Neural Network Application on the Minimum Hybridization Problem

Background – Minimum Hybridization

Minimum Hybridization is the problem of combining a set of evolutionary trees and combining them in the simplest way possible.

We use the following definitions:

- **Phylogenetic Network**
- **Phylogenetic Tree**
- **Reticulation**
- **Cherry**

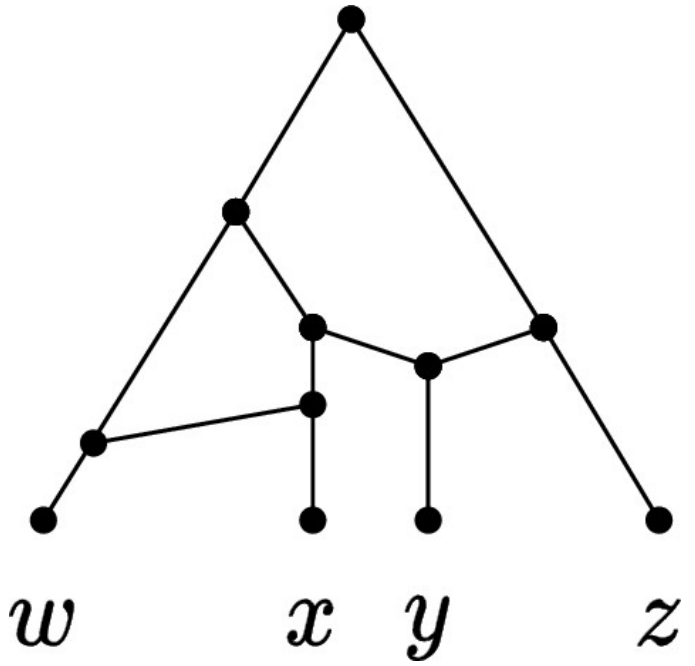
Definition of a Phylogenetic Network

A **Phylogenetic Network**, **N**, is defined as a directed acyclic graph on a set of **taxa**, **X**, which defines the leaf set.

In this case, the taxa would be:
X={w, x, y, z}

Each network has a **reticulation number**, defined as **r(N)**.

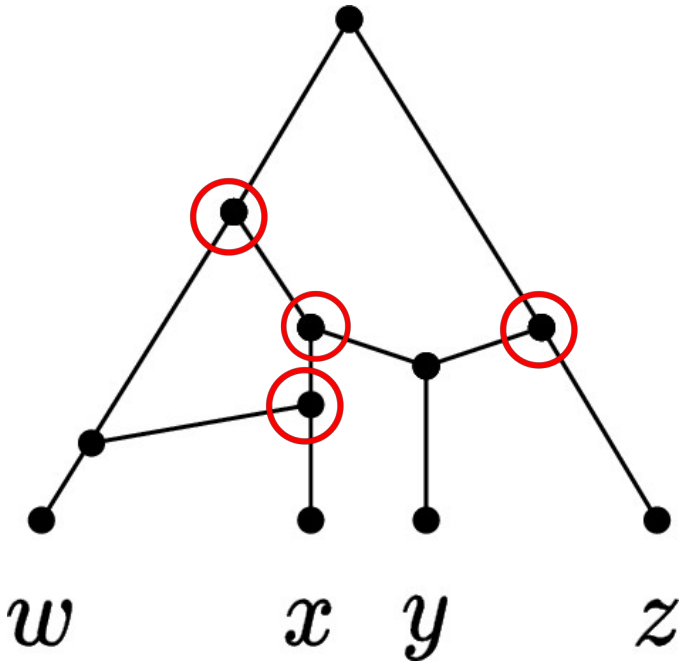
We only considered **binary phylogenetic networks** for this research project.



Definition of a Reticulation

There are three kinds of nodes in a phylogenetic network: **tree nodes, leaf nodes** and **reticulations**.

Tree nodes have in-degree 1 and out-degree 2.

