Embedding Self-Organizing Maps into Neural Networks

Stephen Barnes
stbarnes@stanford.edu

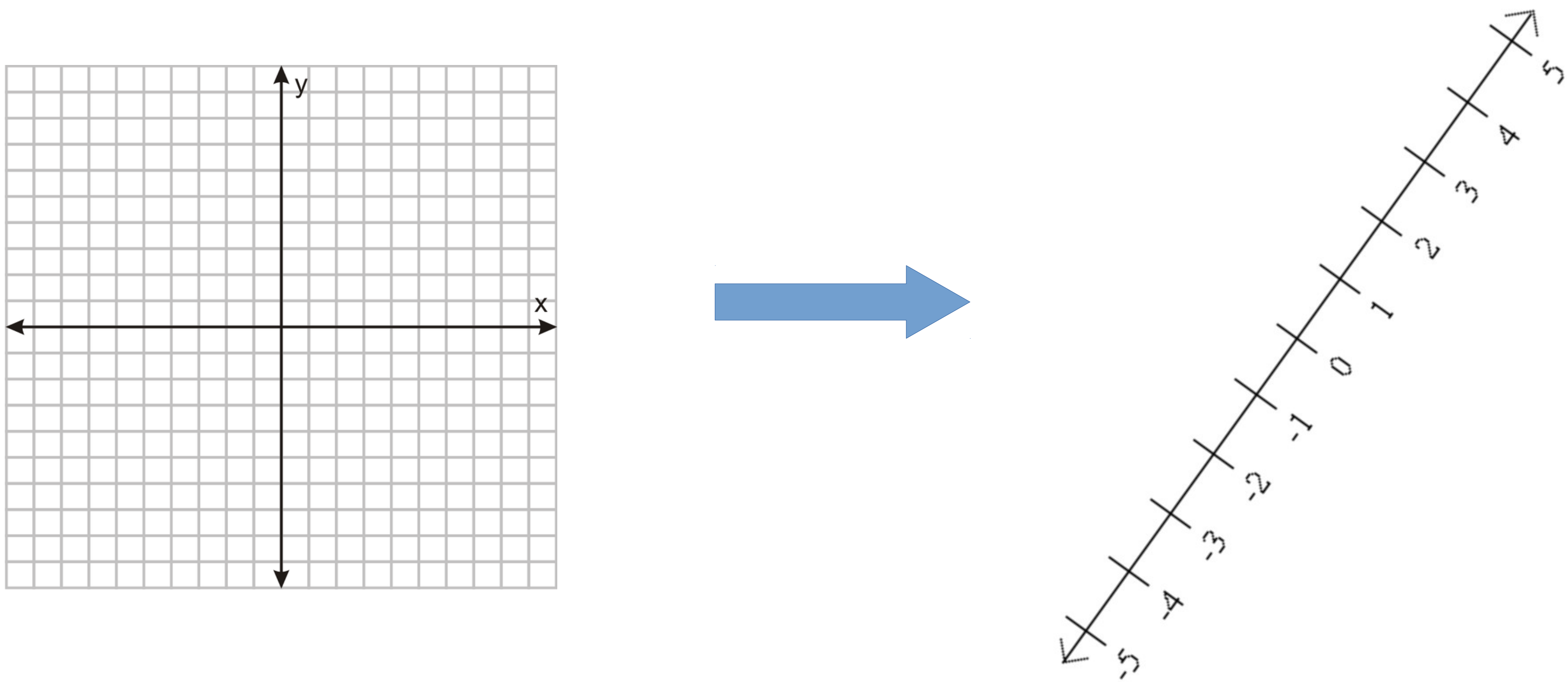
Stanford University

What is a Self-Organizing Map?

What is a Self-Organizing Map?

A mapping from R^n to R^m

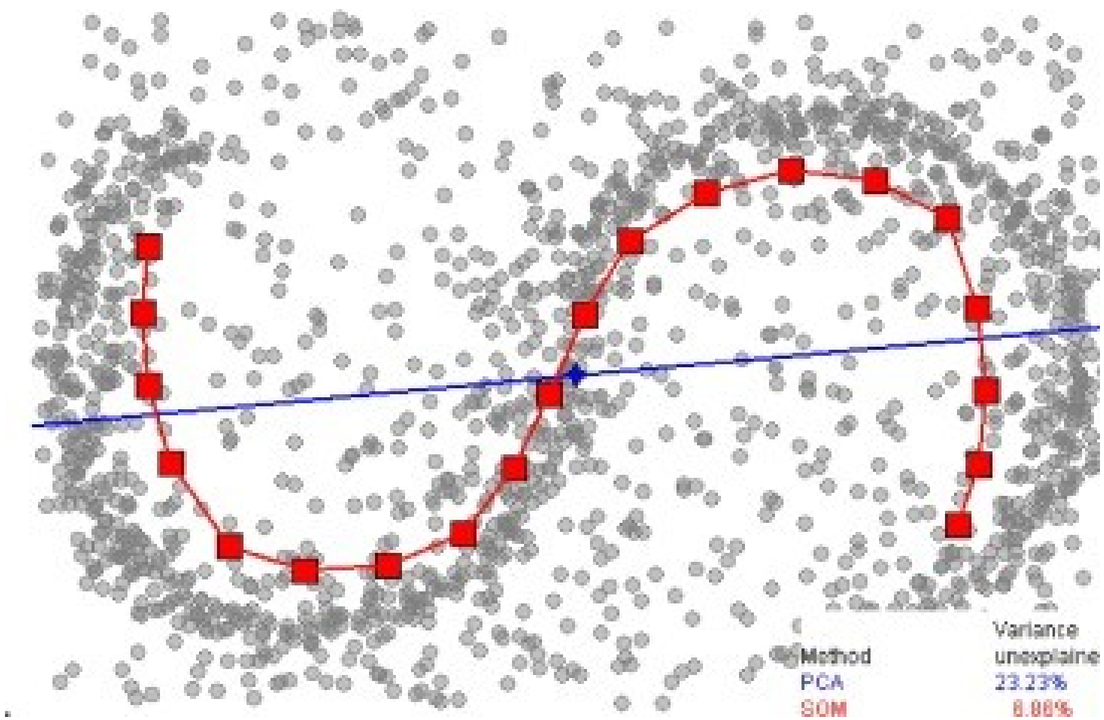
Dimensionality reduction



What is a Self-Organizing Map?

"Self-organizing" – mapping learned from data

Mapping can be nonlinear (red), unlike PCA (blue)

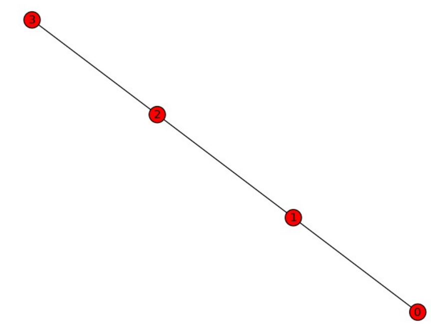


$R^2 \rightarrow R^1$

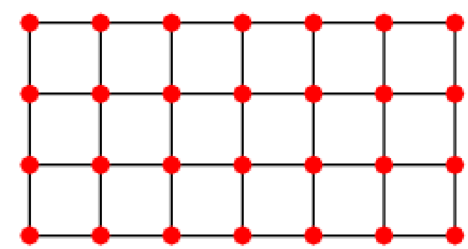
Learning an SOM mapping $R^n \rightarrow R^m$

