A Novel Continual Learning Approach for Competitive Neural Networks[⋆]

Esteban J. Palomo[1,2][0000−0002−8547−9393], Juan Miguel Ortiz-de-Lazcano-Lobato[1,2][0000−0001−6448−0187], José David Fernández-Rodríguez[1,2][0000−0003−3702−2230], Ezequiel López-Rubio[1,2][0000−0001−8231−5687], and Rosa María Maza-Quiroga[1,2][0000−0002−4693−0954]

[1] University of Málaga, Bulevar Louis Pasteur, 35, 29071, Málaga, Spain
{ejpalomo,jmortiz,josedavid,ezeqlr,rosammq}@lcc.uma.es

[2] Biomedic Research Institute of Málaga (IBIMA), C/ Doctor Miguel Díaz Recio, 28, 29010, Málaga, Spain

Abstract. Continual learning tries to address the stability-plasticity dilemma to avoid catastrophic forgetting when dealing with non-stationary distributions. Prior works focused on supervised or reinforcement learning, but few works have considered continual learning for unsupervised learning methods. In this paper, a novel approach to provide continual learning for competitive neural networks is proposed. To this end, we have proposed a different learning rate function that can cope with non-stationary distributions by adapting the model to learn continuously. Experimental results performed with different synthetic images that change over time confirm the performance of our proposal.

Keywords: Continual learning · Unsupervised learning · Competitive neural networks.

[⋆] This work is partially supported by the Ministry of Science, Innovation and Universities of Spain under grant RTI2018-094645-B-I00, project name Automated detection with low-cost hardware of unusual activities in video sequences. It is also partially supported by the Autonomous Government of Andalusia (Spain) under project UMA18-FEDERJA-084, project name Detection of anomalous behavior agents by deep learning in low-cost video surveillance intelligent systems. It is also partially supported by the Autonomous Government of Andalusia (Spain) under project UMA20-FEDERJA-108, project name Detection, characterization and prognosis value of the non-obstructive coronary disease with deep learning. All of them include funds from the European Regional Development Fund (ERDF). It is also partially supported by the University of Malaga (Spain) under grants B1-2019_01, project name Anomaly detection on roads by moving cameras, and B1-2019_02, project name Self-Organizing Neural Systems for Non-Stationary Environments. The authors thankfully acknowledge the computer resources, technical expertise and assistance provided by the SCBI (Supercomputing and Bioinformatics) center of the University of Málaga. They also gratefully acknowledge the support of NVIDIA Corporation with the donation of two Titan X GPUs. The authors also thankfully acknowledge the grant of the Universidad de Málaga and the Instituto de Investigación Biomédica de Málaga - IBIMA.

1 Introduction

Machine learning models are trained from fixed datasets and stationary environments, often overcoming human-level ability. However, these models fail to emulate the process of human learning, which is efficient, robust, and able to learn incrementally, from sequential experience in a non-stationary world [1]. Continual learning is an increasingly relevant area of study that tries to improve the ability of modern learning systems to deal with non-stationary distributions [2]. The main challenge here is the stability-plasticity dilemma: learning requires plasticity for the integration of new knowledge, but also stability in order to prevent the forgetting of previous knowledge. Too much stability will impede the efficient coding of this data, whereas too much plasticity will result in previously encoded data being constantly forgotten [3]. This last problem is also known as catastrophic forgetting, which is defined as a complete forgetting of previously learned information by a neural network exposed to new information [4, 5]. Thus, there has been new research interested in the continual learning problem in recent years [6–9]. However, most of these techniques have focused on supervised and reinforcement learning by attempting to learn a series of tasks sequentially, but little attention has been paid to the use of these techniques in unsupervised learning for clustering tasks where input distributions can change over time.

Unsupervised learning aims to analyze a collection of unlabeled data and find structures or patterns without the intervention of a human expert. The information discovered by unsupervised machine learning algorithms is typically used to organize data into meaningful groups based on their similarities or differences [10]. However, it can also be used to establish relationships between features in a dataset [11] or to reduce the dimensionality of data with a large number of characteristics to a new compact data representation in which the features that are redundant or provide little information are not taken into consideration [12].