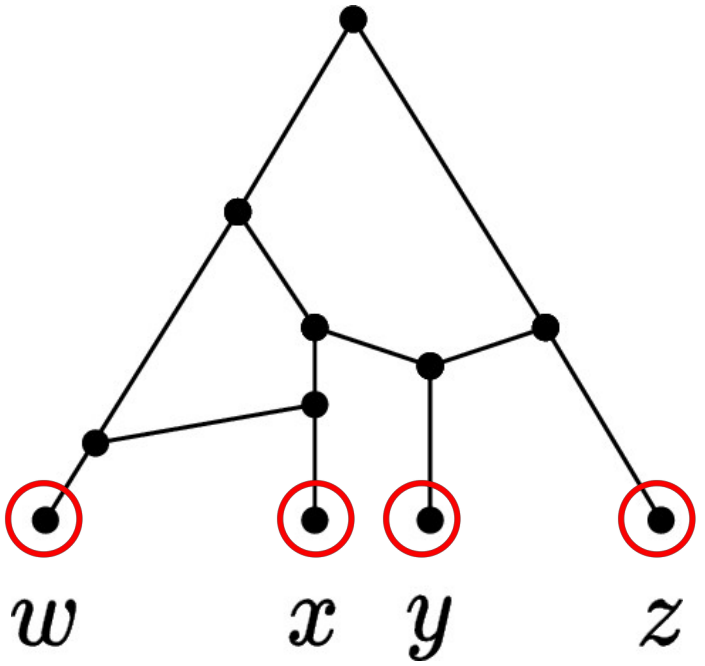


Definition of a Reticulation

There are three kinds of nodes in a phylogenetic network: **tree nodes, leaf nodes** and **reticulations**.

Tree nodes have in-degree 1 and out-degree 2.

Leaf nodes have in-degree 1 and out-degree 0.



Definition of a Reticulation

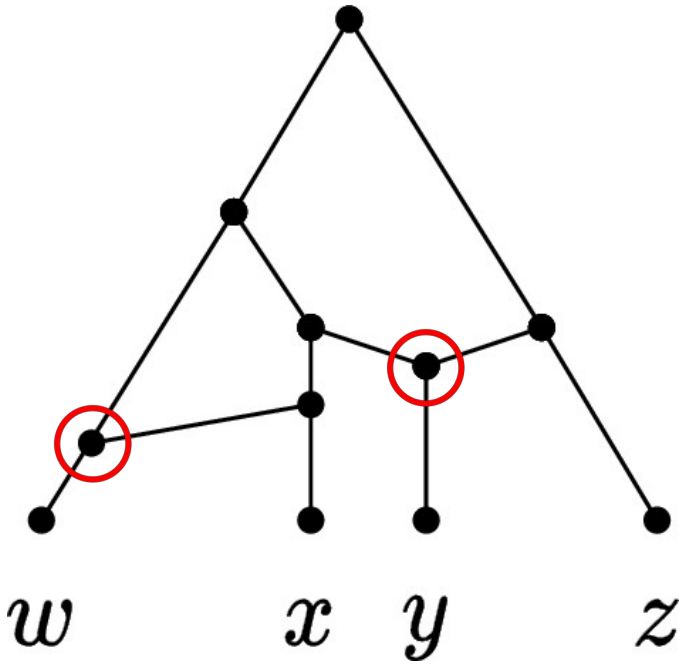
There are three kinds of nodes in a phylogenetic network: **tree nodes, leaf nodes** and **reticulations**.

Tree nodes have in-degree 1 and out-degree 2.

Leaf nodes have in-degree 1 and out-degree 0.

Reticulations have in-degree 2 and out-degree 1.

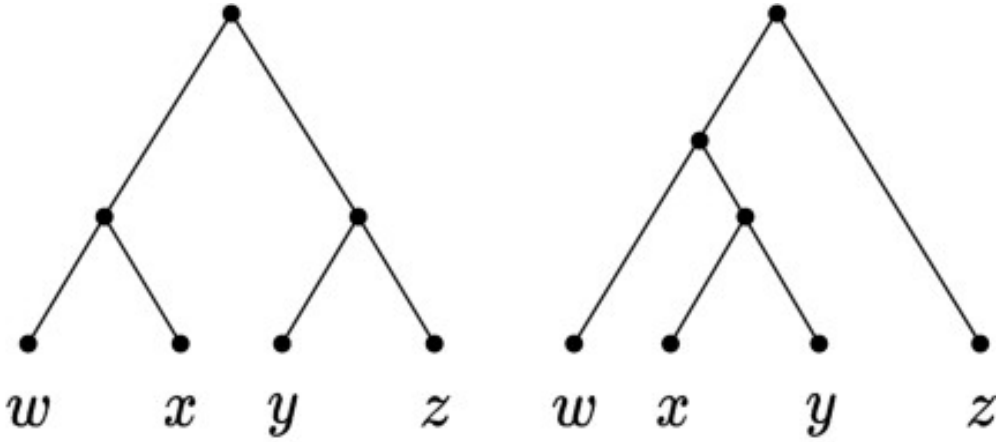
This network has **r(N)=2**.



