Modelling Letter Perception: The Effect of Supervision and Top-Down Information on Simulated Reaction Times*

Michael Klein

Laboratoire de Psychologie Cognitive, CNRS & Aix-Marseille University
3, place Victor Hugo, Bat. 9, Case D, 13331 Marseille
E-mail: Michael.Q.Klein@gmail.com

Stefan Frank

Centre for Language Studies, Radboud University Nijmegen
Erasmusplein 1, 6525 HT Nijmegen
E-mail: s.frank@let.ru.nl

Sylvain Madec and Jonathan Grainger

Laboratoire de Psychologie Cognitive, CNRS & Aix-Marseille University
3, place Victor Hugo, Bat. 9, Case D, 13331 Marseille
E-mail: {I.Jonathan.Grainger, Sylvain.Madec}@gmail.com

In this study, we model human letter-recognition times using neural networks that extract visual features from real images of the letters. We focus on learning, and on how different learning methods and other factors affect the correlation between simulated reaction times and behavioural data. Specifically, we are interested in studying the effect of 3 factors on this correlation: (i) utilisation of an error signal during learning (supervised vs. unsupervised learning), (ii) whether or not the letter labels exert a top-down influence on the extracted features, and (iii) the effect of letter frequencies. To do so, we used Restricted Boltzmann Machines (RBMs), Back-propagation networks, and RBM/Perceptron hybrid architectures. We find the highest correlations ($r = 0.67$) with supervised models when using top-down information of letter labels on the feature layer during training, but only when the letters' frequencies are taken into account during learning. This study shows that to account for human letter identification times, letter frequency seems to be the most important factor. In addition, top down information of letter labels on the extracted visual features appears to be essential (making the difference between a significant and non-significant correlation). Whether or not the model is supervised makes little difference in the correlation to human reaction time data, but fully unsupervised models have more difficulty generating accurate categorisation for letters

*This work was funded by the ERC grant 230313.

with very low frequencies.

Keywords: Restricted Boltzmann Machines; Back-Propagation; Letter recognition; Reaction Time Modelling

1. Introduction

In order not to be overwhelmed by massive unorganised sensory input, but to perceive the world in a meaningful way, humans categorise objects, events, and actions. Categorisation is beneficial, because unknown individual objects that fall into a known category tend to be of similar significance. To categorise an object a perceiver needs to understand which features of an object are relevant for it to be in a particular category and which features form irrelevant (e.g., random) variability. Recognising letters of the alphabet shares this general problem of similarity in variability. However, being two-dimensional and monochrome, letters present a tractable problem and are, thus, suitable material to study the cognitive processes involved in visual categorisation. Despite being among the more simple of categorisation problems, human letter recognition is far from understood. While there is a sheer intractable amount of experimental studies on letter perception dating back more than a hundred years (see Mueller & Weidemann(2012)[1] for a review), the number of explanatory computational models is still quite limited. While recognising isolated printed letters is not a hard problem for pattern recognition algorithms, there are few computational models that connect and explain human behavioural data. Possibly the best current model of human single letter perception [2] correlates simulated letter-perception times with significant peaks in the EEG signal. This model, however, does not learn and relies on a set of input features that are defined by the modeller.

Since learning and feature extraction are among the main strengths of neural models, it should be feasible to find a neural algorithm for learning letter recognition and feature extraction. However, it is far less simple to find a learning algorithm and architecture that are cognitively plausible and to build a model that can simulate human behavioural data. With letter recognition being a cortical process, and cortical learning being of the unsupervised Hebbian-type[3,4], a good starting point appears to be unsupervised correlation learning. A Restricted Boltzmann Machine (RBM) uses unsupervised correlation learning and is also a good algorithm for feature extraction[5].

In the study reported here, we use the RBM algorithm to extract letter features from images of letters, presented to the model in the form of binary

pixels. This is combined with supervised learning into several architectures in order to vary two factors: (i) the top-down influence of the letter labels on the feature extraction and (ii) the location and time at which supervision is employed.

All these models are used to simulate reaction times by turning the output units into spiking neurons and counting the time steps until the first output unit spikes. The generated reaction times are correlated with the human reaction times reported in Madec et al. (2012)[6].

Furthermore, it has been reported that letter naming times correlate strongly with letter frequencies[7]. While this finding provides very important clues and constraints on the underlying representations and processes involved in letter perception, it is in itself not yet a (causal mechanistic) explanation. It is necessary to develop a cognitive model that can link relative exposure to letters during training to observable reaction times in letter recognition. To establish such a link, in this study all architectures are trained both with a training data set in which the number of items for each letter correspond with known French letter frequencies and with a training data set in which all letters are presented an equal number of times.