Learning an SOM mapping $R^n \rightarrow R^m$

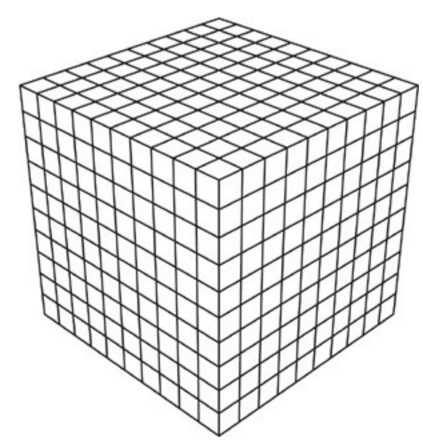
Start with an m-dimensional grid graph



m= 1



m= 2

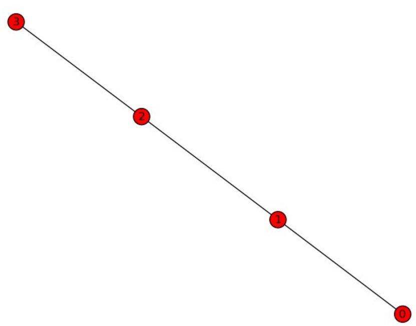


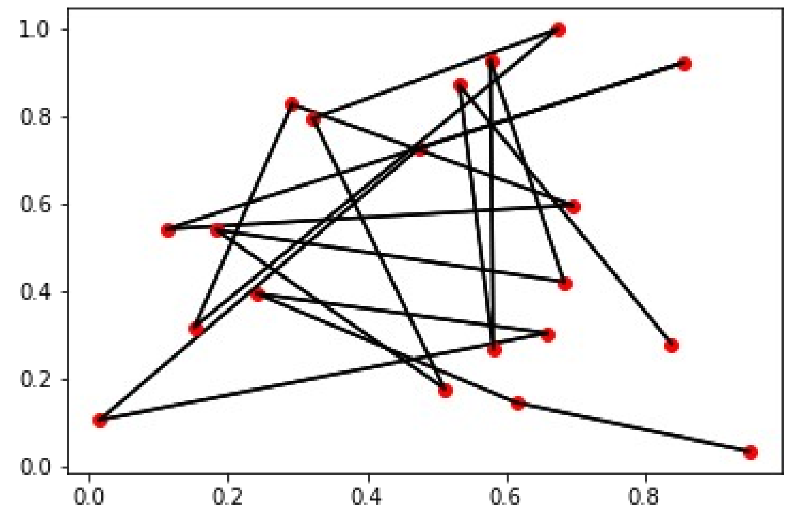
m= 3

Learning an SOM mapping $R^n \rightarrow R^m$

Embed into input space R^n randomly

Each node n has location l_n in R^n

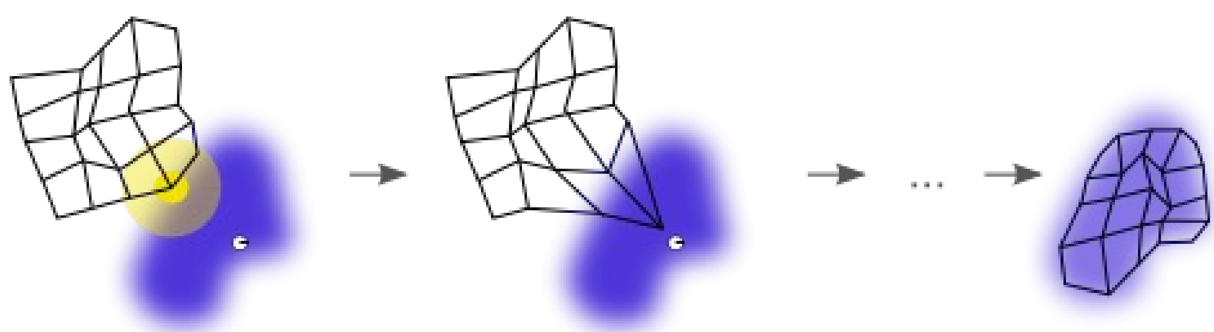




Learning an SOM mapping $R^n \to R^m$

Show each point to the SOM

When SOM is shown a point, nearby graph nodes move closer

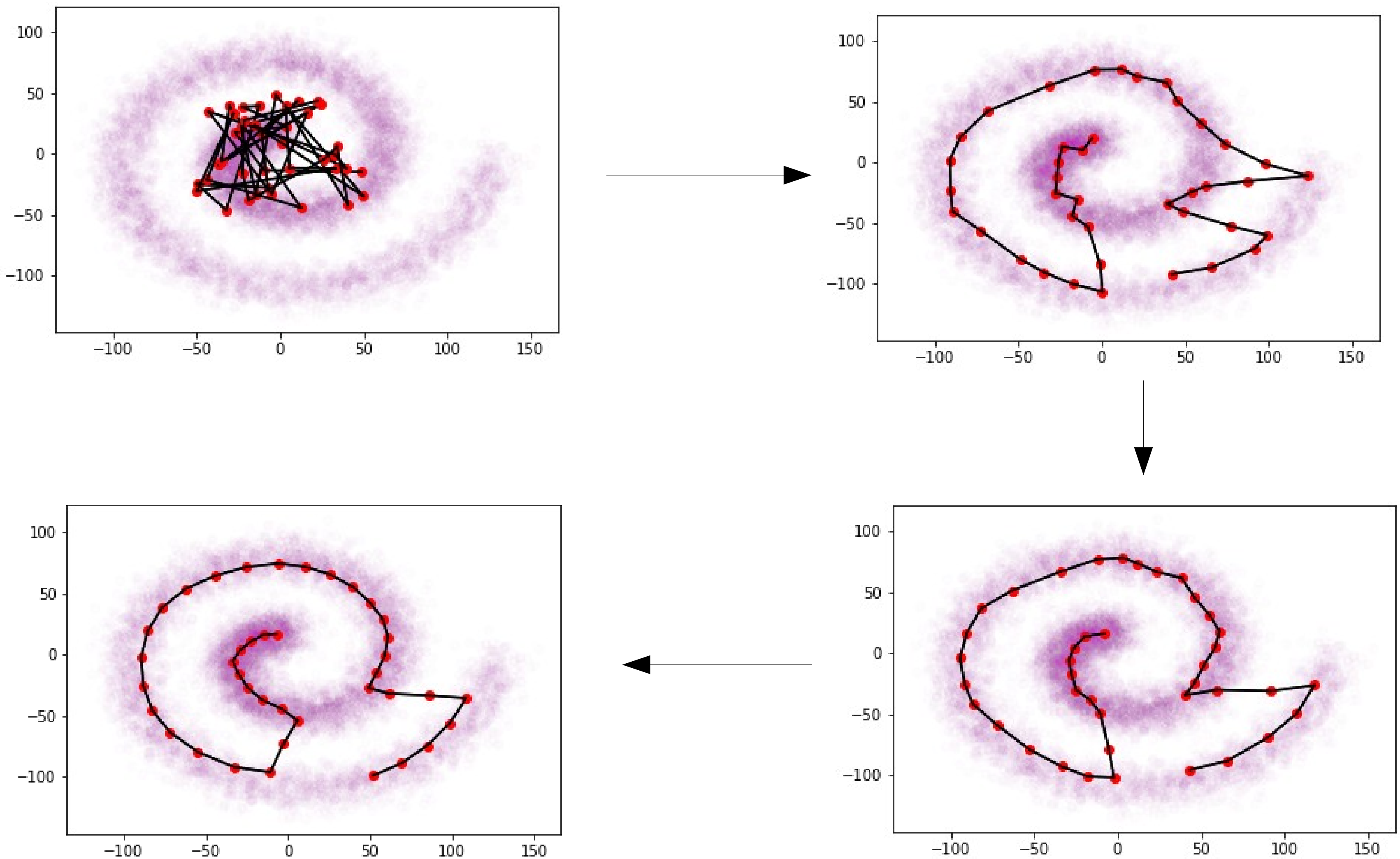


$R^2 \to R^2$

Over many steps, SOM graph copies input distribution

Learning an SOM mapping $R^n \rightarrow R^m$

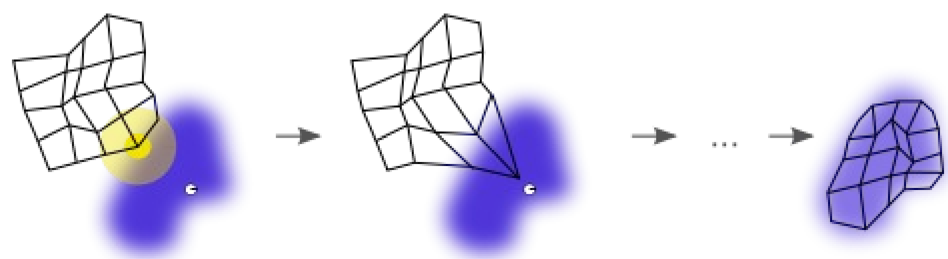
$R^2 \rightarrow R^1$



Updating the graph embedding

When shown a sample, the closest graph node is the “winner” - competitive learning

