF). Image and Vision Computing. 22 (12): 971–981. [doi](https://en.wikipedia.org/wiki/Digital_object_identifier):[10.1016/j.imavis.2004.03.008](https://doi.org/10.1016%2Fj.imavis.2004.03.008). Retrieved 20 September 2013.

Ranzato, Marc’Aurelio; Christopher Poultney; Sumit Chopra; Yann LeCun (2006). ["Efficient Learning of Sparse Representations with an Energy-Based Model"](http://yann.lecun.com/exdb/publis/pdf/ranzato-06.pdf) (PDF). Advances in Neural Information Processing Systems. 19: 1137–1144. Retrieved 20 September 2013.

Ciresan, Dan Claudiu; Ueli Meier; Luca Maria Gambardella; Jürgen Schmidhuber (2011). ["Convolutional neural network committees for handwritten character classification"](http://www.icdar2011.org/fileup/PDF/4520b135.pdf) (PDF). 2011 International Conference on Document Analysis and Recognition (ICDAR): 1135–1139. [doi](https://en.wikipedia.org/wiki/Digital_object_identifier):[10.1109/ICDAR.2011.229](https://doi.org/10.1109%2FICDAR.2011.229). [ISBN](https://en.wikipedia.org/wiki/International_Standard_Book_Number) [978-1-4577-1350-7](https://en.wikipedia.org/wiki/Special:BookSources/978-1-4577-1350-7). Retrieved 20 September 2013.

Wan, Li; Matthew Zeiler; Sixin Zhang; Yann LeCun; Rob Fergus (2013). Regularization of Neural Network using DropConnect. International Conference on Machine Learning(ICML).

Romanuke, Vadim. ["The single convolutional neural network best performance in 18 epochs on the expanded training data at Parallel Computing Center, Khmelnitskiy, Ukraine"](https://drive.google.com/file/d/0B1WkCFOvGHDdWlZvWUlLd0V3ZFU/view?usp=sharing). Retrieved 16 November 2016.

Romanuke, Vadim. ["Parallel Computing Center (Khmelnitskiy, Ukraine) gives a single convolutional neural network performing on MNIST at 0.27 percent error rate"](https://drive.google.com/file/d/0B1WkCFOvGHDdOC0yR0tfbmpidjg/view?usp=sharing). Retrieved 24 November 2016.

Romanuke, Vadim. ["Parallel Computing Center (Khmelnitskiy, Ukraine) represents an ensemble of 5 convolutional neural networks which performs on MNIST at 0.21 percent error rate"](https://drive.google.com/file/d/0B1WkCFOvGHDddElkdkl6bzRLRE0/view?usp=sharing). Retrieved 24 November 2016.