2. Method

2.1. *Simulations*

In this study, we tested four main architectures: (i) an RBM / perceptron hybrid, (ii) a pure RBM, (iii) an RBM fine-tuned with back-propagation, and (iv) a pure back-propagation model. The hybrid model (called *hybrid* because it combines two modules that use two different learning algorithms) uses the RBM algorithm to extract a layer of letter feature and then classifies those into letters using the supervised delta-rule (see figure 1A). This means that there is no effect of the letter labels on the features. The second architecture is a pure RBM, in which images and labels are presented on the input layer and a common hidden layer is trained (see figure 1B). For testing, only the image is presented and the hidden activation is computed. From the hidden activation the activation of the letter neurons is generated. In this architecture learning is fully unsupervised, but the labels have an influence on the emerging features on the hidden layer. The third architecture is identical to the second in the first half of the training. Then, the back-propagation algorithm is used to fine-tune the weights (see figure 1C). In the final model, only back-propagation is used. All models

are built, trained and tested with one, two, and three hidden layers. Every architecture is trained with letter frequencies, with logarithmic frequencies, and without any frequencies. This results in a $4 \times 3 \times 3$ experimental design. Fifty simulations (i.e., networks trained from scratch) were performed per cell of the design. We computed the correlations of the reaction times of every simulation with the reaction times of every other simulation within one cell. Henceforth, the average of all correlations within one cell is called the *internal correlation*.

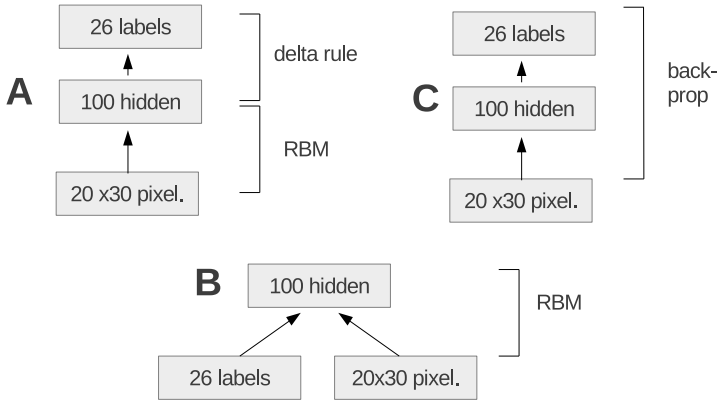

Fig. 1. Three different architectures tested in this study: (A) a hybrid model using unsupervised feature extraction with an RBM and then delta-rule training of the final classification layer; (B) a fully unsupervised RBM network with the labels having a top-down influence on the hidden layer; (C) a standard back-propagation network.

2.2. *Neural Network Algorithms*

2.2.1. *Restricted Boltzmann Machines*

An RBM[5] consists of an input and hidden layer, where every unit in the input layer is connected to every unit in the hidden layer, and each connection has symmetrical weights (i.e., the same value is used for bottom-up recognition and top-down down generation). There are no lateral connections between the units of a layer. In addition, every unit has a bias.

Unit activations are binary and stochastic: The activation of a unit j is

set by first computing the sum of its weighted input, including its bias b_j:

$$z_j = b_j + \sum x_i w_{ij} \tag{1}$$

where x_i is the output of unit i and w_{ij} is the weight between units i and j.

From this sum, the probability that unit j receives an activation of 1 is computed using the logistic function:

$$P(x_j = 1|z_j) = \frac{1}{1 + e^{-z_j}} \tag{2}$$

For training, after random initialisation of weights and biases (here, using Gaussian distribution with mean 0 and standard deviation 0.1), an input vector is applied to the input layer and the hidden activations are computed from it. Then the input is reconstructed from the hidden representations by computing the downward activations. After that, a reconstructed hidden activation is computed from the reconstructed input (see Fig. 2).

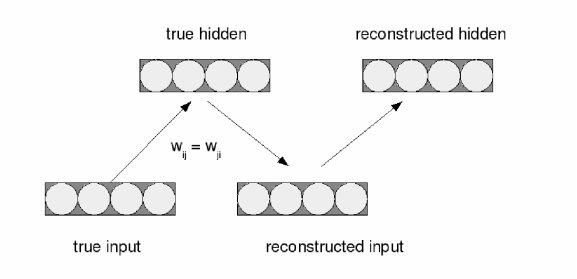

Fig. 2. The up and down algorithm: true hidden representations are generated from the true input. From the true hidden representations the input is reconstructed and from the reconstructed input the hidden representations are reconstructed.