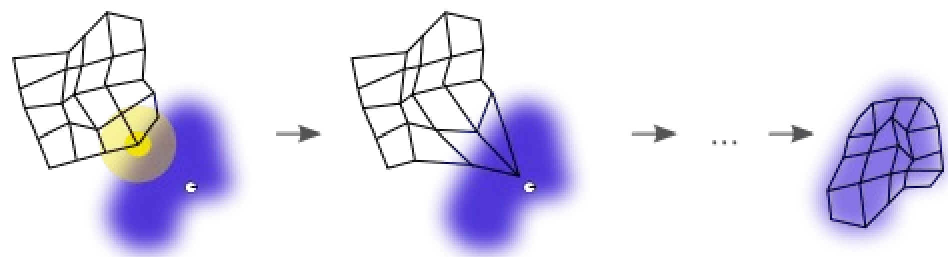
Winning node and nearby nodes move closer



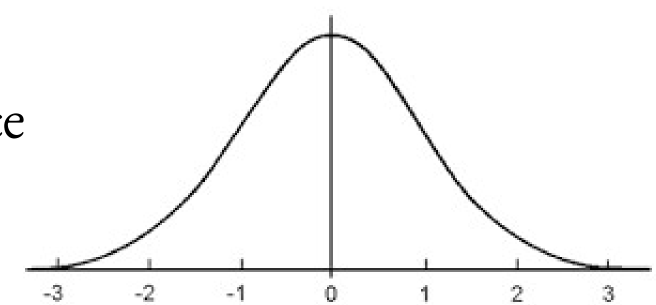
Updating the graph embedding

When shown a sample, the closest graph node is the “winner” - competitive learning

Winning node and nearby nodes move closer



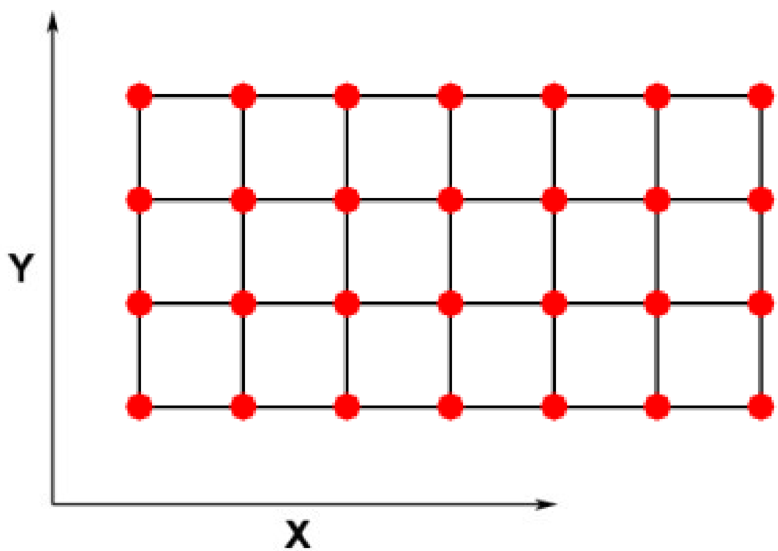
Die-off is exponential in graph distance

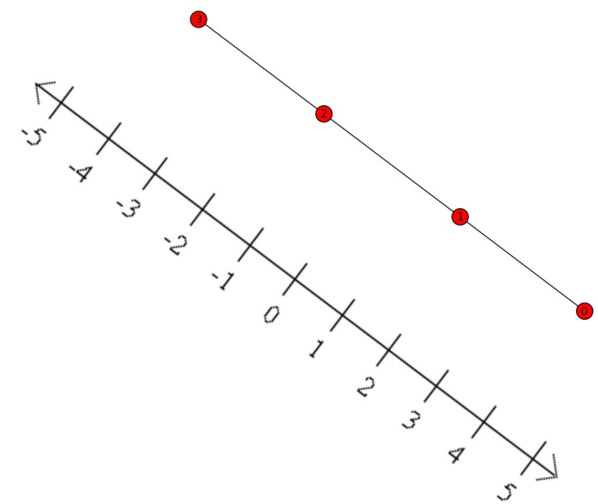


Learning rate decays with samples seen

Learning an SOM mapping $R^n \to R^m$

m-dimensional grid graph defines an m-dimensional “graph space”





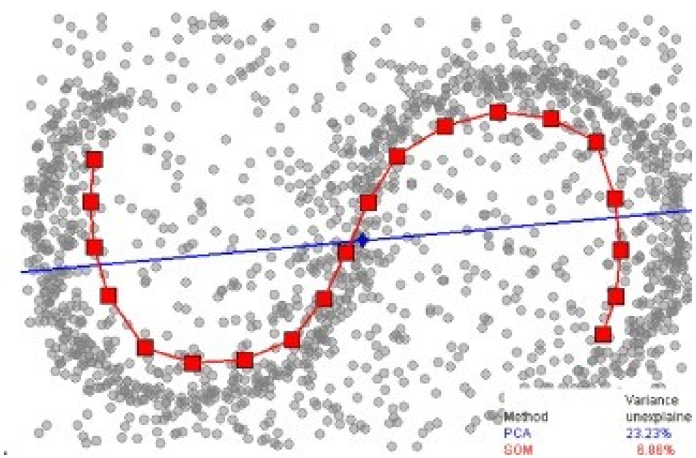
SOM outputs graph-space coordinates of winning node

Learning an SOM mapping $R^n \rightarrow R^m$

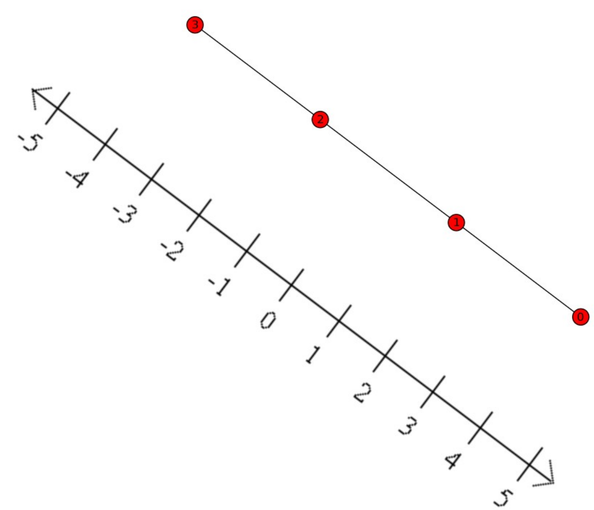
SOM translates input point to coordinate of nearest grid point in graph space

