# CSCI424/CSCI924 - Assignment 3 - Deep Neural Networks

**Christopher Tom Kochovski — Ian Comor**
**Joel Kocherry — Muhammad Asjad**

## Introduction

Deep Neural Networks (DNNs) are architectures of multiple layers of perceptrons, designed to solve complex learning problems. However, DNNs face challenges in terms of training and generalisation. Traditional DNNs with large numbers of interconnections can overfit, and require different training techniques to improve generalization. Pretraining of connection weights and new flavours of neural networks aim to overcome these issues. Convolutional Neural Networks, in particular, are designed to process images by minimizing trainable weights and providing strong generalization abilities. These networks show good promise in complex learning tasks in multiple domains.

## Deep Neural Networks

A Deep Neural Network (DNN) is defined by Hinton *et al.* (2012, p. 84) as "a feed-forward artificial neural network that contains more than one hidden layer of hidden units". DNN's have a number of processing layers that can be used to learn representations of data with multiple tiers of abstraction. The different tiers are obtained by producing non-linear modules that are used to transform the representation at one tier into a representation at a more abstract tier (LeCun *et al.* 2015). By using these tiers, features of the data used for classification is enhanced while irrelevant features are suppressed. Deep convolutional nets and recurrent nets are varieties of DNNs, and algorithms using them have brought about breakthroughs in processing text, images, video, speech and audio (LeCun *et al.* 2015).

### Inital Unpopularity of Deep Neural Network

A major drawback at the inception of DNNs in the 1990s was the limited computational power of computers (Hinton 2012). Furthermore, DNNs require a considerable number of training patterns to arrive at good solutions, which were hampered by the limited size of early labeled data sets (Hinton 2012). These issues led to the initial unpopularity of DNNs.

At this time, there were other competing models such as the Hidden Markov Model and the Support Vector Machine. Even though DNNs achieved similar or slightly better results (depending on the task) than the above models, they were much more difficult to train. Random initialization did not yield good results and one had to be an expert and have sufficient domain knowledge in order to handcraft feature extractors (LeCun *et al.* 2015). Furthermore, they lack the ability to train using unlabeled data, which is problematic as a lot of the real world data available to us is unlabeled.

### Drawbacks of DNN

For learning networks that had more than a few hidden layers, back-propagation alone did not work well in practice due to problems encountered such as the vanishing gradient and curse of dimensionality (Bengio 2009; Glorot and Bengio 2010). The curse of dimensionality is the exponential increase in the number of possible combinations of input values as the number of input neurons increase. Without exponentially larger datasets to keep up, most of the possible combinations are zero, and the network faces training with sparse data (Bengio & Bengio 2000). The learning time does not scale well with an increase in the total number of hidden layers. Other problems using backpropogation include overfitting the test set and the problem of local minima.

#### Overfitting

DNNs with many parameters are very powerful but are prone to overfitting. Overfitting generally occurs when the model is very complex and has too many input parameters compared to the outputs or observations, or a sparse data set (Chapados & Bengio 2001). The model cannot generalise for new patterns because the trainable parameters adapt to the training set. One of the proposed solutions for overfitting is called 'dropout' and was suggested by Srivastava *et al.* (2014). The key idea of this technique is to randomly drop out units along with their connections, which prevents units from becoming excessively dependent.

#### Vanishing Gradient

It was shown by Hochreiter *et al.* (2001) that algorithms relying on the calculation of a complete gradient tend to fade and eventually vanish while performing backpropogation. The gradients are calculated using the derivatives of the error function. From the chain rule

of derivatives, the gradient becomes smaller in the lower layers and eventually becomes too small to make significant weight adjustments. One of the proposed solutions is a novel gradient-based method called the Long Short-Term Memory by Sepp Hochreiter and Jürgen Schmidhuber that is capable of remembering error values for longer durations (Hochreiter *et al.* 2001).

**Local Minima**

When the parameters were randomly initialized for gradient-based optimization methods such as the stochastic gradient descent, the DNN had a high possibility of ending up in a local minima. The network can lead to a poor local minimum if the parameters are not initialized sensibly, and this is a more common as the depth of the network increases (Arel *et al.* 2010). Local minima are suboptimal solutions in the error space. In a deep network with many hidden layers, there are significant numbers of local minima. Unless the weights are already in the local area of the global minimum at the start of training, the result will likely fall into a suboptimal local minima (Arel *et al.* 2010).

