How does depth impact the performance and training of Neural Networks?

Neural networks play a crucial role within machine learning allowing tasks that were previously far more complicated to be completed. This tutorial examines how a key design choice of depth affects a neural network learning process.

A neural network is a type of machine learning model inspired by the structure and function of neurons in the human brain. It learns from earlier data to identify complex patterns within a dataset.

When are they used?

- Image recognition and computer vision
- Natural language processing
- Disease prediction
- Fraud analysis
- And many more applications!

How do they work?

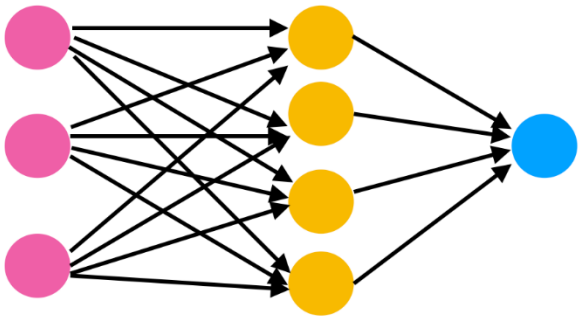

Figure 1 – Neural network architecture (Logan, 2017)

Above is a simple architecture of a feedforward neural network. We can see that the format consists of three distinct types of layers. We have an input layer where the initial data is received, and each input value is multiplied by a corresponding weight. These weighted inputs are then passed to the hidden layer, which there can be a number of. Finally, the output layer produces the final prediction or classification of the network. Therefore, at each neuron the updated weight is given by:

Where is the weight, is the input from the previous step and is some bias term. This is then passed into an activation function before the next layer.

A crucial component of neural networks is the activation function used. These help to create nonlinearity within a network allowing it to model complex relationships with the data and are applied at each neuron. Without them, we would just create a linear model, limiting abilities to solve more complex problems. Common ones include sigmoid, tanh, RELU, Leaky RELU and SoftMax – SoftMax only used in the output layer for multi class classification problems.

For a neural network to be effective it must be able to learn based on prior knowledge to be able to build strong models. One process for this is backpropagation which is used to adjust weights to minimise the error between the actual target and predicted output. It involves computing the gradient of the loss function with respect to each weight using the chain rule and then updating the weights through an optimisation algorithm, typically gradient descent. The chain rule allows the network to compute how a minor change in weight affects the overall loss, even though many layers — this is the essence of backpropagation.

The above equation shows how weights are updated at each point. represents the learning rate which is used to scale the update based on whether the prediction was correct for each neuron. This can be used to increase or decrease the weights to minimise the error.

As mentioned, the way we measure a model's success rate is the loss function. This is used to measure how far we are from predicting the perfect output and common ones are Mean squared error (MSE) for regression and cross entropy for classification.

Depth of Neural Networks

Depth within a neural network refers to how many hidden layers are in the network. By using more hidden layers, we may be able to visualise more complex patterns within a dataset. An example of this could be in images, the first layer recognises edges, the second layer can turn these edges into shapes and further layers may be able to recognise objects. This is important when it comes to tasks like classification of images so we can get more intricate predictions with fewer computations making it crucial in deep learning.

It has been shown that a 3-layer neural network can have a more constant accuracy than a 2-layer neural network unless the 2-layer network has complex and numerous neurons and weights. This means that the size of the network needed is drastically smaller but exponentially more valuable with increased depth which is key within deep learning (Eldan et al, 2016).

The more depth used can have negative connotations on the performance of a model in a number of ways. Firstly, the deeper we make neural networks leads to a greater computational cost due to longer training times due to more operations and greater memory usage. Greater technology has slightly diminished this issue but it is important to consider before creating a model, so you do not unnecessarily increase this.

One potential issue with increased network depth is overfitting. This happens when a network becomes overly complicated and begins to recognise noise and irrelevant patterns in the training data, resulting in a model that doesn't generalise well for unseen data. As depth increases, the network will be more susceptible to overfitting due to a larger number of parameters, which are highly flexible and more prone to memorising data instead of learning meaningful patterns. Many techniques have been used to aim to combat this including dropout, regulation and early stopping.