SOM "unwinds" 1D manifold in R^2 to a line in R^1

Input points in R^2



Graph-space points in R^1

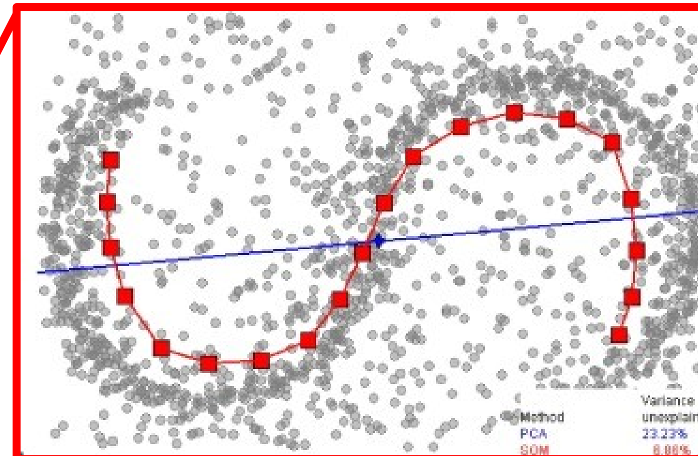


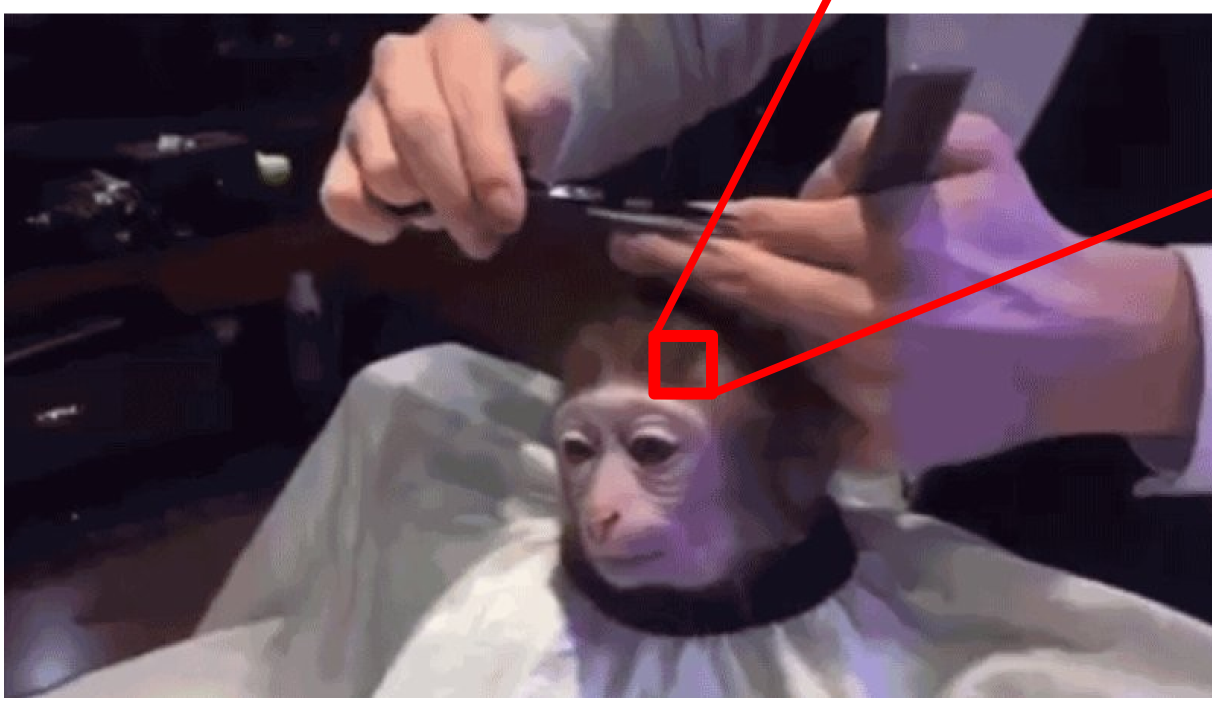
History of SOMs

Designed by Prof. Teuvo Kohonen
in the 1980s

History of SOMs

Building on 1970s models from neuroscience
and morphogenesis models from 1950s



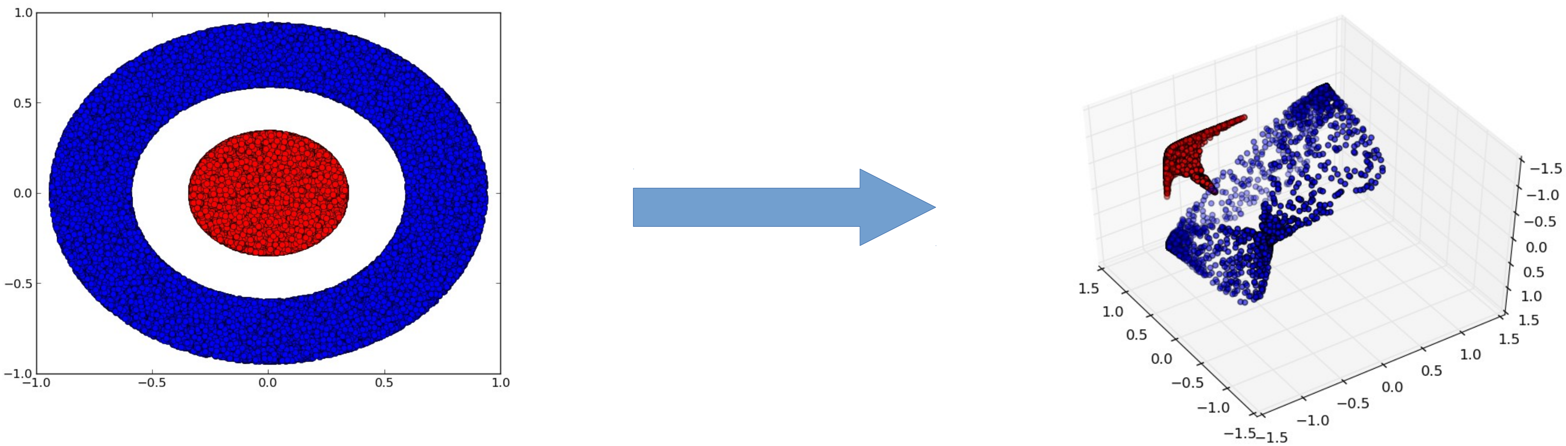


Pictured:
1970s neuroscience models

Inserting SOMs into NNs

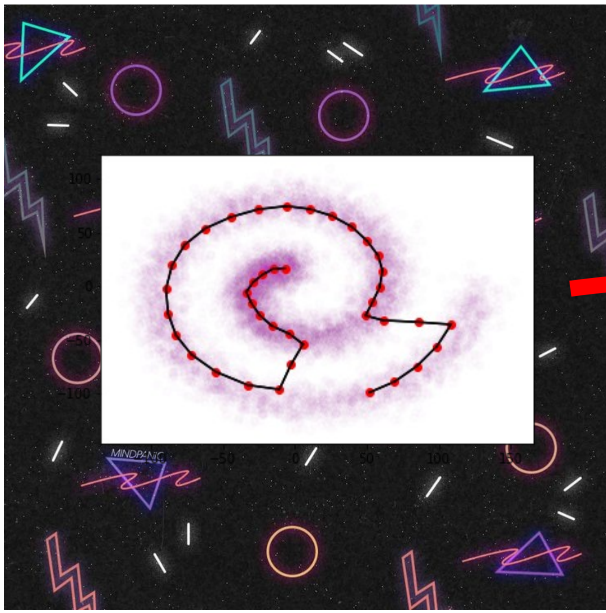
Neural networks transform manifolds to make categories separable