## Growing popularity of DNNs

Despite initial unpopularity due to poor performance and small datasets, DNNs eventually came to wider use. There was an exponential growth of unlabeled datasets in the late 2000s. There was a need to use more unsupervised approaches, which was facilitated by Hinton *et al.* (2006) when Deep Belief Networks (DBNs) were created. This was the first major breakthrough for DNNs. The key idea was to use 'pre-training' in order to learn features rather than having to handcraft them. This improvement made training DNNs much easier as features were learnt automatically. The next major event was the advent of fast Graphic Processing Units that made it possible for networks to be trained more than ten times faster (LeCun *et al.* 2015). A lot of the models created using DNNs started outperforming the other existing models. Another breakthrough was the use of convolution networks for the classification of the ImageNet database. With this rise in popularity it lead to DNNs being adopted in various domains such as computer vision and image processing, encouraging more researchers to explore DNNs and deep learning.

# Popular Training Approaches of DNNs

## Deep Belief Networks

Deep Belief Networks (DBNs) is a stack of Restricted Boltzmann Machines (RBMs) and feature detectors (Hinton *et al.* 2006). Training involves following a greedy, layer-wise, unsupervised approach where each RBM receives pattern representations from the level below and learns to encode them in an unsupervised fashion (Hinton *et al.* 2006). Conceptually these layers act as feature detectors and the network learns higher-level features (which represent more 'abstract' concepts found in the input) from the lower layers (which represent 'low-level features') (Arel *et al.* 2010; Bengio *et al.* 2007). High-level abstract concepts are learned by building on simpler concepts that are learned by earlier layers at the beginning of the network (Bengio *et al.* 2007). The training of DBNs is a two step process in which there is a pre-training phase and a fine-tuning phase. In pre-training the network is trained layer-by-layer greedily in an unsupervised fashion. When the network learns single layers, gradient ascent is used to get the optimum weight vector. Weights are updated using contrastive divergence and Gibbs-sampling.

In the RBM layer a vector is inputted to the visible units which are connected to the hidden layer. While going in the opposite direction the original input is reconstructed stochastically by finding the higher-order data correlations observed at the visible units. This process of going back and forth is repeated several times and is known as Gibbs sampling. Using this layer-wise training strategy helps in better initialization of the weights, thus solving the difficult optimization problem of deep networks.

Furthermore following the greedy layer wise approach, can provide good generalization because it initializes upper layers with better representations of relevant high level abstractions. After the pre-training step, the weights between every adjacent layers hold the values that represent the information present in the input data. But in order to get improved discriminative performance, the network is often fine-tuned according to supervised training criterion. This can be done by using gradient descent which uses the weightings learned in the previous pre-training phase. Overall it is observed that the learning algorithm proposed by Hinton *et al.* (2007) has time complexity linear to the size and depth of the network, which allows us to train deep networks at a much faster rate compared to the old approaches.

## Stacked Autoassociators

A stacked autoassociator is a flavour of DNN in which a stack of RBMs are each fed inputs from the previous machine (Bengio 2009). The RBM stack can each be trained separately, and provide the advantage of a white-box approach where the outputs of each layer is visible (Bengio 2009). Autoassociator stacks have been used to pre-train deep feed-forward neural networks in an unsupervised fashion before backpropagation fine-tuning (Bengio *et al.* 2007; Vincent *et al.* 2008). Architecturally, the autoassociator has the same number of nodes in the input layer as the output layer. The autoassociator is trained to reconstruct the given inputs by compression and reconstruction in a compact form (Bengio, 2009). The aim is to minimize the error between the input and the compressed output (Hinton *et al.* 2006). The output from each layer is passed to the next in an unsupervised manner. The final output is the input to a supervised classifier. The supervised criterion is used to fine-tune all parameters of the deep net to achieve optimum performance (Bengio *et al.* 2007).

