THE HTM SPATIAL POOLER AS A HOPFIELD NETWORK

MIRKO KLUKAS

Abstract. These are **unfinished** notes! My aim is to express Numenta's spatial pooler as a (constrained) Hopfield network. In order to do so we have to express the *w-max-overlap* procedure as an appropriate energy function. This enables us to extend the energy by an additional term to decorrelate pairwise column activity similar to P. Fldiak's approach in his 1990's paper.

Relevant keywords are: Hopfield networks, Boltzmann machines, Energy based models, Local anti-Hebbian learning by Földiák [1]. See the references for relevant and more sources.

Date: May 25, 2017.

Contents

1. The Network

1.1. **Network architecture.** Let m and n be the number of visible and hidden units respectively. Furthermore we add a single bias unit, and connect all the units as follows.

a. **Visible-to-hidden** connections encoded by an $(n \times m)$-matrix W.
b. **Hidden-to-hidden** connections encoded by an $(n \times n)$-matrix H. All hidden-to-hidden connections are symmetric without self connections, i.e. we have $h_{ji} = h_{ji}$, and $h_{ii} = 0$ for any i, j. (For the original spatial pooling algorithm these connections are zero.)
c. **Bias-to-hidden** connections encoded by a vector b of size n

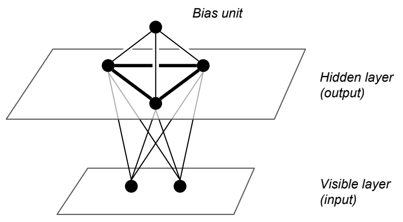

Figure 1. Schematic picture of the network architeture.

1.2. **Weight updates.** After each mini-batch the weights are updated and computed as follows: The visible-to-hidden connections are updated according to a Hebbian-like update rule as described in [2], that is

$$\Delta w_{ij} = y_i \cdot (\varepsilon_+ \, x_j - \varepsilon_- \, \bar{x}_j) \tag{1}$$

After each update the visible-to-hidden connections are clipped to be between 0 and 1, i.e. we set

$$\begin{aligned} w_{ij} &:= 0 \quad \text{if } w_{ij} < 0, \text{ and} \\ w_{ij} &:= 1 \quad \text{if } w_{ij} > 1. \end{aligned}$$

The remaining connections - hidden-to-hidden and bias-to-hidden - are determined by the average activity of the hidden units as follows: Let α_i denote the average activity of an individual unit y_i (estimated by using an exponentially decaying average of the mean probability that a unit is active in each mini batch) then the weights of the bias connections are given by

$$b_i = B_b \cdot \alpha_i. \tag{2}$$

Similar to the *individual* average activity defined above we define the average *pair-wise* activity α_{ij} of two units i and j. With this in hand we set the hidden-to-hidden connections are given by

$$h_{ij} = B_H \cdot (\alpha_{ij} - \alpha_i \; \alpha_j) \quad \text{and} \quad h_{ii} = 0. \tag{3}$$

Opionally we can clip H to be non-negative, i.e.

$$h_{ij} := 0 \text{ if } h_{ij} < 0.$$

Here B_b and B_H are two non-negative factors referred to as **boosting strength**.

2. Energy function

A straight-forward candidate for an energy function resembling the behaviour of the spatial pooler is

$$E(x,y) = -\sum_{i=1}^{n} y_i \cdot \underbrace{\exp\Big(-b_i \cdot y_0 \; - \sum_{j=1}^{n} h_{ij} \cdot y_j\Big)}_{\text{"boosting"}} \cdot \underbrace{\Big(\sum_{j=1}^{m} w_{ij} \cdot x_j\Big)}_{\text{"overlap"}} + S(y) \tag{4}$$

Here $S(y)$ denotes a *size-penalty* (or *size-bound*) forcing the weight of the vectors to be in a predefined desired range. For instance we could use something along the lines of

$$S(y) = \begin{cases} 0 & \text{if } \|y\| \leq w, \text{ and} \\ +\infty & \text{otherwise,} \end{cases}$$

where w is a previously desired *code weight* (that is, the number of active units).

3. Encoding inputs

The energy function E defines a binary encoder by

$$\phi\colon \; x \mapsto \arg\min_y E(x,y).$$

Note that the encoder associated with the above energy function does indeed extend the original *spatial pooling procedure*: if we assume all the hidden-to-hidden connections to be zero, the above energy function does activate the units with the w-topmost *boosted overlap scores*. Note that in practice it is not feasible to compute the actual minimum, and we have to settle with the fact that we might only get to a local minimum.

4. Questions

What are the advantages and disadvantages of the energy "$E = e^A B$"?

...

What's the energy gap here?

See snippets...

What is the right way of finding an energy minimum?