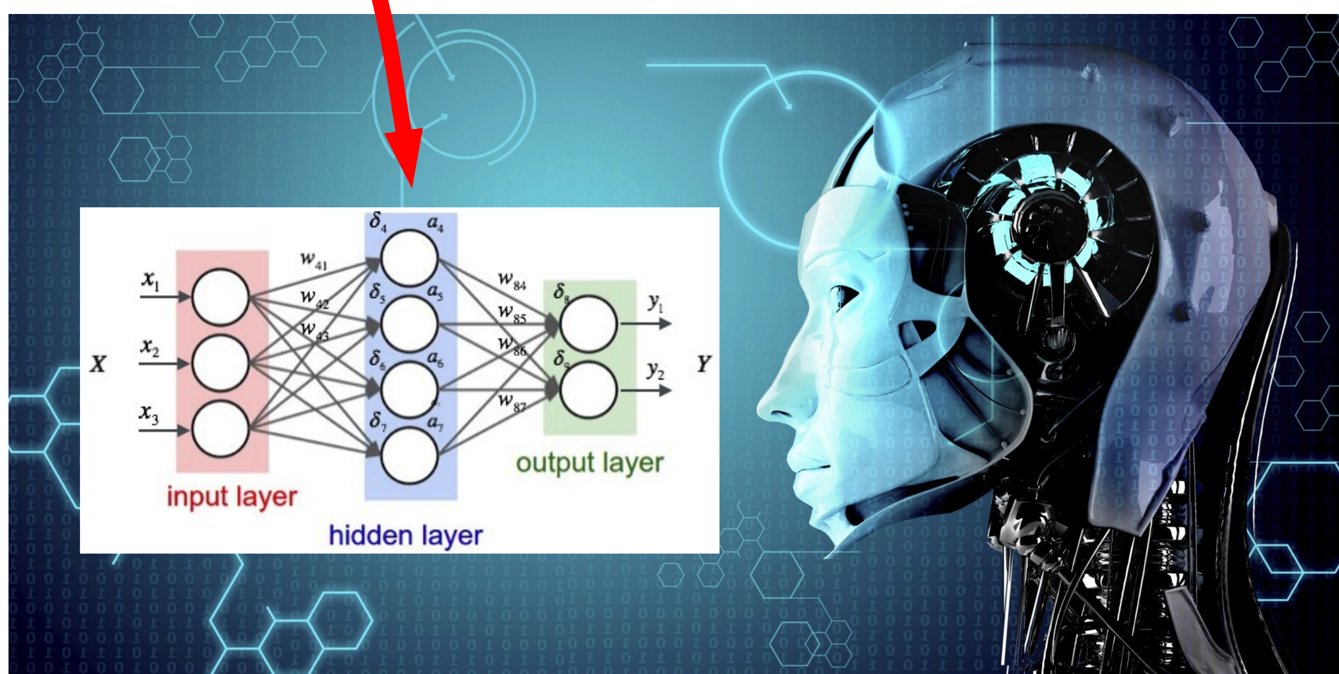
Maybe SOMs can do this better?



Pictures by Chris Olah
http://colah.github.io/posts/2014-03-NN-Manifolds-Topology/

Inserting SOMs into NNs

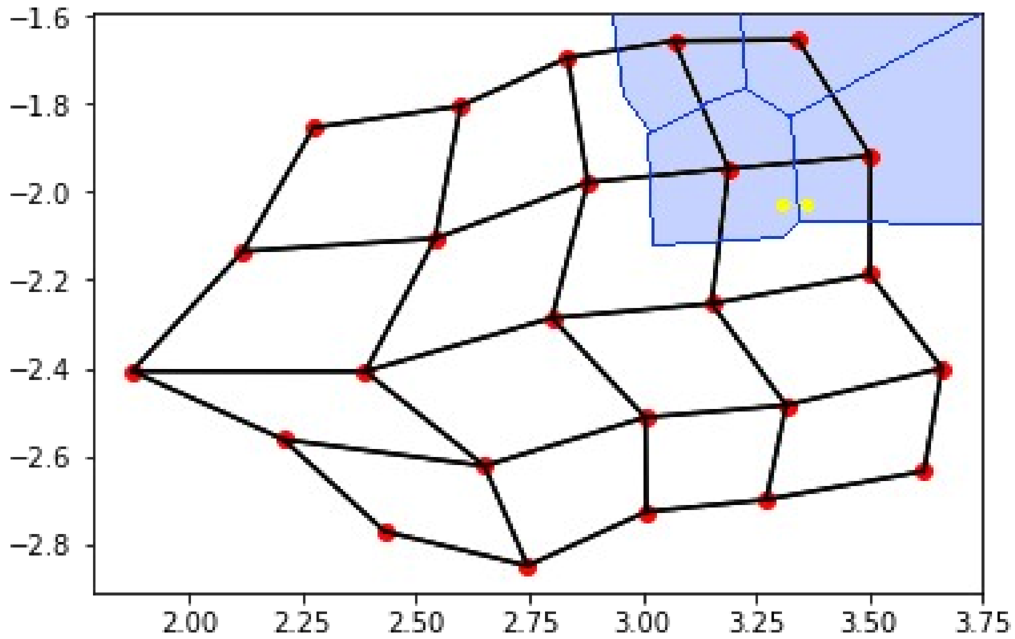




Inserting SOMs into NNs

Problem: SOMs are non-differentiable! Backpropagation is impossible

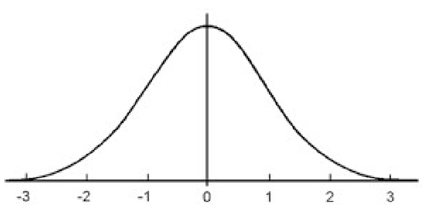
Nodes win all input points inside Voronoi cell – piecewise constant output



Inserting SOMs into NNs

Solution: every node wins, but some win more than others

Output is weighted mean of nodes' graph locations,
weights decaying with distance to input point



Average of graph positions

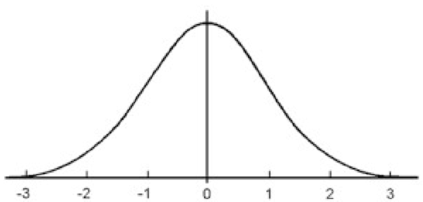
$$\text{Output}(p) = \frac{\sum_v g_v \overbrace{\exp(-\beta \, \|p - l_v\|_2)}^{\text{Weights decaying with distance}}}{\underbrace{\sum_v \exp(-\beta \, \|p - l_v\|_2)}_{\text{Normalized by sum of weights}}}$$

Output of
SOM

Inserting SOMs into NNs

Solution: every node wins, but some win more than others

Output is weighted mean of nodes' graph locations, weights decaying with distance to input point



Graph position of node v

Decay parameter

$$\text{Output}(p) = \frac{\sum_v g_v \exp(-\beta \, \|p - l_v\|_2)}{\sum_v \exp(-\beta \, \|p - l_v\|_2)}$$