## Deep Neural Networks and Images

Deep Neural Networks focus on the need to process and classify complex, high-dimensional data, requiring the use of a large number of neurons in numerous layers. In these networks, each layer is fully connected to its adjacent layers (Larochelle *et al.* 2009). For images with even a modest number of pixels, DNNs do not scale well, and the full interconnectedness make DNNs computationally- and memory-intensive (Larochelle *et al.* 2009). This is because of the intractable numbers of connection weights to be trained (Freeman & Skapura 1992). To process images of practical size while being computationally feasible, an alternative technique must be considered. Convolutional Neural Networks (CNNs) are a flavour of DNNs, structured with a reduced number of trainable weights. This results in a reduction in computing time and storage space, and a feasible solution to training with images of practical size (LeCun *et al.* 1998).

# Convolutional Neural Networks

## Definition

A Convolutional Neural Network (CNN) is a trainable deep neural network that is comprised of numerous stages and has a fully-connected neural network at the end for classification; each stage features a number of convolution and pooling/subsampling layers (LeCun, Kavukcuoglu & Farabet 2010). The input and output of each stage is defined as a feature set, comprised of feature maps. Each stage consists of a convolution layer (filter bank layer) and a feature pooling/subsampling layer. Each successive stage progressively learns higher-level features, thereby being able to identify complex scenes or objects in an image by detecting the simpler components of which it is made (Huang & LeCun 2006). CNNs are therefore able to avoid the need for computationally-intensive feature extraction pre-processing, and can itself extract these features implicitly (LeCun *et al.* 1990). Features are extracted using the idea of convolution image kernels (filters) from image processing (Farabet *et al.* 2013). This also reduces the number of connections as the kernel weights are shared by all neurons in a feature. In contrast to any substantial preprocessing, CNNs are able to process the entirety of the procedure, from raw initial image to output classification (Sermanet, Chintala & LeCun 2012).

## CNN Operations and Process Overview

An image is first fed into the CNN. A set of convolution kernels are chosen that can extract a set of particular image features. Each kernel is convolved over the input image to produce a feature map in the first layer (one feature map for each kernel). The set of feature maps are called the feature set. Each feature map is a set of neurons within the hidden layer that share common weights, which are represented as kernels. Using sets of shared (but non-overlapping) weight groups means the stages appear fully-connected, but have fewer trainable weights, thereby reducing computational complexity and storage space (LeCun *et al.* 1990).

Following every convolution layer is a pooling and subsampling layer, where each feature map is pooled to introduce minor translational invariances, and subsampled to reduce the number of connections (LeCun *et al.* 1998). The minor translational invariance is gained because when features are translated within the pooling region, the output of

the pool is often unchanged compared to before the translation. After the stages have been completed the output is fed to a linear classifier to determine exactly what the image/object is (LeCun *et al.* 1998). Supervised training is performed using a gradient descent method to update the kernels accordingly, where the gradients are calculated through backpropagation (LeCun, Kavukcuoglu & Farabet 2010). However, supervised training requires a large number of labelled samples which in certain situations can be difficult to obtain. In contrast, an unsupervised training method only needs a very small labelled set, and can even be improved with pre-training and global refinement. Global refinement is where random parameters are chosen and the network itself is then trained using a gradient descent method to accomplish the same classification and recognition (LeCun, Kavukcuoglu & Farabet 2010; Jarrett et al. 2009).

### Convolution Layer

Broadly, convolution is the dot product operation of a convolution kernel over a part of a dataset. In the context of images, the kernel is a small image that contains a desired visual feature to find. The output of the convolution operation is a feature map, and since the same feature is searched for over the entire image, the feature map is translation invariant (LeCun, Kavukcuoglu & Farabet 2010). The convolution operation is repeated by all kernels in the layer over the image, producing a feature set. The computation of the convolutional layer with $n$ input feature maps and $m$ kernels is:

$$Y = \sum_{i}^{n} \sum_{j}^{m} + B_j + X_i \star K_j \tag{1}$$

where $Y$ is the output feature set, $X_i$ is the $i^{th}$ input feature map, $B_j$ is the trainable bias parameter for the $j^{th}$ kernel, $K_j$ is the $j^{th}$ kernel, and $\star$ is the convolution operation (LeCun, Kavukcuoglu & Farabet 2010). While LeCun, Kavukcuoglu and Farabet (2010) consider the non-linearity operation as a different layer to convolution, it has at other times been included in the convolution layer (LeCun *et al.* 1998; Huang & LeCun 2006). LeCun, Kavukcuoglu and Farabet (2010) dictate any nonlinear function may be used and most commonly the tanh function or sigmoid function is used. However more recently Krizhevsky, Sutskever and Hinton (2012) discuss Rectified Linear Units (ReLU) which have been identified as a significantly more efficient use of a nonlinear function, $f(x) = \max(0, x)$ in regard to training time and accuracy with gradient descent. ReLU reached 25% training