Because the visible- to-hidden connections are clipped to be between 0 and 1 the overlap scores are all positive. Further more the boosting term is also positive. Hence we are dealing with a special dynamic: assume $H \equiv 0$, and a uniform b, then the radial direction in $\{0,1\}^n$ is gradient-like for $E(x,.)$. In consequence if we start with a random y of weight w and do hill descend with neighbors at Hamming distance 1 we are stuck. Note that if we start the hill descend at zero, and move in the direction of steepest descend, we end up with the w-topmost overlap scores. Alternatively we might add neighbours with the same weight at distance 2, that is, allowing to switch an active unit off and an inactive one on.

5. Local Anti-Hebbian learning after P. Földiák

Here is a quick overview of the approach in [1]. **A word of caution:** Please note that we are **NOT** using the notation from Földiák's paper here (in the paper W denotes the hidden-to-hidden connections).

5.1. Architecture and updates.

The architecture is similar to the one described above. The weight updates however are slightly different:

a. **Visible-to-hidden** connections encoded by an $(n \times m)$-matrix W. Updated according to (Hebbian update)

$$\Delta w_{ij} = \varepsilon_W \; y_i \; (x_j - w_{ij}).$$

b. **Hidden-to-hidden** connections encoded by an $(n \times n)$-matrix H. All hidden-to-hidden connections are symmetric without self connections. Updated according to (anti-Hebbian update)

$$\Delta h_{ij} = -\varepsilon_H \; (y_i \; y_j - s^2).$$

After any update the connections are clipped to be below 0.

c. **Bias** or activation threshold encoded by a vector b of size n. Updated according to

$$\Delta b = \varepsilon_B \; (y_i - s).$$

Here $s \in [0,1]$ denotes a previouly fixed desired unit activation probability (I use s indicating a *sparse* activation probability).

5.2. **Encoding inputs.** (For a quick overview to this approach cf. for instance the scholarpedia article by Hopfield himself [3]) To compute the hidden vector with a visible input clamped, we solve the following differential equation

$$\dot{y}_i = \sigma\big(W_i x + H_i y - b_i\big) - y_i.$$

Here we treat y_i as a *continuous* variable. *–It looks like the ODE computes a flowline (integral curve) of the energy gradient.–* Actually, we are looking for a solution of the equation, but are interested in values such that $\dot{y}_i = 0$. Hence we are looking for a fixed point of

$$f(p) = \sigma\big(Wx + Hp - b\big).$$

–Recall that for the distribution associated to a Hopfield net we have $P(y_i = 1 \mid x) = \sigma\big(W_i x + H_i y - b\big)$. So one would could think of y_i as the units activation probability.–

6. Numenta's local inhibition revisited

Let us assume $h_{ij} \geq 0$ for now. Interpret the weights as a neighbourhood relationship. Informally a high weight means "being close" and a high "competitiveness" between the corresponding units. The situation described in [2] would correspond to binary weights, i.e. either $h_{ij} = 0$ or $h_{ij} = 1$

We calculate a units activity y_i as follows: Turn the outgoing weights h_{ij} for $i \neq j$ into an "activation probability" a_{ij} by setting

$$a_{ij} := \frac{h_{ij}}{\sum\limits_{i \neq k} h_{ik}}.$$

Informally this is a best guess for the activity of the other unit, assuming unit i is active.

With this in hand we calculate an estimate A_i for the "activation" in the neighbourhood of unit i, i.e. we set

$$A_i = \sum_{o_j > o_i} a_{ij}.$$

Here o_i denotes the *boosted overlap score* of unit i (that is $o_i = e^{-b} W_i x$). The unit's score o_i acts as a filter threshold to decide whose unit's "activation" it accumulates. Note that we only accumulate the activity of units with a bigger overlap. Then the **generalized local w-max procedure** can be expressed as

$$y_i = \begin{cases} 1 & \text{if } A_i < s, \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

Here s denotes a predefined desired sparsity $s = \frac{w}{n}$ (w is the desired code weight). We can interpret this as follows: if the estimated activation is already bigger than desired, i.e. bigger than the sparsity threshold s, we turn unit i off. If it is smaller than desired we turn unit i on.

Remark. Further note that for binary weights this does indeed resemble Numenta's local w-max procedure, because A_i then is given by

$$\sum_{o_j > o_i} \tfrac{1}{n_i} = \tfrac{w_i}{n_i},$$

where n_i is the number of neighbours, and w_i is the number of neighbours with a bigger overlap. The condition $A_i < s$ would imply that $w_i < w$ which means that o_i is among the w-biggest overlaps.

References

[1] P. Földiák, Forming sparse representations by local anti-Hebbian learning, *Biological Cybernetics* **64** (1990), 165–170.

[2] Yuwei Cui, Subutai Ahmad, and Jeff Hawkins, *The HTM Spatial Pooler: a neocortical algorithm for online sparse distributed coding*, `bioRxiv:085035` (2017).

[3] John J. Hopfield, *Hopfield network*, http://www.scholarpedia.org/article/Hopfield_network.

E-mail address: `mirko.klukas@gmail.com`

(Mirko Klukas) Institute of Science and Technology Austria (IST Austria), Am Campus 1, A 3400 Klosterneuburg, Austria