Definition of a Phylogenetic Tree

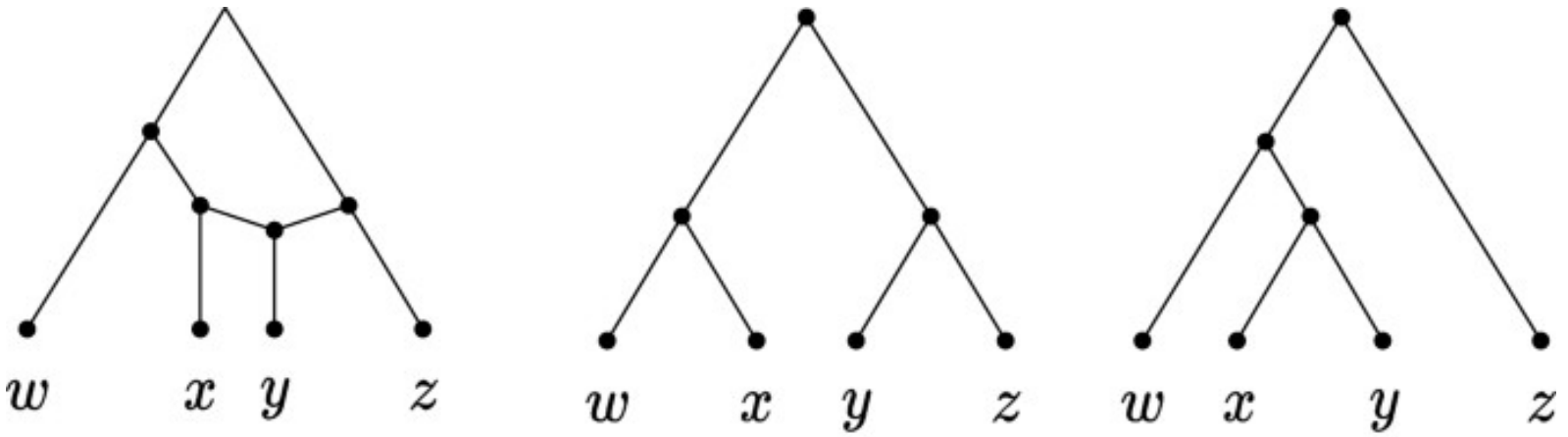
A **Phylogenetic Tree** is a phylogenetic network with **reticulation number 0**: r(N)=0.

Each tree represents an evolutionary tree in biology, with taxa representing species.



