Self-organization on a Sphere with Application to Topological Ordering of Chinese Characters

Andrew P. Papliński(✉)

Monash University, Melbourne, Australia
andrew.paplinski@monash.edu

Abstract. We consider a case of self-organization in which a relatively small number N of data points is mapped on a larger number M of nodes. This is a reverse situation to a typical clustering problem when a node represents a center of the cluster of data points. In our case the objective is to have a Gaussian-like distribution of weights over nodes in the neighbourhood of the winner for a given stimulus. The fact that $M > N$ creates some problem with using learning schemes related to Gaussian Mixture Models. We also show how the objects, Chinese characters in our case, can be topologically ordered on a surface of a 3D sphere. A Chinese character is represented by an angular integral of the Radon Transform (aniRT) which is an RTS-invariant 1-D signature function of an image.

Keywords: Self-organization on a sphere · Probabilistic self-organizing maps · Gaussian Mixture Models · Angular integral of the Radon Transform

1 Introduction

In our ongoing work on multimodal integration of visual and auditory stimuli e.g. [4,15–17] we consider a network of self-organizing modules. Each module performs a self-organizing mapping based, in principle, on Kohonen's algorithm [8,19]. The main difference between our case and a typical clustering algorithm stems from the fact that we have the number of stimuli (data points) N smaller than the number of neurons (nodes) M. In order to maintain the redundancy of the stimuli representation we keep the ratio $M/N \approx 16 \ldots 20$, the number being inferred from the work [12]. In the case on an on-line training [13] additional nodes are generated to maintain the ratio constant. We try to approximate the postsynaptic responses of our module by a Gaussian-like shape over the neurons in the neighbourhood of the winner so that the variants of the stimuli are mapped into the same neighbourhood. The other feature of our model is that the nodes are randomly positioned on a surface of a sphere. In the previous works we used only the northern hemisphere, here we extent the latent space to be the full unity 3-D sphere. The stimuli and the weights are also normalised to be the D-dimensional unity vectors.

A. Hirose et al. (Eds.): ICONIP 2016, Part IV, LNCS 9950, pp. 452–459, 2016.
DOI: 10.1007/978-3-319-46681-1_54

The objectives of this paper are as follows. Firstly, we would like to consider a possibility of applying one of many available versions of probabilistic SOMs stemming from the fundamental Gaussian Mixture Models (GMM). From the significant body of literature on these topics we point to the following recent works [3,7,10,11] and the classics [1]. Secondly, we would like to demonstrate the spherical latent space. As an example of objects to be mapped, we have selected Chinese characters rendered with a specific font and selected from the Unicode CJK table [18].

2 The Stimuli

The Chinese characters from the Unicode table with hex codes from `4E00` to `9FA5` have been rendered using the Microsoft JhengHei UI font of size 32 points. Such rendering produces black-and-white binary images of size 43×43 pixels. For each character image we calculate a signature function, a vector, as an angular integral of the Radon transform (aniRT). Details can be found in [14]. The size of the aniRT vector is equal to the diagonal of the image, 61 in this case. For further processing we select $D = 22$ central components sufficient for a detailed representation. In Fig. 1 we illustrate aniRT vectors for randomly selected 8 Chinese characters. In general, we have pre-calculated aniRT vectors for all Chinese characters from the Unicode table. After that, the aniRT vectors are projected up on the $(D+1)$-dimensional hypersphere. A short note on the selected method of projection is given in the appendix.