Output of SOM

Input point

Embedding location of node v

Inserting SOMs into NNs

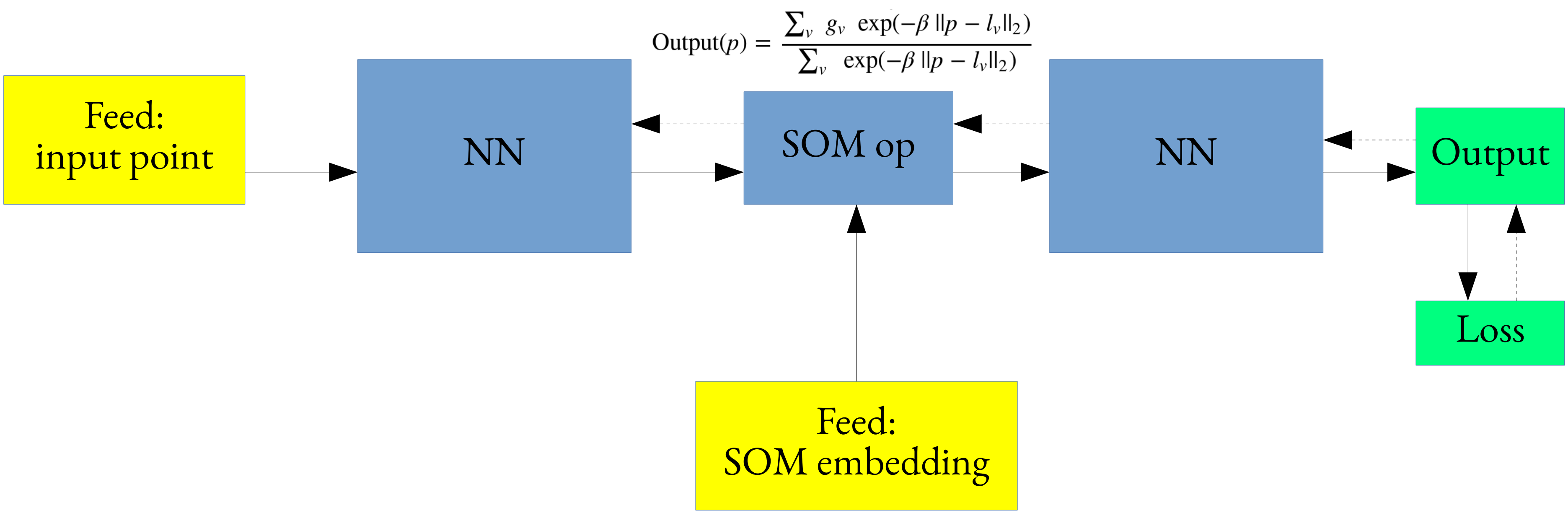
If node embedding is constant,
this is a differentiable function of *p*.

Derivatives can flow back through SOM!

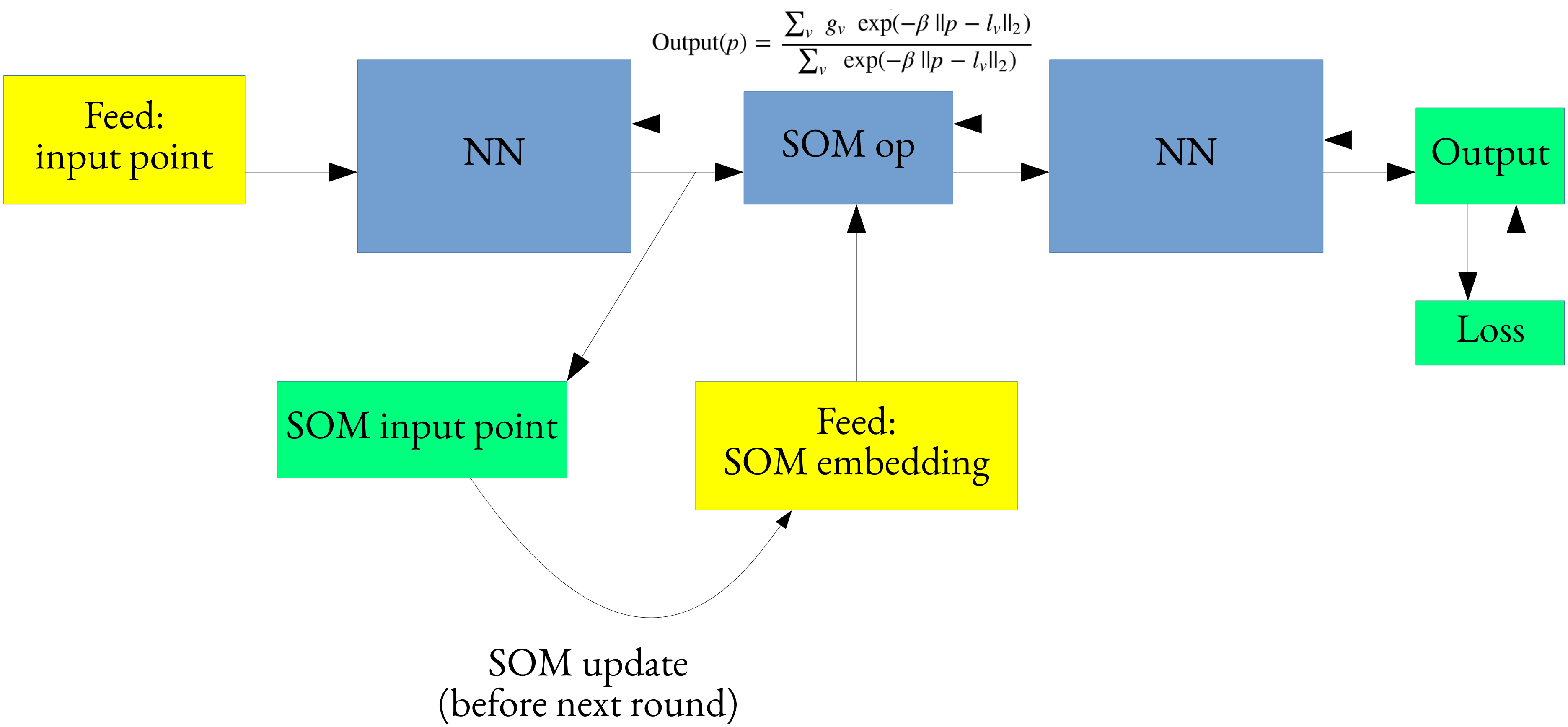
$$\mathrm{Output}(p) = \frac{\sum_v \; g_v \; \exp(-\beta \, \|p - l_v\|_2)}{\sum_v \; \exp(-\beta \, \|p - l_v\|_2)}$$

Implemented as TensorFlow op,
with embedding as TF input.