error rate in 86% less epochs because, when used, the function does not plateau like the sigmoid or tanh functions. Both these functions contain plateus where the gradient is near-zero and descent does not occur. An optional operation may also be applied known as normalization (LeCun, Kavukcuoglu & Farabet 2010) which provides competition between relatively close features in a feature map, and features that are the same spatially.

### Pooling/Subsampling Layer

Each convolution layer is followed by a pooling/subsampling layer. The layer takes each feature map from the convolution layer and subsamples them to smaller sizes by pooling pixel neighbourhoods (LeCun, Kavukcuoglu & Farabet 2010). Pooling techniques may include Max Pooling where the strongest feature from the feature map in a given window is chosen (LeCun, Kavukcuoglu & Farabet 2010), Average Pooling where the average features from the feature map in a given window are chosen (LeCun, Kavukcuoglu & Farabet 2010; Jarrett et al. 2009), or L2 Pooling where the strongest features get given the heaviest weights and the weakest features the softest weights (Sermanet, Chintala & LeCun 2012). The result is a set of feature maps of reduced size, which leads to reduced computational complexity and increased tolerance to minor translational invariances (Tompson *et al.* 2015). All pooling windows in a feature map are non-overlapping, and their union is the entire feature map. The output is a subsampled feature map that becomes the input to the next convolution layer, or the vector input to the linear classifier if it is the final stage (LeCun *et al.* 1998). The most popular technique is max pooling (Jarrett *et al.* 2009).

### Convolutional Neural Network Advantages

DNNs generally do not scale well, and the larger numbers of connections make DNNs computationally intensive (Larochelle *et al.* 2009). CNNs, due to their architecture of sharing weights, scale far better and are therefore less computationally intensive (LeCun *et al.* 1990). More weights do not generally lead to better results, and the more abundant weights in a DNN leads to overfitting the training set (Larochelle *et al.* 2009). This means the DNN cannot generalise for spatial translations or rotations, and require deliberate transformations of input images to assist in generalisation (Larochelle *et al.* 2009). The ability to generalise while demanding minimal computational cost makes CNNs a powerful tool for highly accurate object recognition and classification in the real world.

### Real World Applications of CNNs

The work of Sermanet, Chintala & LeCun (2012) showed specifically that CNNs can classify house numbers from photographs. Another application is detecting and understanding traffic signs, which has applications in autonomous vehicles. By using a modified CNNs Sermanet & LeCun (2011) presented a method of traffic sign recognition which lead to a higher accuracy percentage for both RGB and Grayscale images. By utilizing a sophisticated nonlinear function and changing where the output is fed from the each stage (instead of feeding the feature maps between each layer in a stage, the feature maps are also sent to the linear classifier) Sermanet & LeCun (2011) managed to obtain this higher accuracy. In 2014 the team GoogleLeNet placed first in both classification and detection in the ImageNet large-scale visual recognition challenge (ILSVRC) (Szegedy 2014), using a modified CNN. The results of the CNN bettered the previous year's best result. Applications of GoogleLeNet has already started: Google has modified the neural network and revamped the way that they determine what a suitable thumbnail for YouTube videos are based on the uploaded video's frame (Yang & Tsai 2015). By using this new method, surveyed users had a 65% preference in these new thumbnails over the older thumbnails which were chosen via the old method (Yang & Tsai 2015).

## Conclusion

Originally, DNNs suffered from the limitations because of the traditional backpropogation method. Better learning alternatives including pretraining allow for better generalization of the data. Deep Belief Networks provided the breakthrough in improved training of DNNs. Deep Belief Networks use stacks of feature detectors to learn the complex relationships from detecting abstract components, and are an improvement to the DNN. CNNs can process images of practical size that cannot be done feasibly with a traditional DNN. CNNs rely on prior knowledge of the simple parts that make up objects or scenes, and are designed with this knowledge to find the salient features in the image. CNNs have a less-connected network than DNNs, allowing them to better generalise features and remain tolerant to spatial transformations. This makes them a better alternative to two-dimensional data such as images. As such, CNNs produce state-of-the-art results in object and scene recognition in international competitions and benchmarks, and therefore have a place for real world applications.

