On the relation between perceptrons and non-negative matrix factorization

Hugo Van hamme
Department ESAT, Katholieke Universiteit Leuven, Leuven, Belgium.
Email: hugo.vanhamme@esat.kuleuven.be

A classical result in the theory of artificial neural networks (ANNs) is Cybenko's theorem [1] which states that a perceptron with at least one hidden layer, sigmoidal output non-linearity and a sufficiently high, but finite, number of nodes can approximate any continuous and bounded function on a bounded domain within a given accuracy. This theorem has made multi-layer perceptrons (MLPs) a popular instrument for classification problems.

Perceptrons are inspired by the information processing in neurons [2], which makes them a trivial choice for modeling cognitive processes. Again inspired by the architecture of mammal brains, one would expect a several layers to be required in perceptron models. Deep MLP networks are however very difficult to train with the back-propagation algorithm. Supervisory information is needed and the fact that it cannot be provided for all hidden layers forms the crux of the problem.

In this paper, the parallels and differences between non-negative matrix factorization (NMF) and a single layer perceptron are discussed. NMF is capable of working without supervision but is equally capable of exploiting supervision information, which makes it very well suited to overcome the training problems in multilayer architectures. However, NMF additionally shows behavior that is observed in the brain: it performs lateral inhibition in a single layer and has masking properties.

NMF finds two factors $\mathbf{W} \in \mathbb{R}^{N\times R}$ and $\mathbf{H} \in \mathbb{R}^{R\times T}$, with positive or zero entries such that a data matrix $\mathbf{V} \in \mathbb{R}^{N\times T}$ with positive or zero entries is approximated by $\mathbf{W}\,\mathbf{H}$. The Frobenius norm and the (generalized) Kullback-Leibler divergence (KLD) are considered here as cost function to express the proximity of $\mathbf{V}$ and $\mathbf{W}\,\mathbf{H}$. The weights $\mathbf{W}$ are learned unsupervisedly by applying NMF to training data $\mathbf{V}$, where different training tokens occupy different columns in $\mathbf{V}$.

In this contribution the multiplicative update algorithms proposed in [3] are cast in the flow diagrams of Figure 1. When classifying a single test token, $\mathbf{V}$ and $\mathbf{H}$ become vectors $\mathbf{v}$ and $\mathbf{h}$, while $\mathbf{W}$ is assumed to be known from the training phase. Finding the NMF solution for $\mathbf{h}$ is compared to node activation in an ANN. The left pane shows the single layer perceptron which maps $\mathbf{v}$ to the hidden nodes $\mathbf{y}$ through the weight matrix $\mathbf{W}$. A sigmoidal output non-linearity $\mathbf{h}(\mathbf{y})$ with diagonal structure is then applied. In NMF with Frobenius norm (middle), $\mathbf{h}(\mathbf{y})$ is replaced by the shaded box. The right pane shows the flow diagram for the KLD metric, which differs only from the middle pane by swapping the input projection $\mathbf{W}^t$ and the element-wise division $\oslash$. Both NMF problems are shown to have lateral inhibition behavior, i.e. a hidden node will deactivate nodes with similar weights (or neighboring neurons in the brain). Physically, inhibition leads to categorical perception. Mathematically, inhibition corresponds to the case where the minimal (Frobenius or KLD) cost is achieved on the constraint boundary.

Supervisory information can be introduced in NMF by augmenting $\mathbf{V}$ with $\mathbf{G}$ such that. $G_{ij} = 1$ if the i-th class is present in token j (and zero otherwise)

$$\begin{bmatrix}\mathbf{G}\\ \mathbf{V}\end{bmatrix} \approx \begin{bmatrix}\mathbf{X}\\ \mathbf{W}\end{bmatrix}\mathbf{H}$$

On test tokens, $\mathbf{H}$ can be computed as before from $\mathbf{V}$ and subsequently the class information $\mathbf{G}$ can be estimated by forming $\mathbf{X}\,\mathbf{H}$. In other words, with supervision, $\mathbf{X}$ acts like an output layer of a perceptron. With the previous result, NMF can be interpreted as an easily learnable perceptron with an output mapping that shows lateral inhibition, which adds to the cognitive motivation to use it as a building block for deep ANN structures.

Finally, the cognitive motivation is strengthened by observing that NMF also shows masking behavior in the sense that the sparsity pattern of $\mathbf{H}$ is robust to perturbations in $\mathbf{V}$ and not all changes to the input data affect $\mathbf{H}$ and hence the classification. A first masking mechanism is through L_1 regularization of the cost function. Secondly, a positive bias term $\mathbf{V}_0$ can be added: find the best $\mathbf{H}$ s.t. $\mathbf{V} \approx \mathbf{W}\,\mathbf{H} + \mathbf{V}_0$. The case where $\mathbf{V}_0 = \mathbf{W}\,\mathbf{H_0}$ for some $\mathbf{H}_0$ could model a reminiscent (decaying) neuron activation. The effect is that the cone of possible $\mathbf{V}$ spanned by the model $\mathbf{W}\,\mathbf{H} + \mathbf{V}_0$ is moved away from the origin, which will typically activate more constraints at the optimal $\mathbf{H}$ and can even lead to solutions that are not affected by the data $\mathbf{V}$ at all (masking).

In summary, there are strong parallels between a single layer perceptron and NMF, the latter showing the additional cognitive properties of unsupervised learnability, lateral inhibition and masking.

REFERENCES

[1] G. Cybenko, "Approximations by superpositions of sigmoidal functions," in Mathematics of Control, Signals, and Systems, vol. 2, no. 4, pp. 303–314, 1989.

[2] F. Rosenblatt, "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain," in *Psychological Review*, vol. 65, pp. 386–408, 1958.

[3] D. Lee, and H. Seung, "Algorithms for non-negative matrix factorization," in *Advances in Neural Information Processing Systems*, vol. 13, pp. 556–562, 2001.

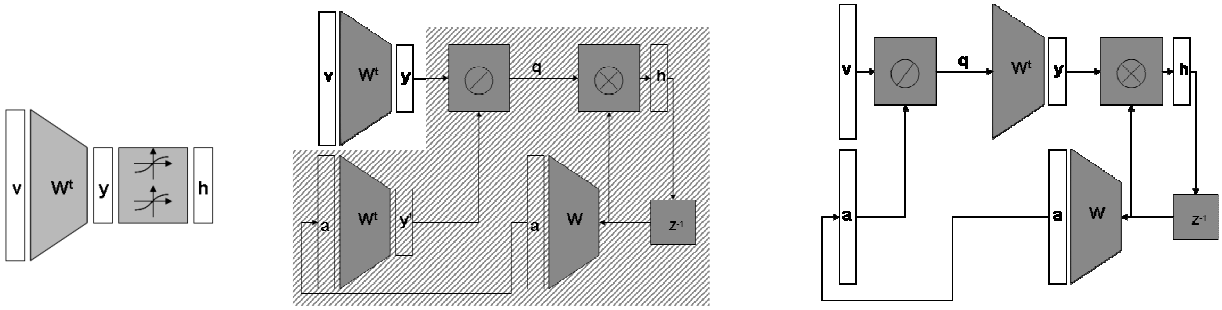

Fig. 1: A flow diagram representation of a single layer perceptron (left), NMF with Frobenius norm (middle) and NMF with KLD (right). $\otimes$ and $\oslash$ are element-wise multiplication and division respectively. "z^{-1}" is a memory over one iteration of the update formulae of [3].