An Hopfield Network for Digit Recognition

Artificial Intelligence Course Project

Gianluca Barbon
818453@stud.unive.it

January 4, 2015

Abstract

This report describes the artificial intelligence course project. This project consists in the implementation of a Hopfield Network using python. The weight matrix will be computed by using different algorithms, in such a way to analyse performances and thus identify the best solution.

1 Introduction

1.1 Neural Networks

They are used for recognition and classification problem, they are capable to learn and so to generalize, that is produce outputs in correspondence of inputs never met before. The training of the net take place presenting a training set (set of example) as input. The answer given by the net for each example will be compared to the desired answer, the difference (or error) between the two will be evaluated and finally the weights will be adjusted by looking at this difference. the process is repeated for the entire training set, until the produced error is minimized, so under a preset threshold.

1.1.1 Classification Problems

Classification problems consists in the classification of an object by looking at its features.

1.2 Hopfield Network [2]

Hopfield networks are neural networks that can be seen as non linear dynamic systems. They are also called recurring networks or feedback networks. We consider neural networks as non linear dynamic systems, where we consider the time variable. In order to do this, we must take into account loops, so we will use recurrent networks. recurrent networks with non linear units are difficult to analyse: they can converge to a stable state, oscillate or follow chaotic trajectories whose behaviour is not predictable. However, the american physicist J.J. Hopfield discovered that with symmetric connections there will exist a stable global energy function. The Hopfield networks have the following properties:

- **sigle layer recurrent networks** in which each neuron is connected to all the others, with the exception of itself (so no cycles are admitted)
- **symmetric:** the synaptic weight matrix is symmetric, so $W = W^T$. This means that the weights is the same in both direction between two neurons.
- **not linear:** in the continuous formulation, each neuron has a non linear invertible activation function

As for the neuron update, we can use three possible approaches:

- **asynchronous update:** where neurons are updated one by one
- **synchronous update:** all neurons are updated at the same moment
- **continuous update:** all neurons are updated in a continuous way

There exists two formulation of the Hopfield model: the discrete one and the continuous one, that differ for the way in which the time flows. For this project we will use the discrete model. In this model the time flows in discrite way and neurons updates in asynchronous way. as for the neuron input, the McCulloch and Pitts model is used, with the adding of an external influence (or bias?) factor:

$$H_i = \underbrace{\sum_{j \neq i} w_{ij} V_j}_{\text{M\&P model}} + \underbrace{I_i}_{\text{external input}}$$

The activation function is the following:

$$V_i = \begin{cases} +1 & \text{se } H_i > 0 \\ -1 & \text{se } H_i < 0 \end{cases} \tag{1}$$

The update of the neurons is a random process and the selection of the unit to be updated can be done in two ways:

1. at each time instant the unit to be updated is chosen randomly (this mode is useful in simulations)
2. each unit is updated independently with constant probability at each time instant

Unlike feedforward networks, a Hopfield network is a dynamic system. It starts from an initial state

$$\vec{V}(0) = (V_1(0), \ldots, V_n(0))^T$$

and evolves through a trajectory until it reach a fixed point in which $V(t+1) = V(t)$ (convergence). The Hopfield theorem supplies a sufficient condition for the convergence of the system. It uses an energy function E that govern the systems :

$$E = -\frac{1}{2}\sum_{i=1}^{n}\sum_{\substack{j=1 \\ j\neq i}}^{n} w_{ij}V_iV_j - \sum_{i=1}^{n} I_iV_i \tag{2}$$

Theorem 1.1 (Hopfield theorem: discrete case)**.** *If the weights matrix in a Hopfield network is symmetric, $diag(W) = 0$, the Energy Function will be a Lyapunov function for the system, so:*

$$\Delta E = E(t+1) - E(t) \leq 0$$

with the equivalence when the system reach a stationary point.

1.3 Hebb's rule

In contrast with computer's byte-addressable memory, that adopt a precise memory address to locate information, the human brain utilize content-addressable memory, that uses the content of data to locate information. The Hebb rule has been introduced to describe such behaviour and states that the weight between two neurons increases if the two neurons activate simultaneously. In detail, the Hebb postulate says that:

> *Let us assume that the persistence or repetition of a reverberatory activity (or "trace") tends to induce lasting cellular changes that add to its stability. [...] When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased.* [1]

The memory is represented like a set of P patterns x^μ, where $\mu = 1, \ldots,$¶: when a new pattern x is presented, the net typically answers producing the pattern in memory that most resembles to x. Accordingly to the Hebb postulate, proportional weights are used in the activation between a pre and a post synaptic neurons:

$$w_{ij} = \frac{1}{N}\sum_{\mu=1}^{P} x_i^\mu x_j^\mu$$

where N is the number of binary units with output $s_1, \ldots, s_N$. The recall mechanism is the following:

$$s_i = \mathrm{sgn}\left(\sum_j w_{ij}s_j\right)$$

Anyway, there are some problems in the use of Hopfield network as content-addressed memories:

- the maximum number of pattern is $0.15N$
- sometimes the net produces spurious states, that is states that do not belong to the memorized patterns
- the recalled pattern is not necessarily the most similar to the input one
- patterns are not recalled with the same emphasis

2 Conclusions

[2]

References

[1] D. O. Hebb. *The Organization of Behavior*. Wiley, New York, 1949.

[2] M. Pelillo. Artificial intelligence course, 2014.

A Appendix: example code

A.1 Class test.java

B Appendix: