Theory

One strategy to levy the flexibility and power of neural networks when faced with a classification problem and small data sets, is to make a so called siamese network. Such networks have shown good results when employed in image recognition(ref Hadsell, FaceNet). The basic premise of these networks is that the feature space is mapped to some metric space where the images of input vectors can be can be compared. Vectors whos images are “close” in the given metric can be thought of as similar. The goal then, is to train the network to map the different categories to different “areas” of the metric space and, in a sense, learn the concept of sameness.

The name siamese network stems from the structuring of the network. It consists of a base network, wich can have any structure, that acts as a function mapping input vectors into an N dimensional space, where N is the number of output nodes for the base network. For a twin network, two (three for triplets) instances of this base network are created with shared weights and biases. The network can thus take two (or three) different vectors as input and the conjoined base networks will output a corresponding number of N dimensional representations. Then follows a merging layer where the distances between the images of the input vectors are calculated. This output is then fed to a layer that outputs a similarity score between each vector.

One method to train a twin network is to first pick one representative, referred to as an anchor, from each class. Then, for each anchor a\_i and each datapoint x\_j pair (a\_i,x\_j) is created. If the pair belong to the same category they are labeled 1 and if they belong to different categegories the pair receives the label 0. These pairs, along with their labels, will then serve as training data for the network. The network is then updated according to a loss function that pushes inputs from different classes apart while clustering similar inputs. Hadsell et al proposes the contrastive loss function:

CONTRASTIVE LOSS y(dist(a,x)^2+(1-y)(max(m-dist(a,x),0))^2

This is basicaly two different loss functions in one. If the the label y = 1, then the distance is minimized, if y=0, it is maximized up to a margin m.

The method is somewhat similar for triplet networks. For each data point x\_i, pick one representative from the same category, a\_s, and one representative from a different category, a\_d. Update the shared network according to the triplet loss function from FaceNet:

Triplet loss max(dist(a\_s,x\_i)^2 – dist(a\_d,x\_i)^2 +m , 0)

Again, m is a margin beyond wich the network will stop updating.

Results:

For the siamese network approach, the results were, on the whole, disappointing. While the networks certainly showed signs of learning, getting good results on the training data, they displayed little to no ability to generalize. Attempts to alleviate overfitting by adding dropout layers or adjusting hyper parameters, did not lead to improvements on predictive ability beyond the level of random guessing.

With our choice of data set, it is not obvious whether this failure is due to the methods employed or a lack of significant correlation between the predictors, microbial community composition, and the various target variables such as phosphate concentrations or temperature at the sample sites.

Sources:

Hadsell et al 2006

http://yann.lecun.com/exdb/publis/pdf/hadsell-chopra-lecun-06.pdf

FaceNet

https://arxiv.org/abs/1503.03832v3