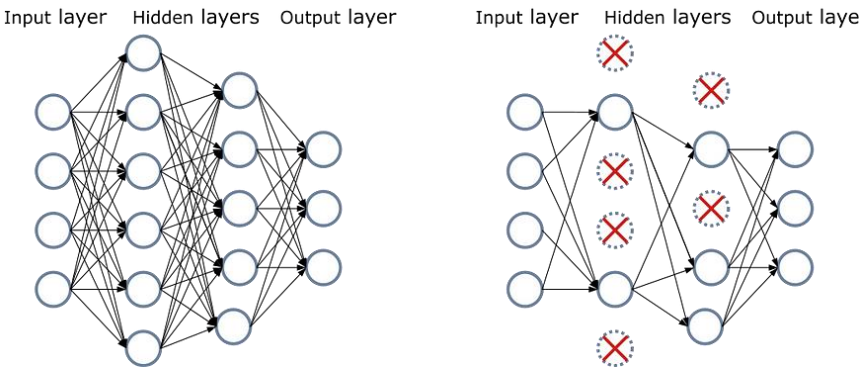

Figure 2: Dropout example for neural networks (Understanding Dropout: A Key to Preventing Overfitting in Neural Networks, 2024)

Dropout is a regularisation technique where certain neurons are randomly switched off during the training phase based on a probability as shown in the figure above. This forces the network to learn multiple redundant representations, improving the model's generalisation ability Dropout prevents

reliance on specific neurons, promoting robustness by training an ensemble of smaller networks with different active subsets, which improves generalisation by lowering the overall test error. However, dropout works worse with small datasets as it decreases the learning capacity meaning it will often oversimplify the model.

Early stopping is a technique that halts training when the validation loss begins to increase, signalling that the model is starting to overfit the training data. By stopping at this point, early stopping prevents the model from learning noise or irrelevant patterns, improving its generalisation to unseen data. However, by stopping too early the model may stop before reaching a global minimum, thus not achieving maximum performance. Patience parameters can be used to give extra epochs to see if the loss is only temporary before improving again.

Another challenge with increasing depth can be vanishing or exploding gradients. This issue occurs during the backpropagation phase when the gradients used to update weights become very small (vanishing) or extremely large(exploding), limiting the network's ability to learn effectively.

Vanishing gradients make it difficult for the earlier layers to receive meaningful updates, slowing down or even halting learning. Exploding gradients, on the other hand, can cause unstable training, leading to large weight updates and divergence in the model.

One way which has been shown to limit this is via residual connections, as introduced in ResNet. ResNet introduces skip connections where the input layer is directly added to the output of a deeper layer connections This helps to preserve the gradient flow and enables better training of deep networks without sacrificing learning performance (He et al, 2016)

The aim of the tutorial is to examine how the performance of a multi-layer perceptron changes as the depth increases. A number of factors may affect this such as amount of training data or task complexity but the focus of this is on the CIFAR 10 dataset within the keras package. Therefore, as we go through the process we will examine:

- Training models with different amounts of layers
- Compare their accuracy, training behaviour and overfitting tendencies
- Does dropout change these views?
- Insights into how depth affects performance and visualise benefits and drawbacks

The CIFAR-10 dataset consists of 60,000 colour images of 10 different classes. The goal of this experiment is to explore how increasing the depth of the neural network helps capture more complex patterns, potentially improving accuracy.

Experiment setup

Network architecture

Input layer – 3072 neurons (32x32x3 flattened images)

Hidden layer – varying layers of depth, 128 neurons, RELU activation function

Output layer – 10 neurons (for each class), SoftMax activation function.

Hyperparameter

Optimiser – Adam – effective learning rate for stable convergence

Loss – Categorical Cross entropy – for classification of each item

Batch size – 64 – balance computational efficiency and generalisation.

Epochs – 20 – sufficient amount to build a strong model whilst avoiding overfitting.

Validation data – 20% - allows us to test the model's performance on unseen data to see how well the model represents the population whilst training.

Based upon this, we would expect to see a lower training loss as the complexity increases due to the ability to exploit these. We use keras to help us create our model and therefore are bale to evaluate a range of metrics.