Finally, the network learns by increasing the weights by the product of the input and the hidden units' activation minus the product of reconstructed input and hidden units:

$$\Delta w_{ij} = \epsilon((v_i h_j)_{\text{data}} - (v_i h_j)_{\text{rcon}}) \tag{3}$$

where $(v_i h_j)_{\text{data}}$ denotes the product of the input data v_i and resulting hidden activation h_j, and $(v_i h_j)_{\text{rcon}}$ is the same product but using the reconstructed activations. Both weights and biases change during learning. The weights of the input biases and hidden biases are changed accordingly. The learning rule is analogue since biases can be treated as weights of connections from units that are always 1.

2.2.2. *Training a Deep-Belief Network*

Building up a fully unsupervised deep-belief network[8] is done in several steps. First, an RBM is trained on the input data with the algorithm described above. Next, the weights of this network are fixed. The input is then applied to this network with fixed weights and the output (at the hidden layer) serves as the input to the next RBM that is build on top of this one. This procedure continues until the desired depth of the overall neural model. Note, however, that to train a fully unsupervised deep-belief network, the final RBM does not only have the output of the pre-final RBM as input, but also the labels. To test such a model after training, the weights of the complete model have to be fixed. Then the input (without labels) is presented to the network. Finally the activation of the labels are generated top-down from the top layer of the model.

Subsequently, such a network can be fine-tuned with the back-propagation algorithm (explained below). To do so, however, the top-layer of the deep-belief network will serve as the pre-final layer and the representations of the labels will serve as a top-layer

2.2.3. *Delta-Rule and Back-Propagation*

The final layer of the hybrid model is trained using perceptron learning (delta rule). To fine-tune the weights of our deep-belief networks, we use the back-propagation algorithm.[9] A back-propagation network can be regarded as a multi-layer perceptron, or the perceptron as a special case (single-layer) of a back-propagation network. Therefore, we will treat both algorithms in this section.

The activation of a unit in both perceptron and back-propagation network is computed in the same manner as in the RBM (see equation 1). Also, the output y_j of a unit j is computed with the logistic function analog to the computation of the probability of a unit being 1 in the RBM. However, since the output is not binary, no sampling is necessary:

$$y_j = \frac{1}{1 + e^{-z_j}} \tag{4}$$

The computation of the error and the weight change for a perceptron is the same as for the final layer of a back-propagation network. In this kind of supervised learning, the output of the network y_k is subtracted from the desired output y_k^* to compute the error e_k. The weight change Δw_{ik} is then calculated by multiplying e_k by the activation x_i of the input neuron and the rate of change α:

$$e_k = y_k^* - y_k \tag{5}$$

$$\Delta w_{ik} = \alpha e_k x_i \tag{6}$$

For the back-propagation network the training of the weights to the pre-final layers is somewhat more complicated. While the rule for the update of the weights can stay the same as in the perceptron and the final layer, the computation of the weight change (Δw_{ik}) is quite different. We can only directly measure errors at the output layer. To get an approximation of the error at the hidden layer, we distribute the error of an output unit j to all the hidden units. The error is distributed proportional to the weight of the connection from the hidden unit to the output unit j. This assumes that the contribution of this hidden unit to the error is proportional to its connection strength to that output unit. The same algorithm applies to all the hidden layers of the network. Equation 7 shows the computation of the error at the hidden layer.

$$e_j = y_j(1 - y_j) \sum_k e_k w_{jk} \tag{7}$$

The error e_j at unit j is the sum of errors e_k of units k to which this hidden unit is projecting, times the weight w_{jk} of the connections multiplied with the output of the unit y_j times $(1 - y_j)$.

2.2.4. *Simulating Reaction Times*

To simulate reaction times after training the networks, we converted the neurons representing the letters into spiking leaky integrator neurons [10,11]. Every neural unit i was represented as a membrane potential u_i dynamically changing over time using the following equation:

$$u_i(t+1) = u_i(t) + c \sum_k (w_{jk} y_k) - l; \tag{8}$$

The resting (and initial) value of the membrane potential was set to -70 (mV) for all units. Leakage l was set to 0.05 and a constant c used to convert input to the unit into voltage was set to 0.2. Reaction times were computed simply by counting the time steps until u_i reached the threshold of $-55mV$.