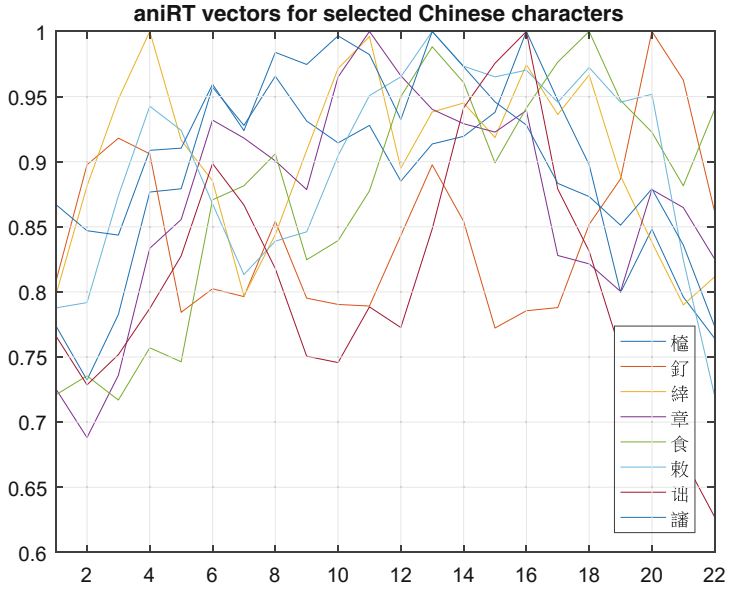

Fig. 1. The aniRT vectors for selected Chinese characters rendered with the Microsoft JhengHei UI font of size 32 points

3 The Self-organizing Module

The self-organizing module that form the networks modelling multimodal integration [15–17] interprets D-dimensional stimuli $\mathbf{x}_n$ in a latent space which is a unity 3-D sphere. There are M neurons/nodes randomly distributed on the surface of the sphere, each node is characterised by the 3-D position vector $\mathbf{v}_m$ and a D-dimensional weight vector $\mathbf{w}_m$. Random, rather than on a regular lattice, distribution of nodes seems to be more biologically plausible. The stimuli are organized into a $D \times N$ matrix, X, one stimulus $\mathbf{x}_n$ per column. The weight vectors, $\mathbf{w}_m$ are organized in a $M \times D$ weight matrix, W, one weight vector per row. Similarly, the position vectors, $\mathbf{v}_m$ are organized in a $M \times 3$ position matrix, V, one location vector per row. All stimuli, weight vectors and position vectors are located on the respective unity hyper-spheres, so that $||\mathbf{x}_n|| = 1,\ ||\mathbf{w}_m|| = 1,\ ||\mathbf{v}_m|| = 1$. Unlike in many clustering-related applications where number of data points $\mathbf{x}_n$, is greater than the number of nodes, that is $N > M$, in our perceptual modelling case, we maintain a stochastically constant ratio of nodes per stimuli, $\gamma = \frac{M}{N} \approx 16$ to 20 to allow for robustness of the stimuli representation. The specific value has been inferred from the columnar organization of the brain [12].

The fact that the number of stimuli is smaller that the number of weights reverses the classical clustering problem. In our case, for a given stimulus $\mathbf{x}_n$ there should be a central node $\mathbf{v}_m$ with the weight vector at this node $\mathbf{w}_m$ being approximately equal to the stimulus, and the neighbouring nodes having weights "Gaussianly" close to the stimulus.

At this point it would be natural to expect that any Gaussian Mixture Model [1] should solve the problem of placing required weights over the nodes allocated to a specific stimuli. We have started with a model inspired by the Elastic Nets [2,5]. Skipping standard details we ended up with the following log-likelihood to minimize:

$$E(W) = -\sum_{m=1}^{M} \log \sum_{n=1}^{N} \mathcal{N}(\mathbf{x}_n, \mathbf{w}_m, \Sigma_n) + \sum_{m=1}^{M} \frac{1}{2d_m} \sum_{j \in \Lambda_m} \nu_{mj} ||\mathbf{w}_m - \mathbf{w}_j||^2_{C_m} \qquad (1)$$

The first term describes a sum of N Gaussians $\mathcal{N}(\mathbf{x}_n, \mathbf{w}_m, \Sigma_n)$, whereas the second term promotes clustering of weights around the winning nodes. The nodes $\mathbf{v}_j$ with weight vector $\mathbf{w}_j$ are located in the neighbourhood Λ_m of the n-th node, see below. From Eq. (1) it is easy to derive the following iterative expression for the next value of the weight matrix, W.

$$W = (I + \beta U)^{-1}(R \cdot X^T + \beta \hat{W}_\Lambda) \qquad (2)$$

where R is a $M \times N$ matrix of responsibilities which are formed from ratios of Gaussians, X is a $D \times N$ matrix of stimuli/data points, $\hat{W}_\Lambda$ is a matrix of weighted means of weights in the neighbourhoods Λ, and U is a diagonal matrix formed from the distances between the position vectors in the neighbourhood $\mathbf{v}_j$. Equation (2) converges very quickly to a matrix W, however, instead of spreading winning nodes according to the stimuli $\mathbf{x}_n$ it does the opposite, namely clustering

data points to a small number of weights. This behaviour stems from the fact that there are not enough data points for a given number of weights and nodes.