Competitive neural networks are classical unsupervised models that naturally categorize the input data [13]. They follow the winner-take-all training approach, where the weight vector of each of its processing units, known as neurons, are adapted so that it approximates the mean or centroid of a data cluster. On each training step, only the winning neuron is updated, that is, the neuron with the weight vector that is closest to the current input vector is the only one that change the cluster it represents by taking into account the information provided by the input vector.

Competitive models excel at adaptive vector quantization [14] and have been successfully applied to image processing. Regarding color image segmentation, [15] shows the efficiency of a competitive learning algorithm as a tool for clustering color space whereas [16] trains a competitive network with chromaticity samples of different colors in HSV space and uses it to determine the dominant colors that are used later in the segmentation phase. The ability of a probabilistic competitive neural network to compress multispectral image data is studied in [17]. In addition, in [18] a rival penalized competitive neural network is part of a system designed to recognize human behavior based on body postures in a video sequence.

The reminder of this paper is organized as follows. In Section 2, the proposed continual learning competitive model is defined. Experimental results are reported in Section 3. Finally, Section 4 is devoted to conclusions.

2 The Model

In a competitive learning network composed of K neurons, the training phase can be understood as a sequence of steps during which the network learns to represent the distribution of the input data. Each neuron i has a weight vector $w_i(t)$, which estimates the centroid of the cluster which the neuron represents. In each step t, a sample (data vector) $x(t)$ from the input data is presented to the network, and the selected neuron s is the one with the closest weight vector in the input space, i.e., the neuron whose weight vector is more similar to the input vector. The Euclidean distance is chosen as the measure to estimate the similarity between a neuron weight vector $w_i(t)$ and the input sample $x(t)$ at the training step t:

$$s = winner(t) = argmin_{1 \leq i \leq K} \left\| x(t) - w_i(t) \right\|_2 \tag{1}$$

Then, only the winning neuron s is updated according to the updating rule:

$$w_s(t+1) = w_s(t) + \alpha(t) \left(x(t) - w_s(t) \right) \tag{2}$$

In that rule, $\alpha(t)$ is a scalar factor called learning rate that represents the size of the correction, and is monotonically decreasing:

$$\alpha(t) = \frac{\alpha(t-1)}{d(t)} \tag{3}$$

where $d(t)$ is a decay function. Typically, the training phase is divided in two stages. In the first one the neurons are widely but crudely distributed to model the distribution of the input data. For that purpose, the learning rate usually decreases linearly, i.e., $d(t) = 1 + at$, or exponentially, i.e., $d(t) = e^{at}$, with $a \in \mathbb{R}^+$. After that, a second stage is intended to fine-tune the positions of the weight vectors and the learning rate is held constant at a low value.

After the training phase, the weights are fixed and each new input vector is assigned to the cluster corresponding to the neuron with the closest weight vector. If the training was successful, the weights of the neurons are distributed so that they represent the cluster centroids of a Voronoi tesselation.

Our goal is to adapt this model to make it able to learn continuously: after learning a specific input distribution, the neural network should be able to adapt and change the configuration of its weight vectors to represent a different input distribution. We seek to achieve this goal by using a different scaling strategy during training: instead of using a scaling factor α purely dependent on t, a function that relies on the Euclidean distances between samples and winning neuron weights $\left\| x(t) - w_s(t) \right\|_2$ in the last N time steps, from t to $t - N + 1$, is

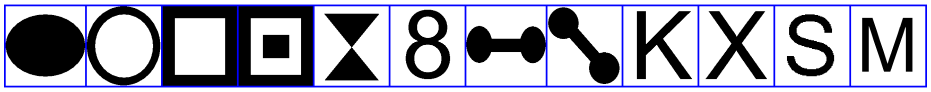

Fig. 1. Binary images used in the experiments, the black pixels in each one representing a rasterized shape.

defined. We will note the Euclidean distance as $e(t)$, since it can be seen as the error at each time step t.