2.3. *Human Reaction Time Data*

The human reaction time data were taken from the study of Madec et al. (2012)[6]. In this study, a behavioural index of letter identification processes was obtained by combining an immediate naming and a conditional delayed naming task (see Figure 3). In the immediate naming task, participants simply named as quickly as possible a target letter that was displayed on a computer screen. Then, on each trial, after naming the letter, participants performed a conditional delayed naming task. After a variable delay following their naming response, either a green or a red circle (light grey and dark grey in Fig. 3) was presented, and participants had to repeat the target letter's name they had just produced, only when they saw a green circle.

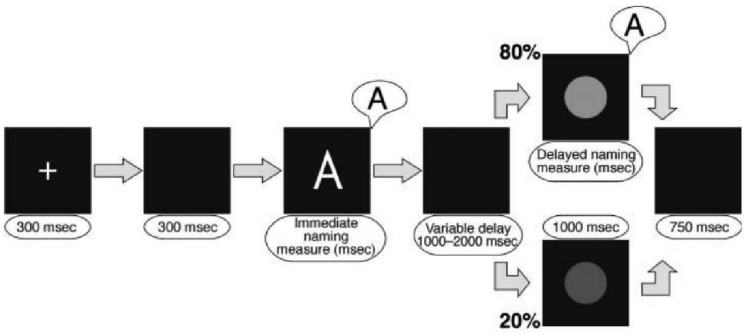

Fig. 3. Schematic depiction of the task in the Madec et al. (2012) study.

The immediate naming measure is assumed to include two main sources of variance that are related to the two main processes involved in letter naming. The first source of variance comes from visual identification processes, and the second source is related to output articulatory processes.

Having the measures of immediate naming and delayed naming, it is possible to compute a simple linear regression with immediate naming times being the dependent variable and delayed naming time being the independent variable (i.e., naming times are explained by delayed naming times). The residual values of the regression (i.e., the remaining unexplained variance), is likely to correspond to the time required by the visual identification processes.

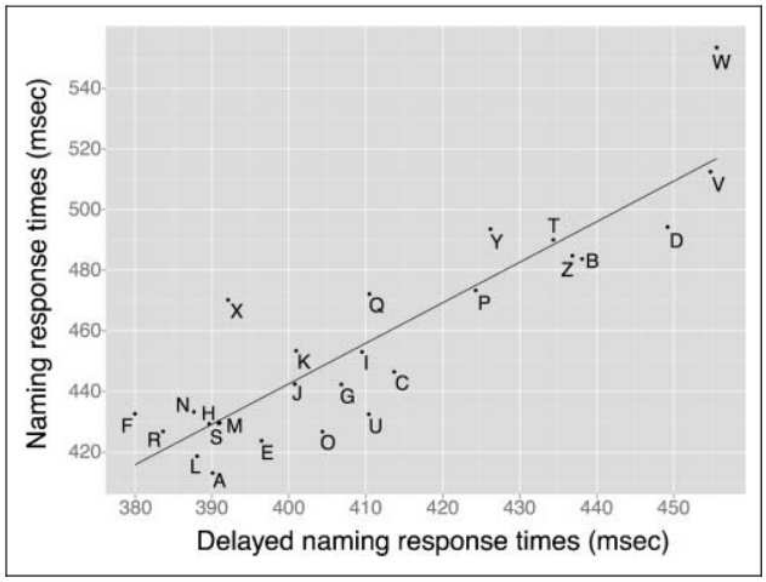

Fig. 4. Immediate naming response times (msec) as a function of delayed naming response times (msec), reproduced from Madec et al. (2012).

3. Results

In all the following simulations, having more than one hidden layer yielded worse correlations with the reaction time data, rather than improving them. Hence, we will not give further details of the simulations with more than one hidden layer, but focus on the results of the architectures with one hidden layer.

Table 1 summarises the results: For each model, it shows the correlation between the simulated response times (averaged over the 50 runs) and human data (averaged over subjects), as well as the internal correlation (the correlation between different runs within the same architecture). Given the degrees of freedom, the correlation with response times is significant ($p < 0.05$) if $r > 0.38$, so three of the models simulate response times that significantly (and positively) correlate with the human response data. Importantly, these models also show very strong internal correlation, indicating that the results are reliable and do not strongly depend on the random initial connection weights or random sampling used in the RBM.