We have adopted the solution originating from the stochastic approximation considerations, which results in solutions close to the original Kohonen learning law, with the additional benefit that the algorithm is obtained by minimization of an energy function. We follow the original paper [6] developed later by [9]. For each stimulus $\mathbf{x}_n$ we calculate the post-synaptic activations, $\mathbf{d}_n = W \cdot \mathbf{x}_n$ and find the node $\mathbf{v}_m^n$ for which the activation attains the maximum. The neighbourhood Λ_n of the winning node is defined by a Gaussian function with the variance $\sigma^2_{\Lambda_n}$ centered at the node $\mathbf{v}_m^n$. The neighbouring nodes are located inside the radius of $r_{\Lambda_n} = 2\sigma_{\Lambda_n}$. Note that the nodes are randomly distributed on a surface of a 3-D sphere. The area of the sphere surface is initially allocated equally to all N stimuli, hence we have $4\pi 1^2/N = \pi r_\Lambda^2$, and $r_\Lambda = 1/\sqrt{N}$, or $\sigma_\Lambda^2 = 1/(4N)$. As a result, we include in the neighbourhood of the winning node $\mathbf{v}_m^n$, all nodes located at $\mathbf{v}_{j_n}$ for which the inner product satisfy the following condition:

$$j_n \in \Lambda_n \text{ if } \mathbf{v}_m^n \mathbf{v}_{j_n}^T > \cos(r_{\Lambda_n}); \quad M_n = |\Lambda_n| \tag{3}$$

where M_n is the number of nodes in the neighbourhood Λ_n. Now, in a way similar to [9], we defined the energy function as:

$$E(W) = \sum_{n=1}^{N} E_n; \quad E_n = \frac{1}{2} \sum_{j_n \in \Lambda_n} ||\mathbf{w}_{j_n} - \mathbf{x}_n^T||^2 \exp\left(-\frac{||\mathbf{v}_m^n - \mathbf{v}_{j_n}||^2}{2\sigma^2_{\Lambda_n}}\right) \tag{4}$$

To minimize the energy, we calculate the derivative with respect to each weight vector:

$$\frac{\partial E_n}{\partial \mathbf{w}_{j_n}} = (\mathbf{w}_{j_n} - \mathbf{x}_n^T) \exp\left(\frac{\mathbf{v}_m^n \mathbf{v}_{j_n}^T - 1}{\sigma^2_{\Lambda_n}}\right) \tag{5}$$

The above expression results in a Kohonen-like learning law that, for the all weights W_{Λ_n} in the neighbourhood Λ_n specified in Eq. (3), and taking into account the fact that weights need to be kept on the D-dimensional hypersphere, can be written in a way equivalent to the "dot-product" law of the following form

$$W_{\Lambda_n}(t+1) = W_{\Lambda_n}(t) + \eta(t)\Lambda_n(t)(\mathbf{x}_n^T - \mathbf{d}_{\Lambda_n} \cdot W_{\Lambda_n}(t)), \quad \mathbf{d}_n = W_{\Lambda_n} \cdot \mathbf{x}_n \tag{6}$$

where the neighbourhood function is

$$\Lambda_n(t) = \exp\left(\frac{V_{\Lambda_n} \cdot \mathbf{v}_{j_n}^T - 1}{\sigma^2_{\Lambda_n}(t)}\right) \tag{7}$$

Note that the sizes of the matrices W_{Λ_n} and V_{Λ_n} are $M_n \times D$ and $M_n \times 3$, respectively. Operations between vectors and matrices in Eq. (6) are performed on the row-by-row basis. The learning gain η is reduced according to another Gaussian curve $\eta(t) = \exp(-t^2/(2\sigma_\eta))$, where σ_η is selected so that $\eta(T/2) = 0.5$, T being the total number of epochs. This choice of σ_η ensures a good proportion between the ordering and the convergence phases. The $\sigma^2_{\Lambda_n}(t)$ which describes the narrowing of the neighbourhood is reduced linearly from the initial value $\sigma^2_{\Lambda_n}(1)$ to its final value $\sigma^2_{\Lambda_n}(T)$.