The core idea is that if the input distribution changes, the $e(t)$ distances will increase, thus increasing the scaling factor α, which allows the neural model to adapt to the new input distribution and redistribute its neurons more effectively. In order to reduce the effect of noise and outliers, which could hinder the learning process of the network, the median of these distances is proposed as a consensus measure, noted as $C(t)$. Then, that value is used as input for a monotonically increasing function f:

$$\alpha(t) = f(C(t)) \tag{4}$$

Different monotonic function types are considered in the experiments:

- Linear: $\alpha(t) = aC(t)$
- Quadratic: $\alpha(t) = a\left(C\left(t\right)\right)^2$
- Cubic: $\alpha(t) = a\left(C\left(t\right)\right)^3$

Since in the context of the updating rule it only makes sense for $\alpha(t)$ to take values in the $[0 \ldots 1]$ range, the results of these functions are clamped to that interval. All of these functions have been modelled with just a single parameter a in order to limit the parameter space to search, as well as to avoid non-monotonic function shapes. Note that the number of parameters remains the same since parameter a replaces parameter of the same name from (3).

3 Experimental Results

3.1 Dataset

To test the proposed learning system, we use it to model rasterized shapes. Each shape comes from a binary image, as seen in Figure 1. For each binary image, the normalized coordinates of all black pixels are randomly sampled 10000 times. We perform training tests for sequences of two sampled shapes: in each test, we feed the learning system first 10000 samples from one shape, then 10000 samples from another shape, to test whether it can adapt to different models over time. The system will first adapt to model the input distribution of the first shape, then it will change to attempt to capture the second one.

To quantitatively evaluate the results of each experiment, we measure how good is the system at modeling the samples from the last shape presented to

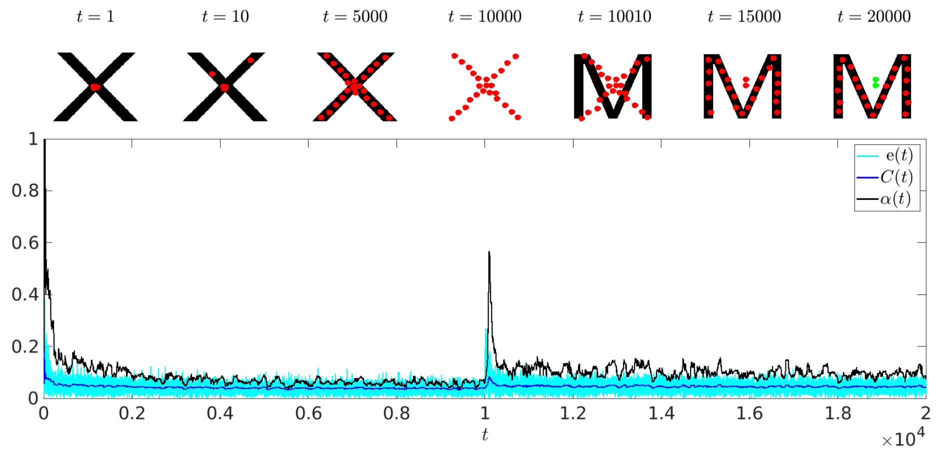

Fig. 2. One of the conducted experiments: from X to M, with $N = 50$, and a cubic function $\alpha(t) = 1000 \cdot C(t)^3$. *Above*: configuration of the neuron weight vectors (red dots) at different training steps, with the shape the samples are coming from in black (no background shape is drawn at $t = 10000$, as this is the time step when we finish feeding samples from the first shape, just before starting feeding samples from the new shape). At $t = 20000$, the neurons marked in green are dead, i.e., no sample is nearer to them than to any other one.*Below*: graph showing the evolution of e(t), $C(t)$ and $\alpha(t)$ through the online training process.

them, using the Mean Quantization Error:

$$\text{MQE} = \frac{1}{K}\sum_{i=1}^{K} \text{qe}_i$$

In the above equation, qe_i is the quantization error of neuron i, defined as the sum of distances between the final weight vector w_i (after training is completed) and the samples of its receptive field R_i, i.e., the $x_1, \ldots, x_j, \ldots x_{R_i}$ input samples that are closer to w_i than to any other weight vector:

$$\text{qe}_i = \sum_{j=1}^{R_i} \|x_j - w_i\|$$

For each sequence of two shapes, we perform an array of tests for all three possible function types (linear, quadratic, cubic), and for each function type a different array of a values, since each function type requires its a parameter to be in a different scale to approximately keep the output of the function roughly at the $[0 \ldots 1]$ range. We have logarithmically spaced (noted as l.s. in the following) the tested values:

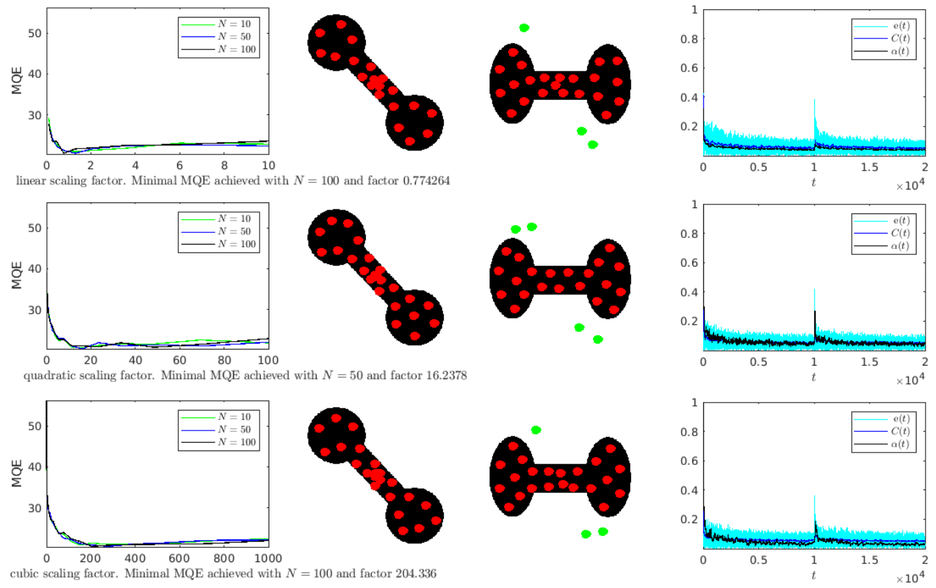

Fig. 3. Results for a sequence of two images: first the inclined dumbbell shape, then the horizontal version. Each row corresponds to a function type (linear, quadratic, cubic). *First column*: each graph shows the results of an array of experiments; we represent the MQE value achieved after the learning phase (one MQE for each experiment); and each line represents the MQE values for an array of experiments, all of them with a specific N and varying scaling factor a. The parameters achieving the best MQE for each function type are recorded below each graph. *Second, third and fourth columns*: in each row, for the experiment with the best MQE for the function type associated to that column, the second and third columns show the neurons (red dots) at $t = 10000$ (right after the samples from the first shape) and at $t = 20000$ (right after the samples from the second shape), respectively. In the third column, neurons marked in green are dead, i.e., they do not represent any sample. The graphs in the fourth column show the evolution of $e(t)$, $C(t)$ and $\alpha(t)$ for each respective experiment with the best MQE.

– For linear functions, 10 l.s. values between 0.1 and 10.
– For quadratic functions, 20 l.s. values between 0.1 and 100.
– For cubic functions, 30 l.s. values between 0.1 and 1000.

We experiment with three values for N, the size of the time window to compute $C(t)$: 10, 50 and 100. Additionally, we fix the number of neurons at $K = 25$, but please note that (as a result) steady-state mean quantization error (MQE) values will also depend on the area and geometry of the shape: the larger the area, the larger the subsets of samples closer to each neuron, naturally leading to bigger MQE values. We also initialize each neuron's $w_i(0)$ to the mean of a random selection of $10000/K$ samples from the first shape, which in practice tends to cluster together all neurons towards the centroid of the distribution.

