# Kohohen’s SOMs

The article of R.N.G. Naguid et al. called *The detection of nodal metastasis in breast cancer using neural network techniques* used a neural network based on Kohohen’s self-organizing maps (SOMS). The architecture of thenetwork consisted of a two-dimensional Kohonen layer acting as a 6 x 6 SOM followed by a back-propagation unit consisting of one hidden layer and one output neuron. The hidden layer consists of two neurons which, together with the output neuron, employ either a delta learning rule and a sigmoid transfer function, or an extended delta-bardelta learning rule and a hyperbolic tangent transfer function, depending upon the marker group under analysis.

Wikipedia:  
A **self-organizing map** (**SOM**) or **self-organizing feature map** (**SOFM**) is a type of [artificial neural network](https://en.wikipedia.org/wiki/Artificial_neural_network) (ANN) that is trained using [unsupervised learning](https://en.wikipedia.org/wiki/Unsupervised_learning) to produce a low-dimensional (typically two-dimensional), discretized representation of the input space of the training samples, called a **map**, and is therefore a method to do [dimensionality reduction](https://en.wikipedia.org/wiki/Dimensionality_reduction). Self-organizing maps differ from other artificial neural networks as they apply [competitive learning](https://en.wikipedia.org/wiki/Competitive_learning) as opposed to error-correction learning (such as [backpropagation](https://en.wikipedia.org/wiki/Backpropagation) with [gradient descent](https://en.wikipedia.org/wiki/Gradient_descent)), and in the sense that they use a neighborhood function to preserve the [topological](https://en.wikipedia.org/wiki/Topology) properties of the input space.   
This makes SOMs useful for [visualization](https://en.wikipedia.org/wiki/Scientific_visualization) by creating low-dimensional views of high-dimensional data, akin to [multidimensional scaling](https://en.wikipedia.org/wiki/Multidimensional_scaling). The artificial neural network introduced by the [Finnish](https://en.wikipedia.org/wiki/Finland) professor [Teuvo Kohonen](https://en.wikipedia.org/wiki/Teuvo_Kohonen) in the 1980s is sometimes called a **Kohonen map** or **network**.[[1]](https://en.wikipedia.org/wiki/Self-organizing_map#cite_note-KohonenMap-1)[[2]](https://en.wikipedia.org/wiki/Self-organizing_map#cite_note-2) The Kohonen net is a computationally convenient abstraction building on biological models of neural systems from the 1970s[[3]](https://en.wikipedia.org/wiki/Self-organizing_map#cite_note-3) and [morphogenesis](https://en.wikipedia.org/wiki/Morphogenesis) models dating back to [Alan Turing](https://en.wikipedia.org/wiki/Alan_Turing) in the 1950s.[[4]](https://en.wikipedia.org/wiki/Self-organizing_map#cite_note-4)

**The Algorithm:**

1. Each node’s weights are initialized.
2. A vector is chosen at random from the set of training data.
3. Every node is examined to calculate which one’s weights are most like the input vector. The winning node is commonly known as the **Best Matching Unit** (BMU).
4. Then the neighbourhood of the BMU is calculated. The amount of neighbors decreases over time.
5. The winning weight is rewarded with becoming more like the sample vector. The nighbors also become more like the sample vector. The closer a node is to the BMU, the more its weights get altered and the farther away the neighbor is from the BMU, the less it learns.
6. Repeat step 2 for N iterations.

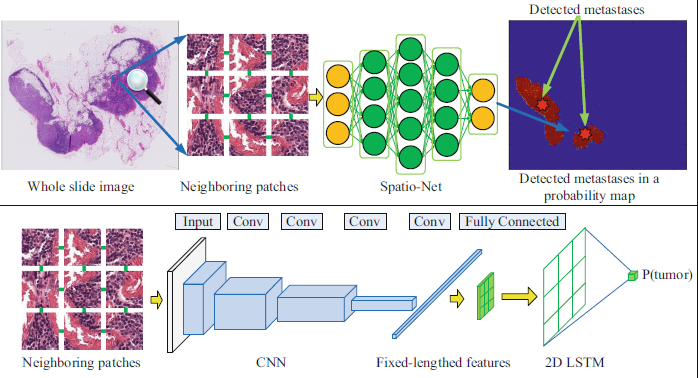
**Best Matching Unit** is a technique which calculates the distance from each weight to the sample vector, by running through all weight vectors. The weight with the shortest distance is the winner. There are numerous ways to determine the distance, however, the most commonly used method is the [Euclidean Distance,](https://en.wikipedia.org/wiki/Euclidean_distance) and that’s what is used in the following implementation.

**Implementation:**

Coming to implementation part, there are various Python libraries (minisom, sompy) out there which you could directly use to implement SOM. You could also write your own implementation of it.

# Spatio-Net Architectures

An article of Bin Kong et al. titled *Cancer Metastasis Detection via Spatially structured Deep Network* used a Spatio-Net to tackle the metastasis detection problem in Whole Slide Images



The Spatio-Net architecture (the bottom row of Fig. 2) includes two main modules:

CNN and 2D-LSTM. The CNN acts as an effective feature extractor to

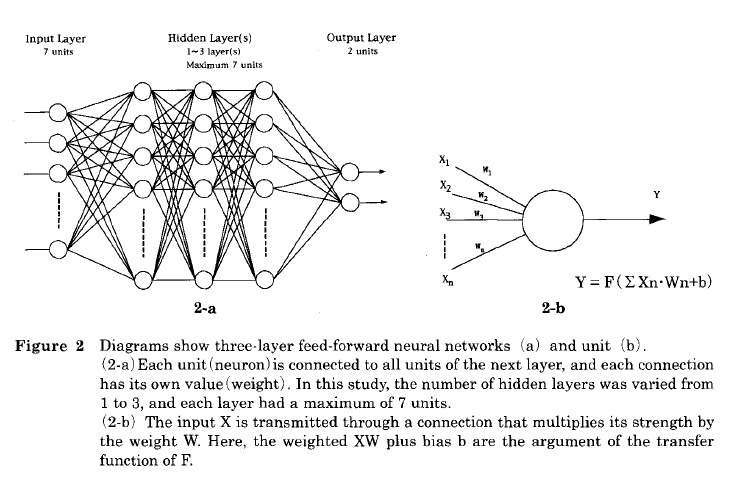
encode each patch and its neighboring patches into compact fixed-length vectors,

resulting in a small grid of vectors. Subsequently, we embed the spatially

structured information in this grid, which is further explored in 2D-LSTM.

# Trial and error

The article *Application of Neural Networks to the Prediction of Lymph Node Metastasis in Oral Cancer* trained different networks until the error (difference between output and correct answer) reached the preset minimum. The final network is the one in the image below.

We could do this with our input images as a simpler network, if the other networks we use are more complicated.