## References

Arel, I, Rose, D & Karnowski, T 2010, 'Deep Machine Learning? A New Frontier in Artificial Intelligence Research [Research Frontier]', *IEEE Computational Intelligence Magazine*, vol. 5, no. 4, p. 13, viewed 14 October 2015, Publisher Provided Full Text Searching File, EBSCO*host*.

Bengio, Y 2009, 'Learning Deep Architectures for AI', *Foundations and Trends®in Machine Learning*, vol. 2, no. 1, pp. 1-127, viewed 3 October 2015, <http://www.iro.umontreal.ca/~bengioy/papers/ftml.pdf>.

Bengio, S & Bengio, Y 2000, 'Taking on the Curse of Dimensionality in Joint Distributions Using Neural Networks', *IEEE Transactions on Neural Networks*, vol. 11, no. 3, pp. 550-557, viewed 3 October 2015, Business Source Complete, EBSCO*host*.

Bengio, Y, Lamblin, P, Popovici, D & Larochelle, H 2007, 'Greedy Layer-Wise Training of Deep Networks', *Advances In Neural Information Processing Systems 19*, p. 153-158, viewed 4 October 2015, Publisher Provided Full Text Searching File, EBSCO*host*.

Chapados, N & Bengio, Y 2001, 'Input decay: Simple and Effective Soft Variable Selection', *Proceedings Of The International Joint Conference On Neural Networks*, vol. 2, pp. 1233-1237, Washington DC, USA, viewed 16 October 2015, Scopus®, EBSCO*host*.

Farabet, C, Couprie, C, Najman, L & LeCun, Y 2013, 'Learning Hierarchical Features for Scene Labeling', *IEEE Transactions On Pattern Analysis & Machine Intelligence*, vol. 35, no. 8, pp. 1915-1929, viewed 6 October 2015, Computers & Applied Sciences Complete, EBSCO*host*.

Freeman, JA & Skapura, DM 1992, *Neural Networks Algorithms, Applications, and Programming Techniques*, Addison-Wesley Publishing Company, United States of America.

Glorot, X & Bengio, Y 2010, 'Understanding the Difficulty of Training Deep Feedforward Neural Networks', *Proceedings of the 13$^{th}$ International Conference on Artificial Intelligence and Statistics (AISTATS)*, Sardinia, Italy, viewed 10 October 2015, <http://jmlr.org/proceedings/papers/v9/glorot10a/glorot10a.pdf>.

Hinton, G, Srivastava, N, Swersky, K, Tieleman, T & Mohamed, A n.d., 'The ups and downs of backpropogation', lecture notes, viewed 1 October 2015, <http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec13.pdf>.

Hinton, GE 2006, 'A Fast Learning Algorithm for Deep Belief Nets', *Neural Computation*, vol. 18, no. 7, pp. 1527-1554, viewed 30 September 2015, Psychology and Behavioural Sciences Collection, EBSCOh*ost*.

Hinton, G, Deng, L, Yu, D, Dahl, G, Mohamed, A, Jaitly, N, Senior, A, Vanhoucke, V, Nguyen, P, Sainath, T & Kingsbury, B 2012, 'Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups', *IEEE Signal Processing Magazine*, vol. 29, no. 6, p. 82-97, viewed 6 October 2015, Publisher Provided Full Text Searching File, EBSCO*host*.

Hochreiter, S, Bengio, Y, Frasconi, P & Schmidhuber, J 2001, 'Gradient Flow in Recurrent Nets: The Difficulty of Learning Long-Term Dependecies', in Kolen, JF & Kremer, SC (eds), 'A Field Guide to Dynamical Recurrent Networks', Wiley-IEEE Press, viewed 6 October 2015, <ftp://ftp.idsia.ch/pub/juergen/gradientflow.pdf>.