4 Self-organization

In general, the learning law of Eq. (6) can be applied in an on-line/incremental fashion similar to that described in [13]. In this section, we concentrate on demonstrating learning on the sphere, hence, we select randomly $N = 80$ Chinese characters as described in Sect. 2. The results of learning on the sphere are presented in Fig. 2. For $N = 80$ the number of nodes is $M = 20 \times N = 1600$. Since the dimensionality of stimuli is $D = 23$, the sizes of the matrices X, V and W are 23×80, 1600×3 and 1600×23, respectively. In Fig. 2 the neighbourhood for each stimulus $\mathbf{x}_n$ is marked by coloured dots showing the positions of the V_{Λ_n}. The lines indicate the Voronoi tessellation with respect to the positions of the winning nodes. Subject to projection distortion each neighbourhood contains approximately 20 nodes, and the tessellation cells are approximately equal in area. The 3-D view generated by MATLAB uses the orthographic projection, therefore the cells close to the edges appear to be more densely packed. The sphere is semi-transparent, therefore, the characters on the hidden surface are

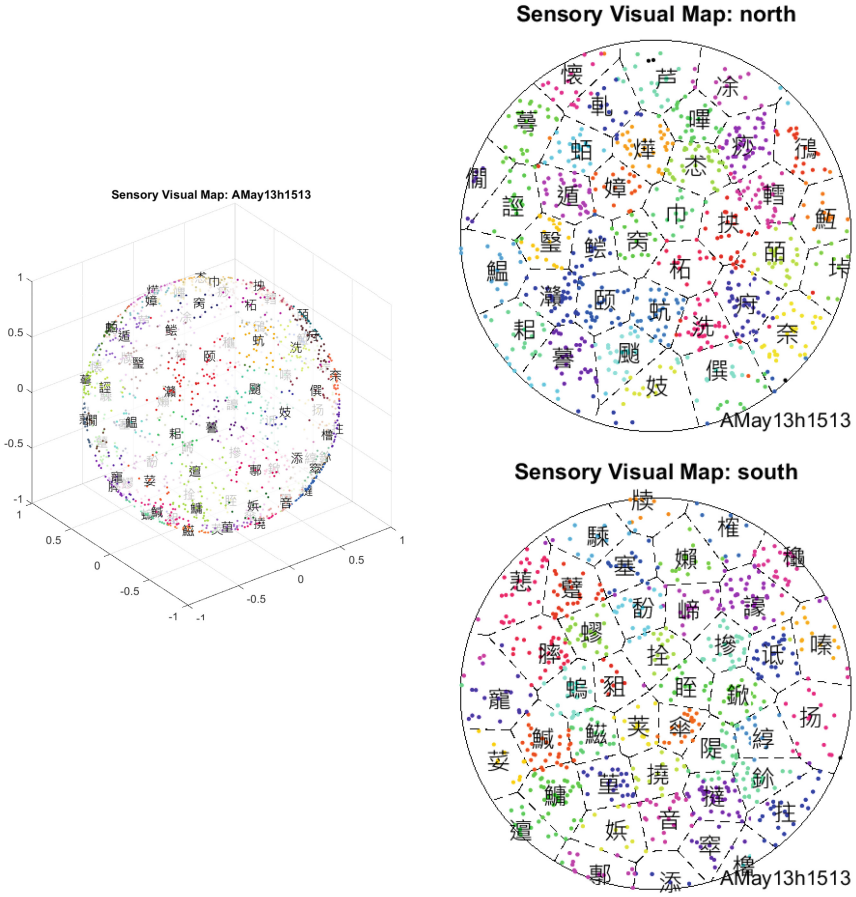

Fig. 2. Learning on a sphere: 3-D view and projection of two hemispheres

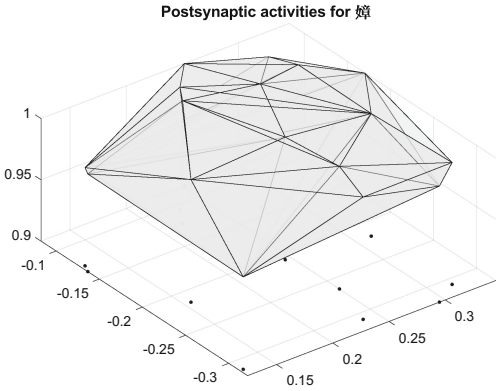

Fig. 3. A map of postsynaptic activities in the neighbourhood of a single stimulus

visible. Our projection method maintains the same proportion along the diameter and the semi-circle, as a consequence looks less distorted.

In Fig. 3 we present a surface of postsynaptic activities for one selected stimulus $\mathbf{x}_n$, namely the character 嫜 (hex '5ADC') that can be found in the upper-central part of the northern hemisphere. The neighbourhood Λ_n consists of 20 nodes V_{Λ_n} with related weights W_{Λ_n}. The components of the vector of postsynaptic activities $\mathbf{d}_{\Lambda_n} = W_{\Lambda_n} \cdot \mathbf{x}_n$ are plotted against the position of the nodes. The positions of the nodes are projected down on a plane $z = 0.9$ and marked with dots in Fig. 3. We use the Delaunay triangulation to create the surface.