Significant correlations were only obtained when using a realistic letter-frequency distribution during training. This reflects the importance of let-

Table 1. Correlation of Simulated Reaction Times with Behavioural Data and Models' Internal Correlation

architecture	freq	correlation	int. correl.
hybrid	no	−0.22	0.09
	yes	0.21	0.09
RBM	no	0.13	0.23
	yes	0.61	0.8
	log	0.53	0.9
RBM + BP	no	−0.27	0.22
	yes	0.67	0.99
BP	no	−0.17	0.57
	yes	0.67	0.97

ter frequency as a predictor of human response times. However, in the pure RBM model, the very low frequency items (such as K, W, X, Y, and Z) did not result in any above-threshold activation. That is, they were not properly learned. We attempted to solve this problem by using logarithmically transformed letter frequencies (increasing the relative frequency of the infrequent letters). Although this was successful, it also reduced the correlation with human data to $r = 0.53$.

Another solution to the problem of low-frequency letter recognition turned out to be the fine-tuning of the RBM weights using back-propagation training. In addition, this increased the correlation with human response times to $r = 0.67$. For this model, the relation between simulated and actual response times (for each letter) is plotted in Figure 5.

When the model was trained with back-propagation from the beginning, the correlation was the same as in case of mixed RBM and back-propagation training ($r = 0.67$). Also, this model did respond to low-frequency letters. Hence, there was no benefit of applying the RBM prior to back-propagation.

The correlation between these latter models and the behavioural data ($r = 0.67$) were substantially higher than the correlation between (French) letter frequencies[7] and the behavioural data ($r = 0.49$), indicating that the model explains more than just the effect of letter frequencies.

4. Conclusions

In the study reported in this paper, we investigated the possibilities of modelling human letter perception with Restricted Boltzmann Machines. We found that fully unsupervised RBMs have problems learning low-frequency letters, but when trained with the logarithmic frequencies, they perform adequately and can be considered a valuable model of human letter per-

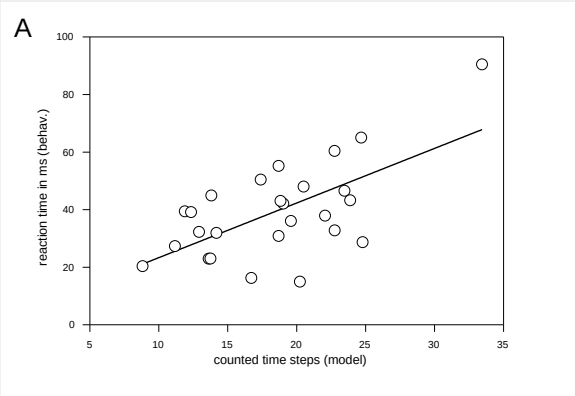

Fig. 5. Relation between simulated and behavioural data for RBM with back-prop fine-tuning.

ception. However, RBMs fine-tuned with back-propagation are superior to pure RBMs and, since pure back-propagation networks perform on the same level, it seems that nothing is gained by prior training with the RBM algorithm. Altogether, it appears that RBMs, while faster to train, are not superior to back-propagation networks when it comes to the modeling of human reaction time data. Furthermore, it seems that top down information of letter labels on the extracted visual features appear to be essential, since only the hybrid failed at simulating reaction times that showed significant correlations with human data. Finally, this study could confirm the essential role of letter frequency as a strong factor of letter recognition times and successfully distinguished cognitive architectures that link letter frequencies with reaction times from those that do not.

References

1. S. T. Mueller and C. T. Weidemann, *Acta Psychologica* **139**, 19 (2012).
2. A. Rey, S. Dufau, S. Massol and J. Grainger, *Cognitive Neuropsychology* **26**, 7 (2009).
3. R. Malinow, *Science* **252**, 722 (1991).
4. H. Markram, J. Lüubke, M. Frotscher and B. Sakmann, *Science* **275**, 213 (1997).
5. P. Smolensky, Information processing in dynamical systems: Foundations of harmony theory, in *Parallel Distributed Processing: Volume 1: Foundations*, eds. D. E. Rumelhart and J. L. McClelland (MIT Press, Cambridge, MA, 1986) pp. 194–281.
6. S. Madec, A. Rey, S. Dufau, M. Klein and J. Grainger, *J Cogn Neurosci* **24**,

1645 (2012).
7. B. New and J. Grainger, *Acta Psychologica* **138**, 322 (2011).
8. G. E. Hinton, *Trends in Cognitive Sciences* **11**, 428 (2007).
9. D. E. Rumelhart, G. E. Hinton and R. J. Williams, *Nature* **323**, 533 (1986).
10. L. Lapicque, *J Physiol Pathol Gen* **9**, 620 (1997).
11. A. N. Burkitt, *Biological Cybernetics* **95**, 1 (2006).