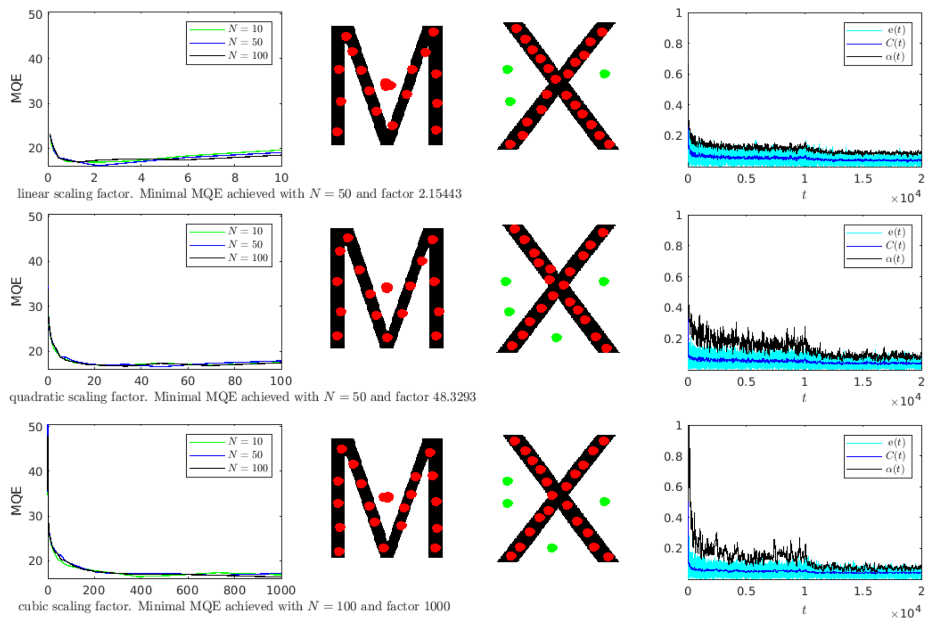

Fig. 4. Results for a sequence of two images: first the shape of the letter M, then the shape of the letter X. The arrangement of the array of results is the same as in Figure 3. Please refer to the caption of that Figure. In the third column, we have marked in green the neurons that are dead, i.e., no sample from the second image is nearer to them than to other ones. Note that in the examples shown in the second, third and fourth columns, there are no dramatic increases of $\alpha(t)$ after $t = 10000$, but the arrangement of the shapes is such that the neurons rearrange into a good clustering of the second shape with a continuously low $\alpha(t)$.

3.2 Results

The learning system is able to adapt when new distributions are fed to it. For example, in Figure 2 we can see an experiment where we feed our learning system first the shape of the letter X, then the shape of the letter M. As it can be seen, when the samples fed to the system are switched to the second shape, the value of $e(t)$ significantly increases, leading to an increment of $\alpha(t)$, so that the system adapts to the new shape.

In fact, we have checked that the system is able to adapt to new shapes with experiments testing many parameter combinations and many different sequences of shapes. However, there is no combination of function type, scaling factor a and N that minimizes MQE for all possible sequences of input shapes. Figure 3 shows a summary of the experiments where the system is presented first the inclined dumbbell shape, (⟍), then the horizontal one (⟷). The figure is arrayed so each row shows results for experiments with a specific function type (linear, quadratic, cubic). The first column of the figure shows a graph for each function type: summarizing the MQE achieved for experiments with different values for N and