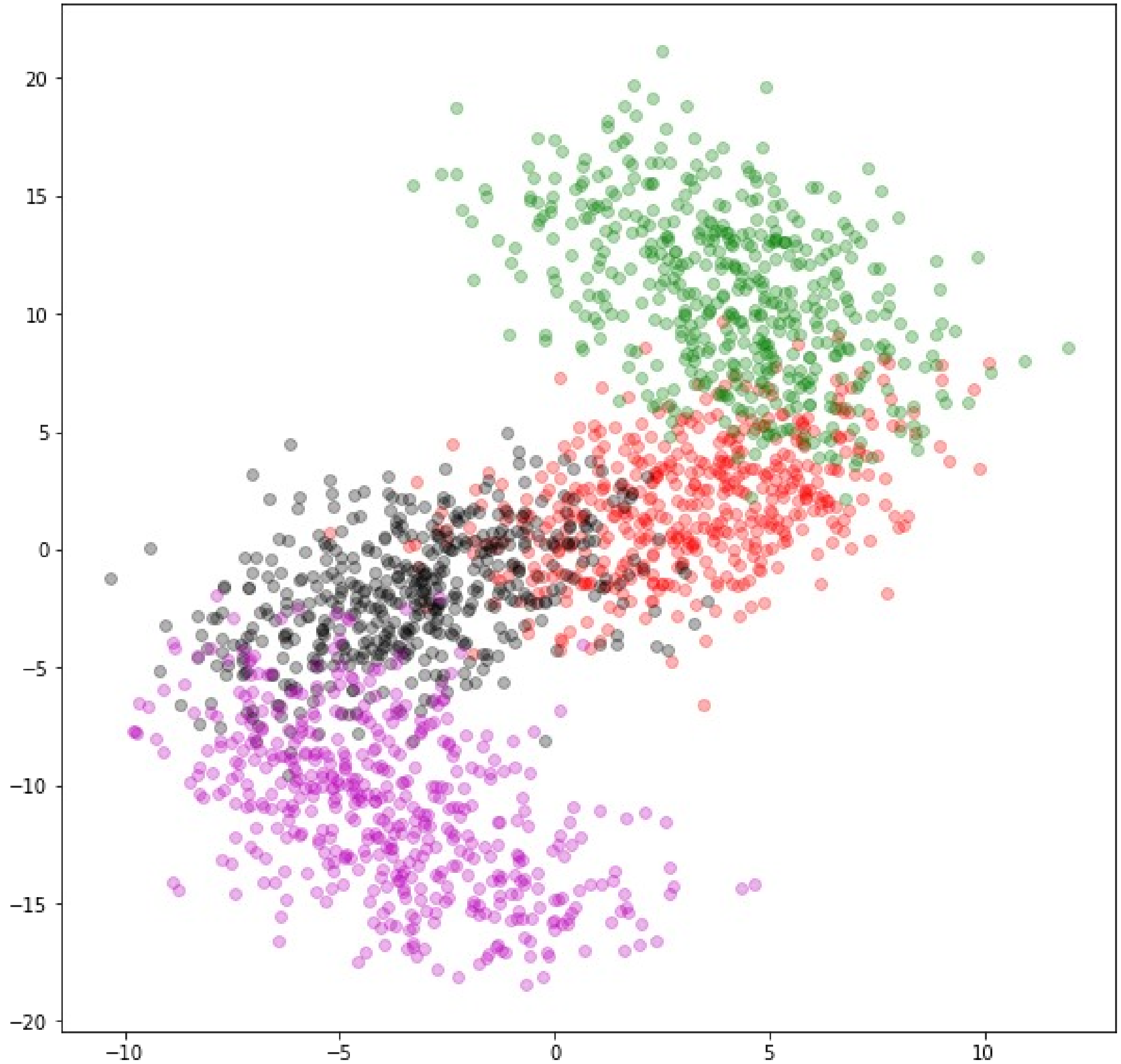
TensorFlow structure



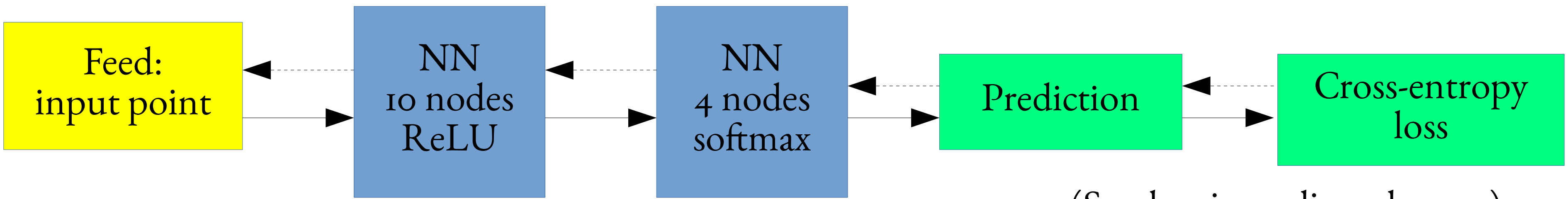
TensorFlow structure



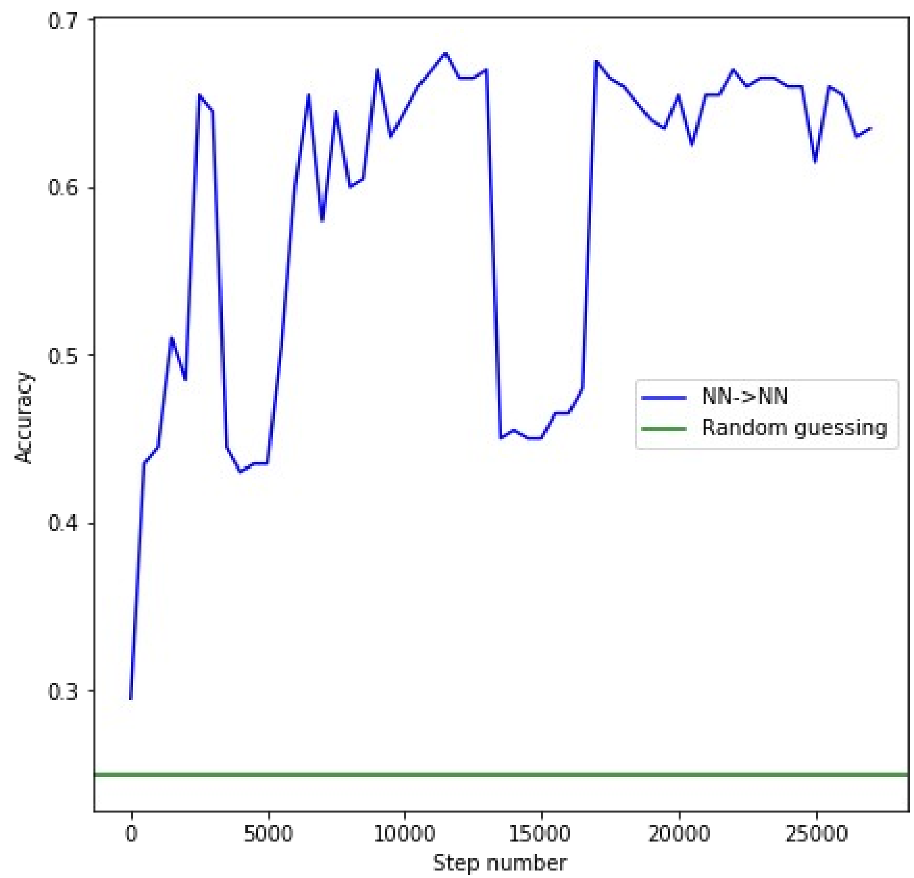
Testing: spiral classification task



Results: pure NN

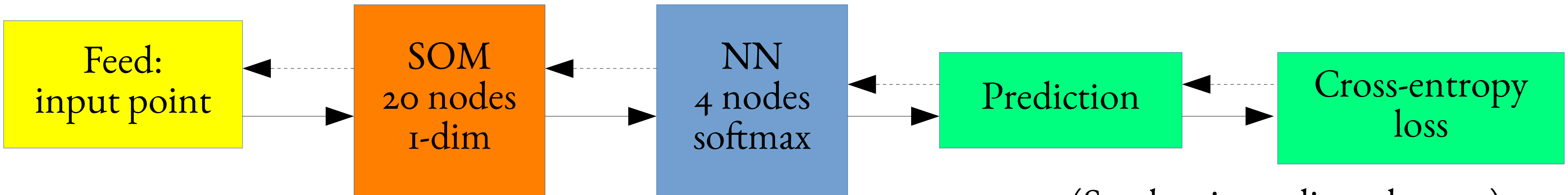


(Stochastic gradient descent)