As a final comment, please note that the topological ordering of Chinese characters is not done in the order a Chinese speaker might expect, e.g., with respect to radicals. The ordering is strictly based on the aniRT vectors representing the characters as described in Sect. 2.

Conclusion

We have considered problems related to self-organization on a surface of a 3-D sphere in a situation when a number of stimuli is significantly lower than the number of nodes. In this case algorithms originated from the Gaussian Mixture Models might not produce the desired allocation of stimuli to a clusters of nodes. We used an energy function originating from probabilistic considerations which results in a learning law equivalent to the Kohonen-like dot-product law that delivers satisfactory postsynaptic behaviour.

Appendix

Projections between a sphere and a plane related to the problem of creating Earth maps have a long and varied history. The list of available projections is rather long and the Wikipedia[1] is a good point to start.

[1] https://en.wikipedia.org/wiki/List_of_map_projections.

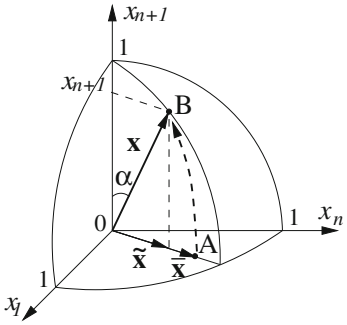

Fig. 4. Projection of a point A from the inside of the n-dimensional hypersphere (shown as a unity circle in 2-D space), onto the $(n+1)$-dimensional unity hypersphere (shown as a point B on the 3-D sphere)

Refer to Fig. 4 and consider an n-dimensional vector $\bar{\mathbf{x}}$ (point A) that needs to be projected onto the $(n+1)$-dimensional unity hypersphere to obtain a unity vector $\mathbf{x}$ (point B). In order to minimise distortions, we select the position of the point B on the grand circle proportional to the position of the point A on the radius. Hence:

$$\alpha = \frac{\pi}{2}\|\bar{\mathbf{x}}\| \tag{8}$$

From this it is easy to obtain the vector $\mathbf{x}$ as:

$$\mathbf{x} = [\tilde{\mathbf{x}}, \cos\alpha] = [k\bar{\mathbf{x}}, \cos\alpha]\,, \quad k = \frac{\sin\alpha}{\|\bar{\mathbf{x}}\|} = \frac{\pi \sin\alpha}{2\alpha} \tag{9}$$

Projecting down the $(n+1)$-dimensional vector $\mathbf{x}$ of Eq. (9) from the surface of the $(n+1)$-dimensional sphere we obtain the related n-dimensional vector $\bar{\mathbf{x}}$ located inside the n-dimensional sphere in the following way:

$$\bar{\mathbf{x}} = \frac{1}{k} \cdot \tilde{\mathbf{x}} = \frac{1}{k} \cdot \mathbf{x}_{1:n} \tag{10}$$