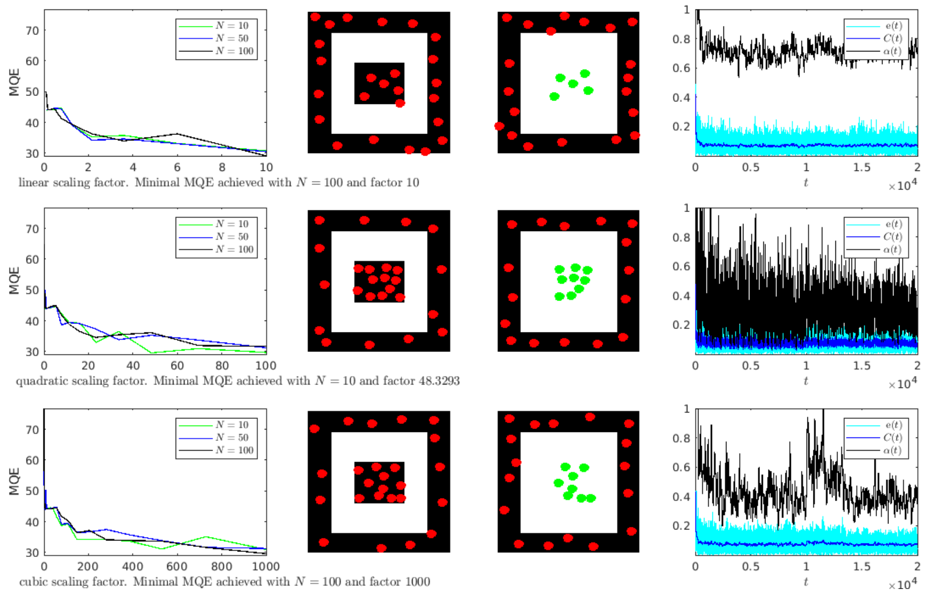

Fig. 5. Results for a sequence of two images: first a small square inside a wall, the just the wall. The arrangement of the array of results is the same as in Figure 4. Please refer to the caption of that Figure. Since the shapes are such that many neurons end up dead in the center, the best results are achieved by maintaining consistently high $\alpha(t)$ values and thus very jittery movements of neurons. In the experiment shown in the second, third and fourth plots, this behavior enabled one of the neurons inside the inner square to be recruited into modeling the wall of the second shape, lowering the overall MQE.

a. Among those, we select the combination with minimal MQE and we show the state of the neurons for that experiment right after all the samples from the first shape (second column), and after all the samples from the second shape (third column). Note that in the third column, we can see some dead neurons that do not represent any sample from that shape; this is caused because, depending on the specific arrangement of the shapes, some neurons are never selected again in the training process. Finally, the fourth column shows $e(t)$, $C(t)$ and $\alpha(t)$ for that same experiment. Please note that in the middle of these graphs (at $t = 10000$) the learning system has adapted to the first shape and starts to be presented with samples from the second shape, sharply but temporarily increasing $C(t)$ and $\alpha(t)$, which then go again to a background level when the learning system adapts to the new shape.

Our expectation was that the system adaptation to new shapes (i.e. input distribution) would be determined by a significantly increase in the value of $e(t)$, leading to an increment of $\alpha(t)$, in turn enabling the system to adapt to the new shape. However, while this behavior could be observed in some experiments such as the ones shown in Figures 2 and 3), this did not happen always. Instead, sometimes $\alpha(t)$ would not change significantly, meaning that within the constraints of the parameters of the training process, the system detected that very slow adaptation rates were enough to guarantee the neurons to arrange into a good clustering of the second shape. An example can be seen in Figure 4. In other cases, the second shape is easy to represent many neurons are not needed. They likely end up as dead and they do not contribute to the clustering because their receptive fields are empty. In theses cases the best $\alpha(t)$ functions induce consistently high $\alpha(t)$ values, which allow the neurons to have initial jittery trajectories that prevent them to be dead during the online training, thus contributing to an overall lower MQE (an example is shown in Figure 5).

4 Conclusions

In this paper, a novel continual learning approach for competitive neural networks is proposed. The main characteristic of this competitive neural network is the ability to learn continuously by changing the traditional learning rate which decreases linearly for a function of the N most recent $e(t)$ values, so that the learning rate $\alpha(t)$ increases as the $e(t)$ increment when input samples that belong to a different input distribution are presented. Different monotonic function types for the learning rate are considered, namely linear, quadratic, cubic, and exponential.

Experimental results with two-dimensional binary images show that the model is able to adapt to a new input distribution when a different input distribution has been previously presented. Actually, the model is able to find a function to adapt to the new input distribution in the most proper way, which sometimes is a peak and sometimes is similar to a low or a high constant function. However, the best combination of scaling factor a, N, and function type for minimizing the MQE depends on the input distribution. Moreover, we observed dead neurons that do not represent any sample from the input distribution at hand. Nevertheless, this can be solved by using a self-organizing mechanism in addition to a competitive mechanism, which is the research line we plan to follow in future works.