Displaying Trees in a Network

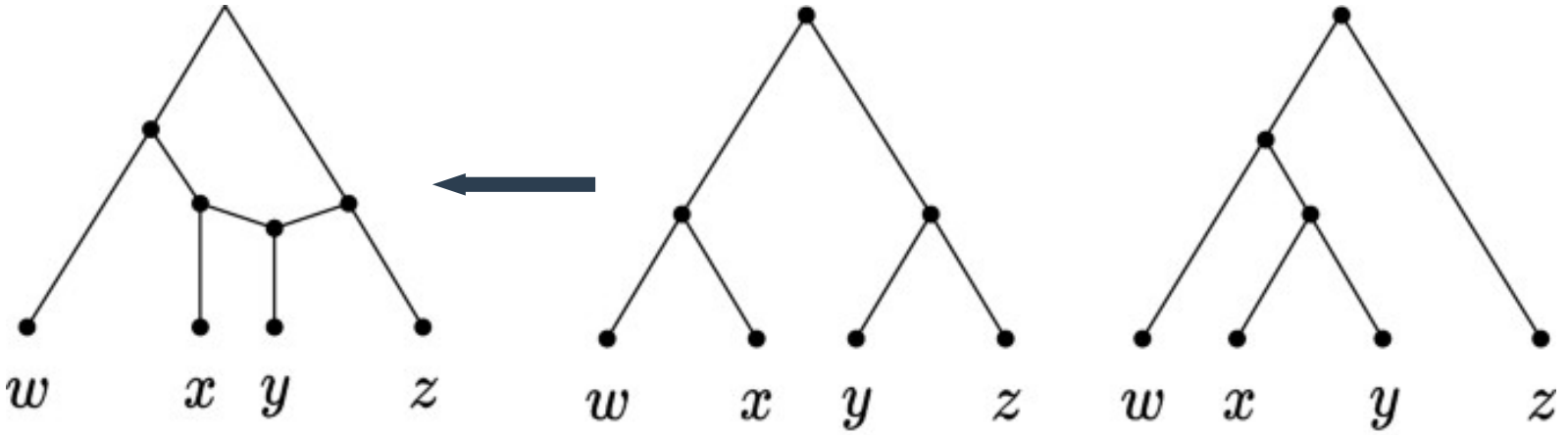
A tree is **displayed** in a network if there is an **embedding** of the tree within the network – i.e. the relationships defined by the tree must be preserved in the network.



Displaying Trees in a Network

A tree is **displayed** in a network if there is an **embedding** of the tree within the network – i.e. the relationships defined by the tree must be preserved in the network.

For example, the below network on the left **displays both** input trees on the right.



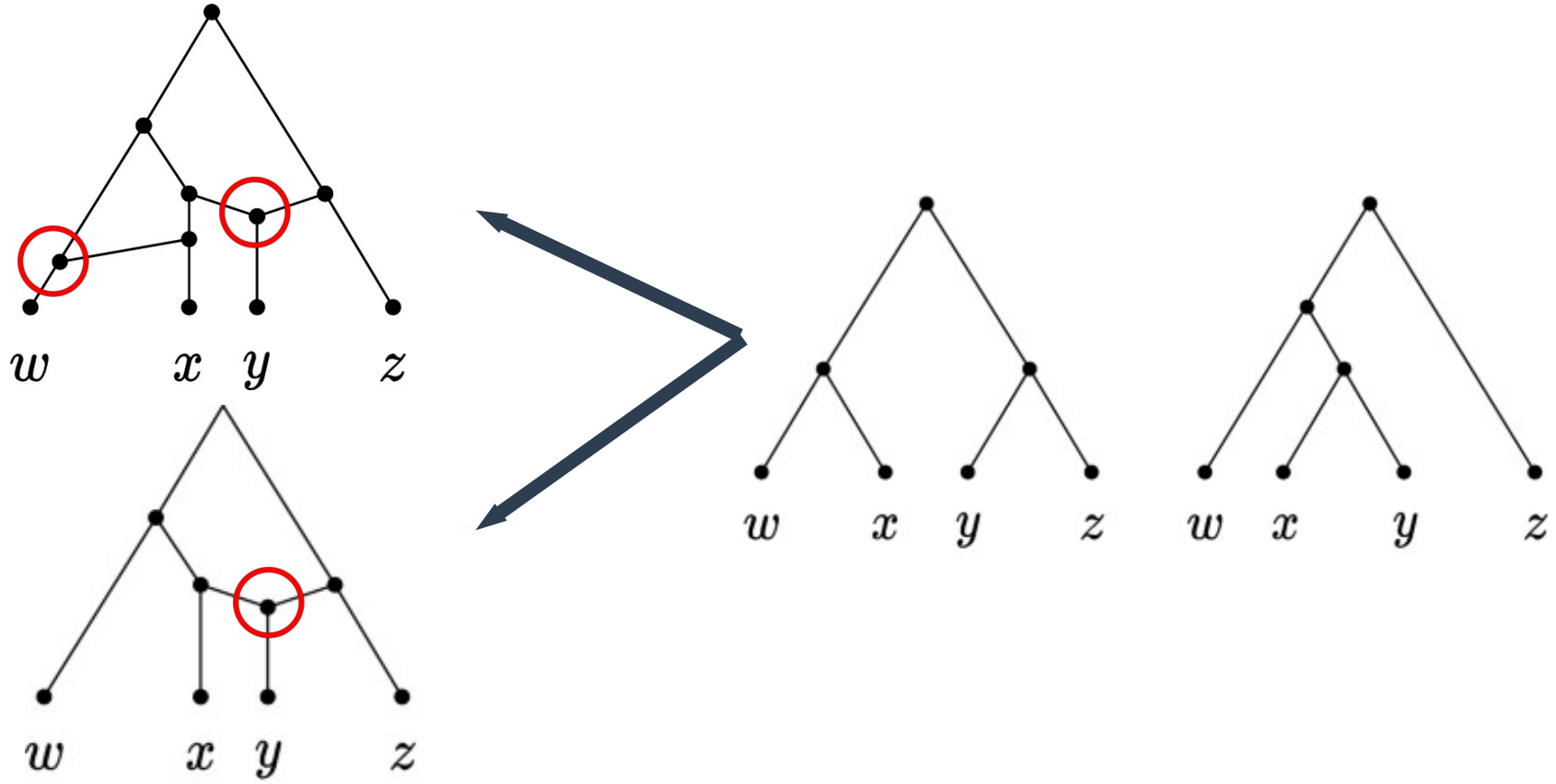
The Goal of the Minimum Hybridization Problem

