# Individual Assignment

## Network Analysis

## Hans Alberto Franke

## Feb/21

**Individual Assignment questions**

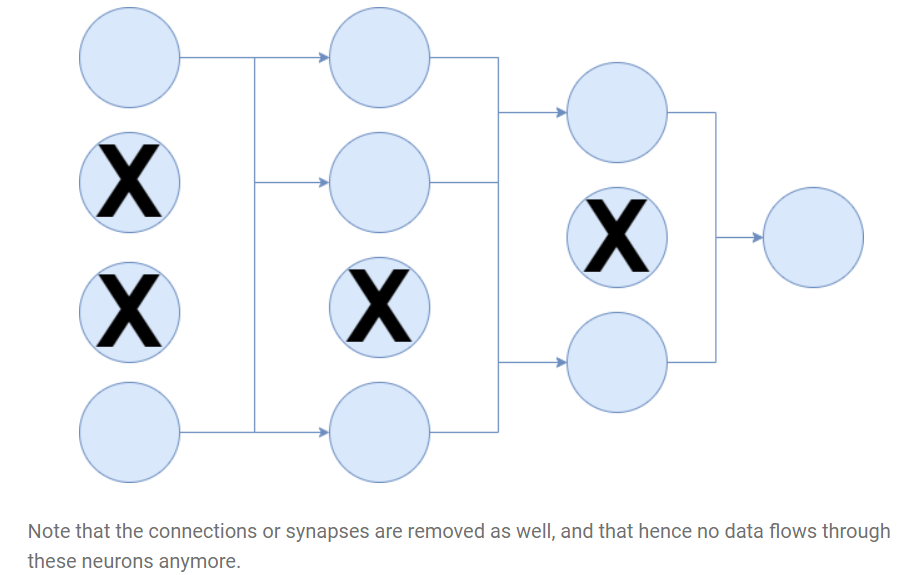
The questions below are related to the group assignment and lectures but are not suitable for group work. Please answer these questions individually in writing, and submit the result on Blackboard by the end of March 9th.

## Describe the principles of overfitting and how dropout can reduce this (Question 1, 5 points)

### **Answer**:

**Overfitting,** being too sensitive to your training data. In other words, your model predict “too good” in training set, but not so well in real data or even in test set. Both underfitting and overfitting are to be avoided, as your model will perform worse than it could perform theoretically. Fortunately, certain techniques – called regularizers – can be used to reduce the impact of overfitting. **Dropout** is one of them

With Dropout, the training process essentially drops out neurons in a neural network. They are temporarily removed from the network, which can be visualized as follows:



Source: <https://www.machinecurve.com/index.php/2019/12/16/what-is-dropout-reduce-overfitting-in-your-neural-networks/>

This removal of neurons and synapses during training is performed at random, with a parameter p that is tunable (or, given empirical tests, best set to 0.5 for hidden layers and close to 1.0 for the input layer). This effectively means that, according to the authors, the “thinned” network is sampled from the global architecture, and used for training.

At test time, “it is not feasible to explicitly average the predictions from exponentially many thinned models” (Srivastava et al., 2014). That’s true: it would become a computational burden when hundreds of thousands of epochs/minibatches have to be averaged, especially when networks become really large.

Fortunately, there is a solution – which is simple, but produces the same result. By using one neural network, where the weight outputs are scaled down according to the p with which a unit was retained during training. This means that the expected output at training time is the same as the true output at test time, resolving the computational issue and making Dropout usable in practice.

Computing the gradient is done with respect to the error, but also with respect to what all other units are doing (Srivastava et al., 2014). This means that certain neurons, through changes in their weights, may fix the mistakes of other neurons. These, Srivastava et al. (2014) argue, lead to complex co-adaptations that may not generalize to unseen data, resulting in overfitting.

Dropout, then, prevents these co-adaptations by – as we wrote before – making the presence of other hidden [neurons] unreliable. Neurons simply cannot rely on other units to correct their mistakes, which reduces the number of co-adaptations that do not generalize to unseen data, and thus presumably reduces overfitting as well.

### **Dropout** prevents **overfitting** due to a layer's "over-reliance" on a few of its inputs. Because these inputs aren't always present during training (i.e. they are dropped at random), the layer learns to use all of its inputs, improving generalization.

--------------------------------------------------------------------------------------------------------------------------------------

Read the research paper “Performance-optimized hierarchical models predict neural responses in higher visual cortex”, available from:

<http://www.pnas.org/content/pnas/111/23/8619.full.pdf>

## Write a short (~500 word) summary of the experimental approach and results. (Question 2, 10 points)

---------------------------------------------------------------------------------------------------------------------------------------

It should be clear from the Group Assignment that even a relatively simple deep convolutional learning network is quite computationally intensive to run on a personal computer. Here we will move to a web-based interface for deep learning, at: http://playground.tensorflow.org/

Here, you can classify the object positions in different data sets (left panel) using deep convolutional network of differing complexity, different numbers of feature maps (number of ‘neurons’ in each hidden layer), different numbers of layers and different inputs (‘features’ column). At the top, you can also change the activation function and add normalisation (regularization). In the left column, you can change the ratio of training and test data, and add noise to the network to improve generalization to simulate imperfect inputs.

## Play around with these settings and see how they affect your ability to learn classification of different data sets. Write down what you found and how you interpret the effects of these settings. This question is intentionally open to allow you to explore the process. (Question 3, 8 points)

## What is the minimum you need in the network to classify the spiral shape with a test set loss of below 0.1? (Question 4, 7 points)

**Backpropagation of error:**

So far, we have avoided explaining backpropagation in detail. We have discussed what backpropagation does, and used Keras’s implementation to train our networks. But we have not looked at how it works because the mathematics are complex and do not fit well with the goals of this course. However, this is a major principle in machine learning with neural networks, so the final part of the assignment will attempt to explain it.

To start this part of the assignment, we suggest watching videos 3 and 4 from the playlist at the following link. You may like to start by watching videos 1 and 2, as these set things up for videos 3 and 4:

https://www.youtube.com/playlist?list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU

Please to answer the following question after viewing the content above:

## Explain the principle of backpropagation of error in plain English in about 500 words. This can be answered with minimal mathematical content and should be IN YOUR OWN WORDS. What is backpropagation trying to achieve, and how does it do so? (Question 5, 8 points)

**BONUS QUESTION:**

If you are also attempting to answer question 6, you should also look at the page here:

https://github.com/stephencwelch/Neural-NetworksDemystified/blob/master/Part%204%20Backpropagation.ipynb

## Describe the process of backpropagation in mathematical terms. Here, explain (in English, in about 500 words) what each equation you give does, and relate this to the answers given in Question 5. You are welcome to express equations in your own R or python code rather than using equation layout, but you need to make clear you understand what each line is doing. (Question 6, 5 points).