References

1. R. Hadsell, D. Rao, A. A. Rusu, and R. Pascanu, "Embracing Change: Continual Learning in Deep Neural Networks," *Trends in Cognitive Sciences*, vol. 24, no. 12, pp. 1028–1040, dec 2020.
2. D. Rao, F. Visin, A. A. Rusu, Y. W. Teh, R. Pascanu, and R. Hadsell, "Continual unsupervised representation learning," in *Advances in Neural Information Processing Systems*, vol. 32. Neural information processing systems foundation, oct 2019.
3. M. Mermillod, A. Bugaiska, and P. Bonin, "The stability-plasticity dilemma: investigating the continuum from catastrophic forgetting to age-limited learning effects," *Frontiers in Psychology*, vol. 4, p. 504, 2013.

4. M. McCloskey and N. J. Cohen, “Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem,” *Psychology of Learning and Motivation - Advances in Research and Theory*, vol. 24, no. C, pp. 109–165, jan 1989.
5. I. J. Goodfellow, M. Mirza, D. Xiao, A. Courville, and Y. Bengio, “An empirical investigation of catastrophic forgetting in gradient-based neural networks,” in *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*. International Conference on Learning Representations, ICLR, dec 2014.
6. H. Shin, J. K. Lee, J. Kim, and J. Kim, “Continual learning with deep generative replay,” in *Advances in Neural Information Processing Systems*, vol. 2017-Decem. Neural information processing systems foundation, may 2017, pp. 2991–3000.
7. F. Zenke, B. Poole, and S. Ganguli, “Continual learning through synaptic intelligence,” in *34th International Conference on Machine Learning, ICML 2017*, vol. 8. International Machine Learning Society (IMLS), mar 2017, pp. 6072–6082.
8. C. V. Nguyen, Y. Li, T. D. Bui, and R. E. Turner, “Variational continual learning,” in *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*. International Conference on Learning Representations, ICLR, oct 2018.
9. J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell, “Overcoming catastrophic forgetting in neural networks,” *Proceedings of the National Academy of Sciences of the United States of America*, vol. 114, no. 13, pp. 3521–3526, dec 2016.
10. A. K. Jain, “Data clustering: 50 years beyond k-means,” *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010.
11. M.-S. Chen, J. Han, and P. Yu, “Data mining: an overview from a database perspective,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 8, no. 6, pp. 866–883, 1996.
12. J. I. T. and C. Jorge, “Principal component analysis: a review and recent developments,” *Phil. Trans. R. Soc. A.*, vol. 374:20150202, 2016.
13. D. E. Rumelhart and D. Zipser, “Feature Discovery by Competitive Learning*,” *Cognitive Science*, vol. 9, no. 1, pp. 75–112, jan 1985.
14. S. C. Ahalt, A. K. Krishnamurthy, P. Chen, and D. E. Melton, “Competitive learning algorithms for vector quantization,” *Neural Networks*, vol. 3, no. 3, pp. 277–290, 1990. [Online]. Available: https://www.sciencedirect.com/science/article/pii/089360809090071R
15. T. Uchiyama and M. A. Arbib, “Color image segmentation using competitive learning,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 12, p. 1197–1206, dec 1994. [Online]. Available: https://doi.org/10.1109/34.387488
16. F. García-Lamont, J. Cervantes, and A. e. a. López-Chau, “Automatic computing of number of clusters for color image segmentation employing fuzzy c-means by extracting chromaticity features of colors,” *Pattern Anal. Applic.*, vol. 23, pp. 59–84, 2020.
17. E. López-Rubio and J. M. Ortiz-de-Lazcano-Lobato, “Dynamic competitive probabilistic principal components analysis,” *Int. J. Neural Syst.*, vol. 19, no. 2, pp. 91–103, 2009. [Online]. Available: https://doi.org/10.1142/S0129065709001860
18. H. Yuan, C.-H. Duo, and W. hua Niu, “A human behavior recognition method based on latent semantic analysis,” *J. Inf. Hiding Multim. Signal Process.*, vol. 7, pp. 489–498, 2016.