The **goal** of the **Minimum Hybridization problem**:

To combine a set of phylogenetic trees into a phylogenetic network with **minimum reticulation number** such that they are all displayed.

Example

Both these networks display both trees on the right. However, the top network has **r(N)=2**, while the bottom network has **r(N)=1**.



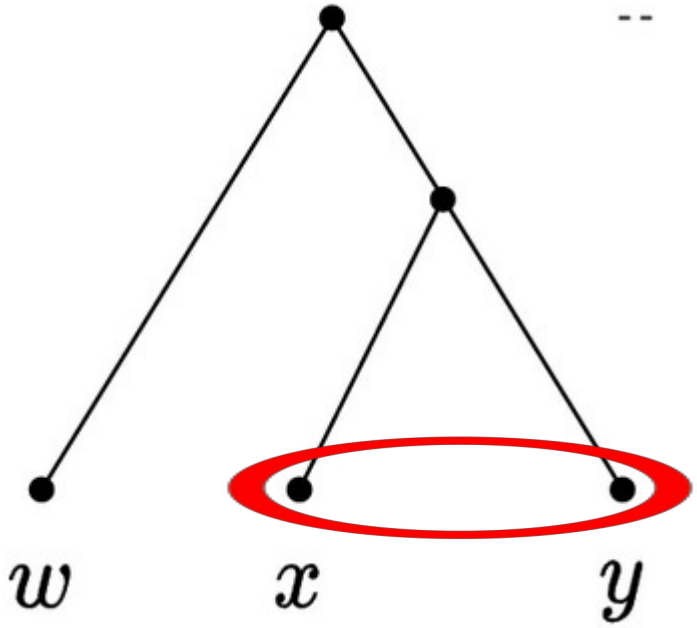
Cherry Picking

Instead of finding the optimal solution, which is NP-Hard, we opt to use a **heuristic: Cherry-picking.**

Cherry-picking works by picking cherries off the tree set and reconstructing it into a network.