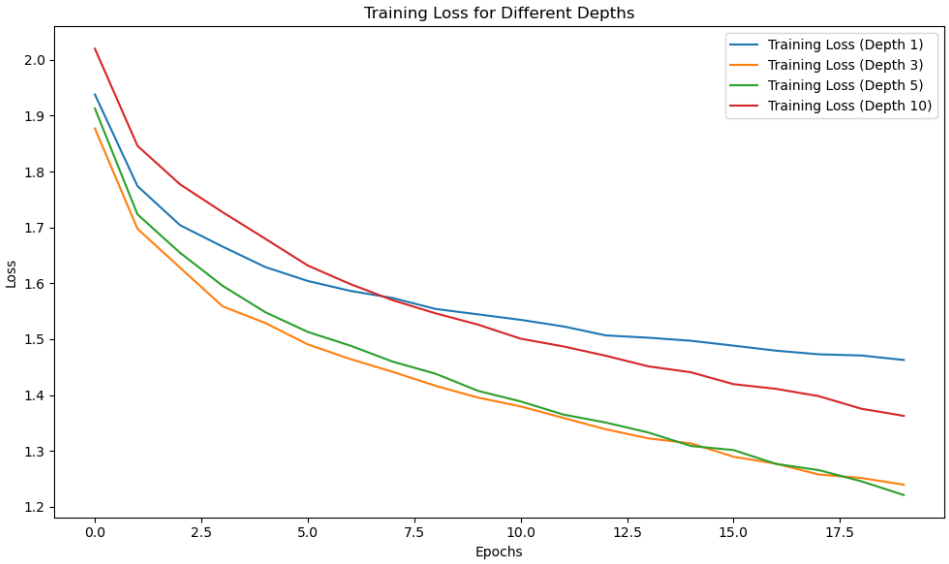

Figure 3: Training loss of different depths

Firstly, we can examine the performance during the training phase of the test. As we run through more epochs, the model can learn far better thus reducing the training loss as the model is able to recognise certain intricacies. In this case, the model with 3 and 5 layers performs better than the larger and smaller layers, suggesting this would be the optimal range.

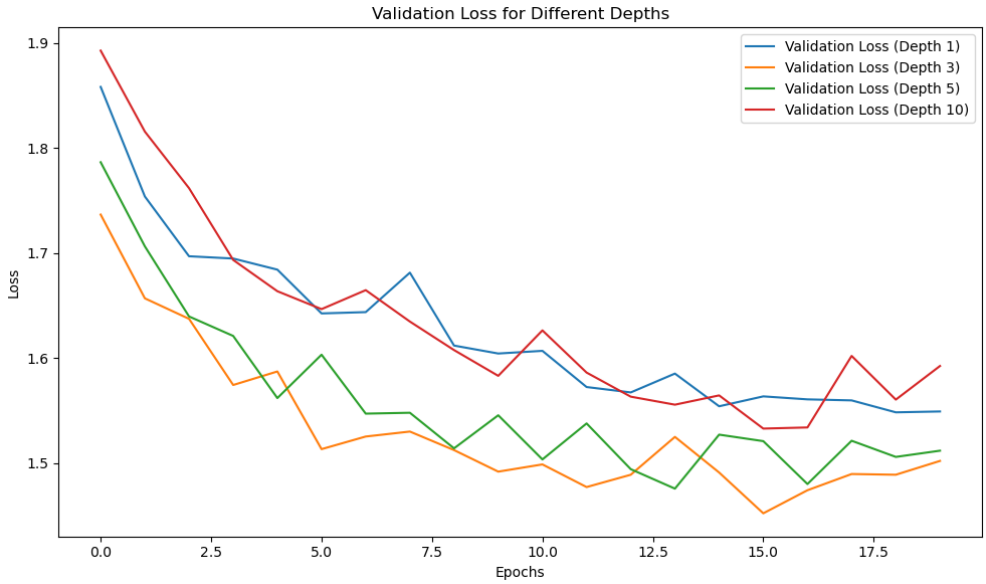

Figure 4: Validation loss over epochs

However, more importantly to see if our model generalised well, we need to visualise how it performs on data it has not seen. We do this by using our validation data and examine the loss at each epoch. Yet again, the model with 3 layers appears to perform best. The performance over the single layer shows that the increase in depth has allowed us to increase the model's ability to classify more images correctly. However, as we start to lose this effect with the more layers, we add we begin to see the negativity of depth as the model will begin too overfit. We would expect to see these exaggerated as validation loss will begin to increase the more epochs that are used.

We can also visualise the test loss of the final model in the table below which again reiterates that the model is performing best with 3 layers with a loss of 1.4685 which is better than the 1 and 10 layer networks.

Therefore, it is clear that the model with three layers offers the best balance between learning capacity and generalisation. The 1-layer model lacked complexity whereas above 5 layers the model began to overfit as shown by rising validation loss.

As previously found, a dropout layer can allow more depth to be used as neurons are turned off during the learning process. We can apply this to our dataset and by choosing to focus on the five hidden layers as it will help us see if the dropout layer helps the increased depth. However, after running this for our code, we don't see any improvement therefore showing we may be best off using another technique or limiting the depth to have a better generalisation of our model.

Overall, deep neural networks play a crucial role in machine learning into understanding complex relationships and help build more solidified models. However, it is important to evaluate when depth can become detrimental and begin to not build a model that generalises the whole dataset. It is also crucial to consider whether a simple neural network is enough in comparison to other machine learning techniques.

GitHub repository

https://github.com/edeery3/Tutorial

References

Eldan, R. and Shamir, O., 2016, June. The power of depth for feedforward neural networks. In Conference on learning theory (pp. 907-940). PMLR.

Understanding Dropout: A Key to Preventing Overfitting in Neural Networks, https://vitalitylearning.medium.com/understanding-dropout-a-key-to-preventing-overfitting-in-neural-networks-21b28dd7c9b1, Medium, 2024

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R., 2014. Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research, 15(1), pp.1929-1958.

He, K., Zhang, X., Ren, S. and Sun, J., 2016. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778). recognition (pp. 770-778).