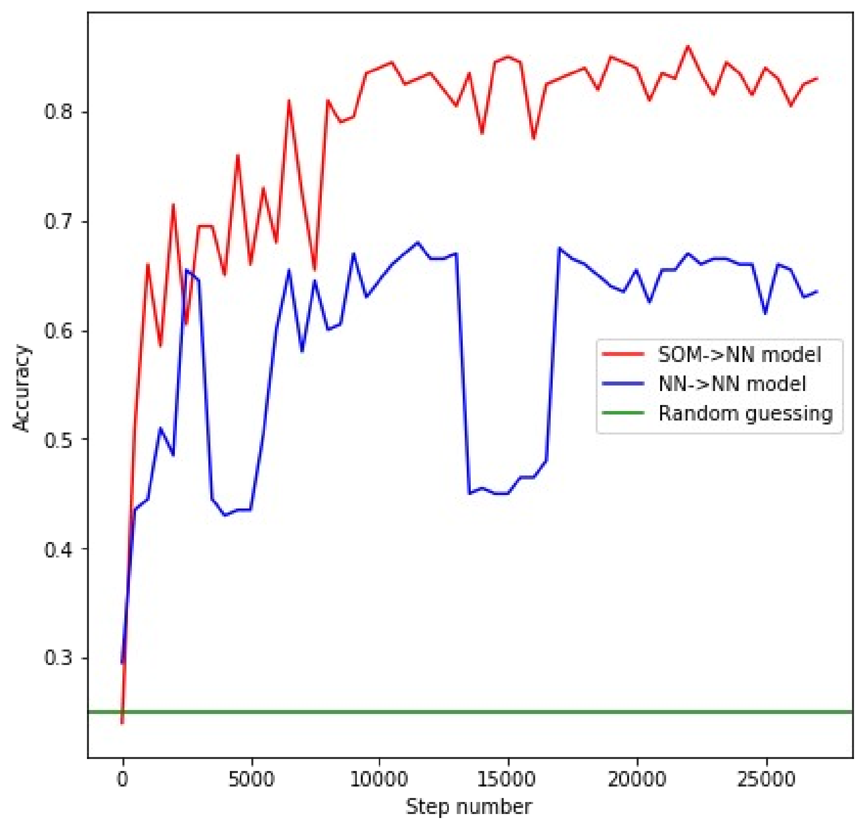


~65% accuracy

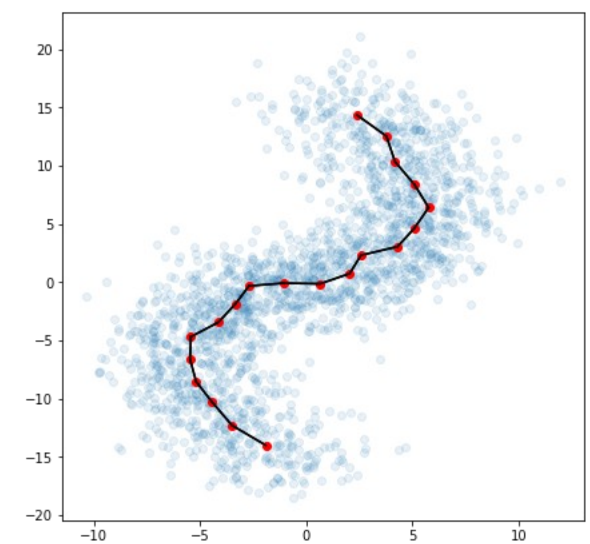
Results: SOM preprocessing



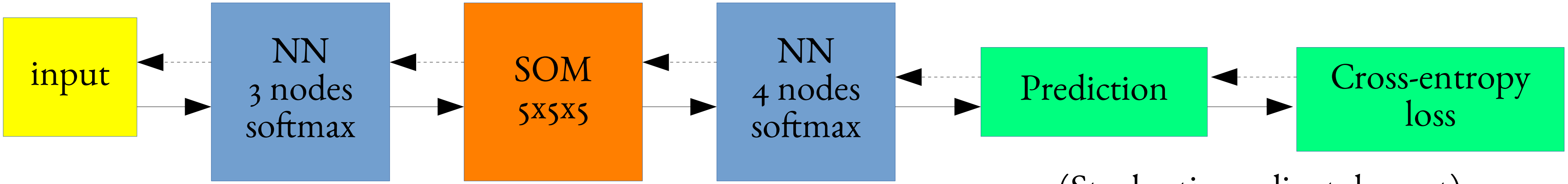
(Stochastic gradient descent)



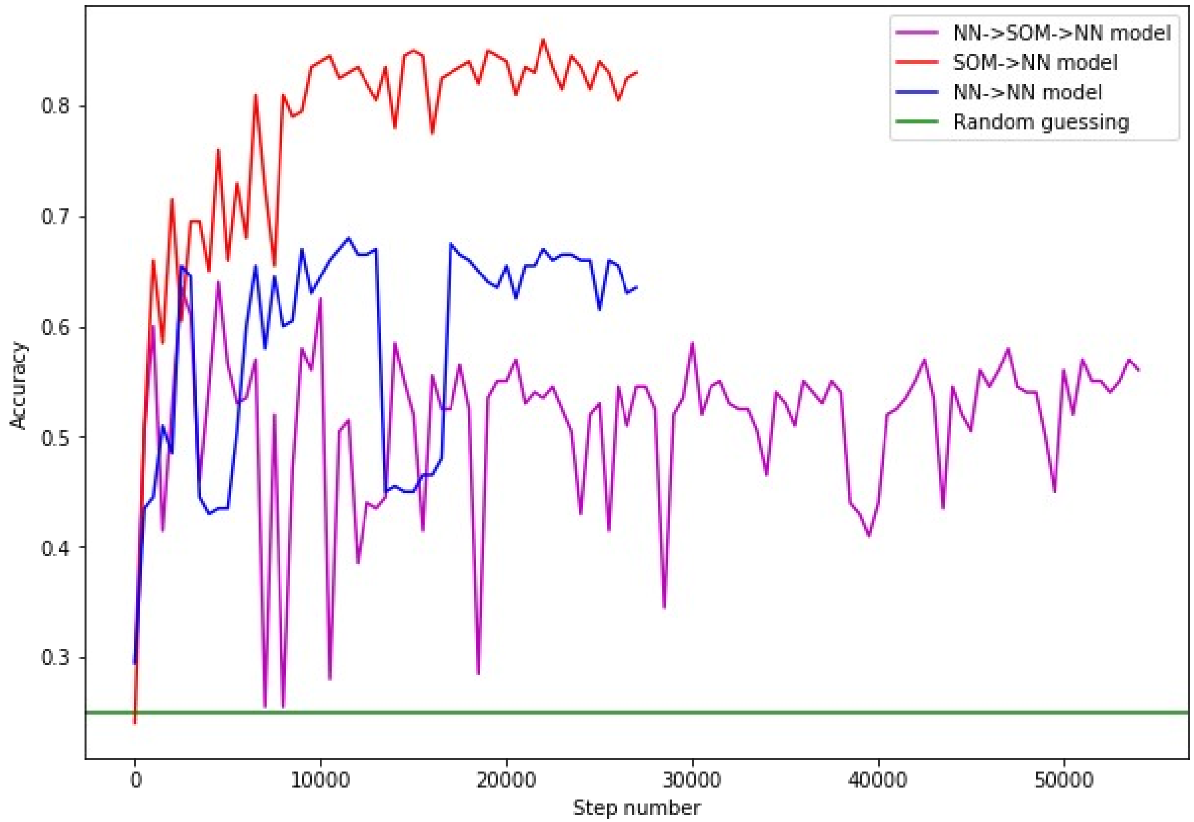
~85% accuracy



Results: interposed SOM



(Stochastic gradient descent)



~55% accuracy

Q&A