References

1. Bishop, C.M., Svensen, M., Williams, C.K.I.: GTM: the generative topographic mapping. Neural Comput. **10**(1), 215–234 (1998)
2. Carreira-Perpiñán, M.A., Goodhill, G.J.: Generalised elastic nets, pp. 1–52 (2003). arXiv.org: http://arxiv.org/abs/1108.2840
3. Cheng, S.S., Fu, H.C., Wang, H.M.: Model-based clustering by probabilistic self-organizing maps. IEEE Trans. Neural Netw. **20**(5), 805–826 (2009)
4. Chou, S., Papliński, A.P., Gustafsson, L.: Speaker-dependent bimodal integration of Chinese phonemes and letters using multimodal self-organizing networks. In: Proceedings of International Joint Conference on Neural Networks, Orlando, Florida, pp. 248–253, August 2007

5. Cohen, D., Papliński, A.P.: A comparative evaluation of the generative topographic mapping and the elastic net for the formation of ocular dominance stripes. In: Proceedings of WCCI-IJCNN, pp. 3237–3244. IEEE (2012)
6. Heskes, T.: Energy functions for self-organizing maps. In: Kohonen Maps, pp. 303–315. Elsevier (1999)
7. Heskes, T.: Self-organizing maps, vector quantization, and mixture modeling. IEEE Trans. Neural Netw. **12**(6), 1299–1305 (2001)
8. Kohonen, T.: Self-organising Maps, 3rd edn. Springer, Heidelberg (2001)
9. Lau, K., Yin, H., Hubbard, S.: Kernel self-organising maps for classification. Neurocomputing **69**(6), 2033–2040 (2006)
10. Lebbah, M., Jaziri, R., Bennani, Y., Chenot, J.H.: Probabilistic self-organizing map for clusteringand visualizing non-i.i.d data. Int. J. Comput. Intell. Appl. **14**(2), 1–29 (2015)
11. Lopez-Rubio, E.: Probabilistic self-organizing maps for continuous data. IEEE Trans. Neural Netw. **21**(10), 1543–1554 (2010)
12. Mountcastle, V.B.: The columnar organization of the neocortex. Brain **120**, 701–722 (1997)
13. Papliński, A.P.: Incremental Self-Organizing Map (iSOM) in categorization of visual objects. In: Huang, T., Zeng, Z., Li, C., Leung, C.S. (eds.) ICONIP 2012. LNCS, vol. 7664, pp. 125–132. Springer, Heidelberg (2012). doi:10.1007/978-3-642-34481-7_16
14. Papliński, A.P.: The Angular Integral of the Radon Transform (aniRT) as a feature vector in categorization of visual objects. In: Guo, C., Hou, Z.-G., Zeng, Z. (eds.) ISNN 2013. LNCS, vol. 7951, pp. 523–531. Springer, Heidelberg (2013). doi:10.1007/978-3-642-39065-4_63
15. Papliński, A.P., Gustafsson, L., Mount, W.M.: A model of binding concepts to spoken names. Aust. J. Intell. Inf. Process. Syst. **11**(2), 1–5 (2010)
16. Papliński, A.P., Gustafsson, L., Mount, W.M.: A recurrent multimodal network for binding written words and sensory-based semantics into concepts. In: Lu, B.-L., Zhang, L., Kwok, J. (eds.) ICONIP 2011. LNCS, vol. 7062, pp. 413–422. Springer, Heidelberg (2011). doi:10.1007/978-3-642-24955-6_50
17. Papliński, A.P., Mount, W.M.: Bimodal Incremental Self-organizing Network (BiSON) with application to learning Chinese characters. In: Lee, M., Hirose, A., Hou, Z.-G., Kil, R.M. (eds.) ICONIP 2013. LNCS, vol. 8226, pp. 121–128. Springer, Heidelberg (2013). doi:10.1007/978-3-642-42054-2_16
18. Unicode: CJK unified ideographs (2015). http://unicode.org/charts/PDF/U4E00.pdf
19. Yin, H.: The self-organizing maps: background, theories, extensions and applications. Comput. Intell. Compend. **762**, 715–762 (2008)