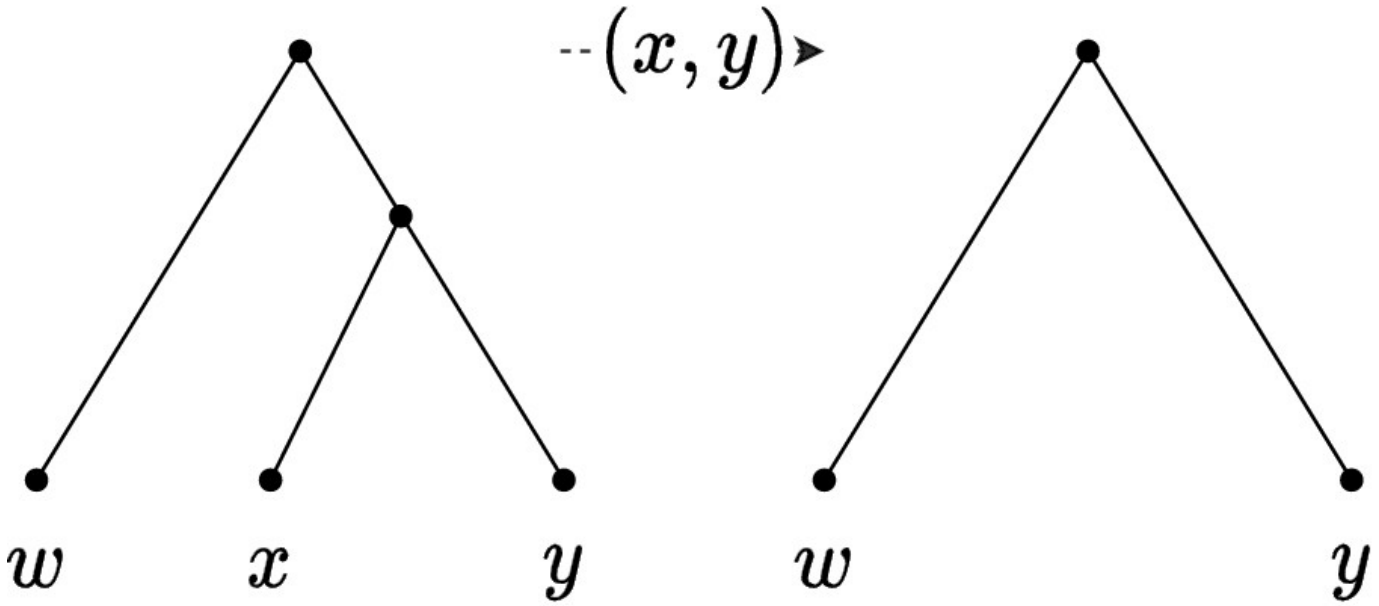
Definition of a Cherry

A **Cherry** is defined as an **ordered pair of leaves, (x,y), x≠y,** where x and y have the **same parent**.



Picking a Cherry

By picking cherries we **reduce the size of the tree set**. We pick them until there are no trees remaining. This gives us a sequence which we can then use to reconstruct a network that displays all the trees in the set.



Heuristics for Cherry-picking

We can use **Supervised Machine Learning** to decide which cherries are **more likely** to reduce the tree set further and create a shorter cherry picking sequence.

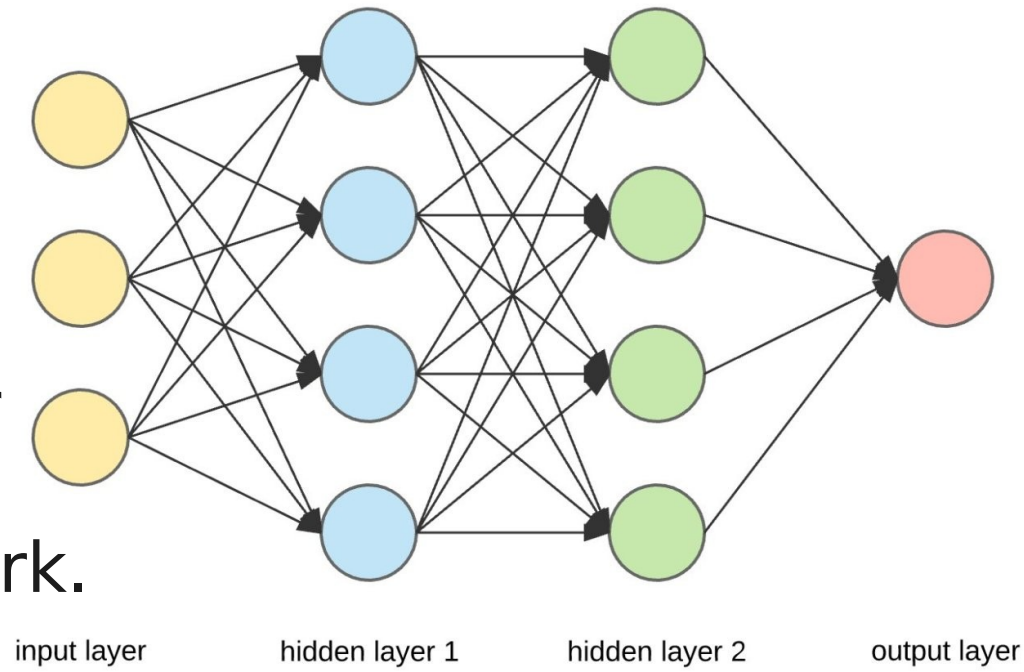
The source paper we looked at used a **Random Forest Classifier.**

Our research investigated the use of a **Neural Network**.