Huang, FJ & LeCun, Y 2006, 'Large-scale Learning with SVM and Convolution Nets for Generic Object Categorization', viewed 2 October 2015, <yann.lecun.com/exdb/publis/pdf/huang-lecun-06.pdf>.

Jarrett, K, Kavukcuoglu, K, Ranzato, M & LeCun, Y 2009, 'What is the Best Multi-stage Architecture for Object Recognition?', *2009 IEEE 12$^{th}$ International Conference On Computer Vision*, pp. 2146-2153, Kyoto, Japan, viewed 25 September 2015, Publisher Provided Full Text Searching File, EBSCO*host*.

Krizhevsky, A, Sutskever, I & Hinton, GE 2012 'ImageNet Classification with Deep Convolutional Neural Networks', in Pereira F, Burges, CJC, Bottou, L & Weinberger, KQ (eds), *Advances in Neural Information Processing Systems 25*, Curran Associates, Inc., pp. 1097-1105, viewed 25 September 2015, <http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf>.

Larochelle, H, Bengio, Y, Louradour, J & Lamblin, P 2009, 'Exploring Strategies for Training Deep Neural Networks', *Journal Of Machine Learning Research*, vol. 10, no. 1, pp. 1-40, viewed 2 October 2015, Business Source Complete, EBSCO*host*.

LeCun, Y & Bengio, Y 1995, 'Convolutional Networks for Image, Speech, and Time-Series', viewed 3 October 2015, <yann.lecun.com/exdb/publis/pdf/lecun-bengio-95a.pdf>.

LeCun, Y, Bengio, Y & Hinton, G 2015, 'Deep learning', Nature, vol. 521, no. 7553, pp. 436-444, viewed 14 October 2015, MasterFILE Complete, EBSCO*host*.

LeCun, Y, Boser, B, Denker, JS, Henderson, D, Howard, RE, Hubbard, W & Jackel, LD 1990, 'Handwritten Digit Recognition with a Back-Propogation Network', viewed 1 October 2015, <yann.lecun.com/exdb/publis/pdf/lecun-90c.pdf>.

LeCun, Y, Bottou, L, Bengio, Y & Haffner, P 1998, 'Gradient-based Learning Applied to Document Recognition', *Proceedings Of The IEEE*, vol. 86, no. 11, pp. 2278-2324, viewed 20 September 2015, Publisher Provided Full Text Searching File, EBSCO*host*.

LeCun, Y, Kavukcuoglu, K & Farabet, C 2010, 'Convolutional Networks and Applications in Vision', *Proceedings Of 2010 IEEE International Symposium On Circuits & Systems (ISCAS)*, pp. 253-256, Paris, viewed 2 October 2015, Publisher Provided Full Text Searching File, EBSCO*host*.

Sermanet, P, Chintala, S & LeCun, Y 2012, 'Convolutional Neural Networks Applied to House Numbers Digit Classification', *Proceedings Of The 21$^{st}$ International Conference On Pattern Recognition (ICPR2012)*, pp. 3288-3291, Tsukuba, Japan, viewed 1 October 2015, Publisher Provided Full Text Searching File, EBSCO*host*.

Sermanet, P & LeCun, Y 2011, 'Traffic Sign Recognition with Multi-scale Convolutional Networks', *Proceedings of International Joint Conference on Neural Networks (IJCNN'11)*, pp. 2809-2813, July, San Jose, California, USA, viewed 2 October 2015, <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6033589&isnumber=6033131>.

Szegedy, C <https://plus.google.com/113064407842505371107>, 5 September 2014, 'Building a Deeper Understanding of Images', Google Research Blog, Google, viewed 2 October 2015, <http://googleresearch.blogspot.com.au/2014/09/building-deeper-understanding-of-images.html>.

Tompson, J, Goroshin, R, Jain, A, LeCun, Y & Bregler, C 2014, 'Efficient Object Localization Using Convolutional Networks', viewed 15 October 2015, arXiv, EBSCOhost, <http://arxiv.org/pdf/1411.4280v3.pdf>.

Yang, W & Tsai, MH 2015, 'Improving YouTube Video Thumbnails with Deep Neural Nets', viewed 13 October 2015, <googleresearch.blogspot.com.au/2015/10/improving-youtube-video-thumbnails-with.html>.