The Neural Network Algorithm

The **Neural Network Algorithm** is a machine learning algorithm which uses the concept of a **neuron** and **layers** to predict an outcome.

However, by changing the number of hidden layers and neurons in each layer, we change the performance of the algorithm for the particular data set. This is the **architecture** of the network.



WHY USE A NEURAL NETWORK?

Theoretically, a neural network can **improve indefinitely** as long as it is given more input data, depending on the architecture it is defined on.

On the other hand, after a certain amount of data, random forests will **no longer improve**. They also do not scale well to large scale training sets.

Fine Tuning Hyper-parameters

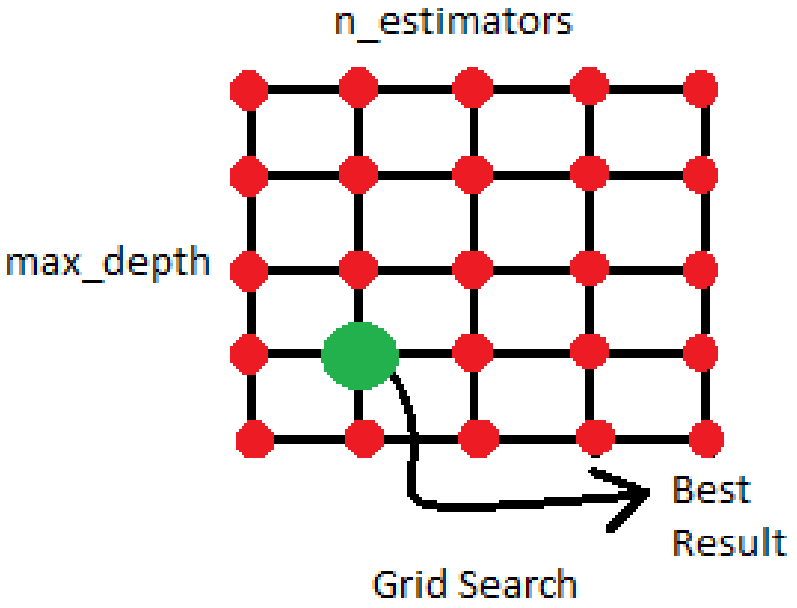
There are a number of **parameters** that can modify the accuracy of the model:

- **Optimizer** and **Optimizer Learning Rate**
- **Activation Function** (how we define weights)
- **Kernel Initializer** (initial weight)
- **No. of Epochs** (how many times we cycle the training data set)
- **Batch Size** (what size of training examples we use at a time)
- **Number of Layers**
- **Number of Neurons in each layer**

Creating a Sample Neural Network

We first used a neural network defined on a small training set of the problem, to try and define some **parameters** of the algorithm.

To achieve this, we used a **Cross Validation Technique**, called **GridSearchCV**, which can iterate through a series of parameter combinations and returns the optimal configuration.



Generating the Large Train Set

In order to create a model that can be compared to the source paper’s model, we generated a data set of 1,000 networks to be used for training.

GridSearchCV was then used to find a good configuration of neurons and layers. A few were similar in performance in terms of accuracy, but an architecture of **3 layers of 60, 70 and 4 neurons** was chosen.

Testing the Model

The algorithm was then trained on the data set of 1,000, and finally tested on **1,296 different tree sets**.

We also tested the performance of the random forest algorithm to compare to the Neural Network.

The results are still computing.

Resources

The Original Paper:

https://almob.biomedcentral.com/articles/10.1186/s13015-023-00233-3#Sec10

The Source Code:

https://github.com/estherjulien/learn2cherrypick/tree/master

Packages used:

Numpy, Tensorflow, Networkx, Pandas, Scikeras, Scikit-Learn