Executive Summary of the Thesis

A brain-inspired approach to overcome catastrophic forgetting in Hebbian deep neural networks

Laurea Magistrale in Computer Science and Engineering - Ingegneria Informatica

Author: Riccardo Casciotti

Advisor: Prof. Alberto Antonietti

Co-advisors: Francesco De Santis,, Prof. Alessandra Pedrocchi

Academic year: 2024-2025

1. Introduction

Artificial Neural Networks (ANNs) are mathematical models designed to mimic brain functionalities, consisting of interconnected artificial neurons and synapses represented as connection weights. However, despite their biological inspiration, artificial models differ significantly from real brains, specifically in the way they learn new things.

1.1. Human Lifelong Learning versus AI Continual Learning

Human lifelong learning is the ability to continually adapt to new tasks without losing previously acquired knowledge. In artificial intelligence, learning seeks to replicate this capability, but it faces the major challenge of catastrophic forgetting. Indeed, learning a new task causes the model to perform poorly on an old task that it had previously learned, essentially because the new learning has overwritten the network parameters that stored the old task. The root cause is that the network's weights are used to store knowledge for all tasks, so gradient-based training on a new task will update those weights in favor of the new task's objectives, with no inherent regard for past tasks. Balancing plasticity (learning new tasks) and memory stability (retaining past knowledge) is thus essential. Taking inspiration from biological systems like the human brain offers a promising path toward solving catastrophic forgetting in ANNs.

1.2. Hebbian Learning and Neuromodulation

In the human brain, learning and memory formation is regulated through synaptic plasticity, which is the ability of synapses to change their strength based on activity. Synaptic plasticity is defined by a well-known principle: Hebbian Learning, which is summarized in the following statement "Neurons that fire together, wire together". The statement refers to the behavior of the synapse in response to neuron activations. Indeed, if a pre-synaptic neuron promotes the activation of post-synaptic neurons, then the synapse strengthens, on the other hand, if the former does not promote any kind of activation into post-synaptic neurons, the synapse weakens.
However, Hebbian learning alone is not enough to stimulate effective and lasting learning, which relies also on the use of neuromodulation. Neuromodulators are several molecules released

by neurons in order to modify the plasticity rules controlling their synapses. For instance, the presence of dopamine in the hippocampus has shown to be directly related to memory formation and information consolidation [3]. In this context, neuromodulators help to determine which synaptic changes are reinforced—effectively guiding Hebbian mechanisms to prioritize and store significant information while filtering out less relevant details. Moreover, neuromodulators contribute not only to local changes in synaptic strength but also to structural plasticity, influencing the growth, retraction, and reorganization of neural connections throughout the brain.

1.3. Objective

The principle under which the human brain works makes it a unique natural marvel. Its complexity and its efficacy in learning new tasks make it a viable and worth studying solution to find answers to artificial intelligence problems. This work implements a brain-inspired solution in an existing bio-plausible architecture, to overcome catastrophic forgetting in a Task-Free continual learning scenario[4]. The model adopted, called *SoftHebb* [1], consists in a Convolutional Neural Network with a weight update rule that directly recalls Hebbian learning happening in human synapses. The main objective of this project has been to conceptualise and apply a neuromodulation-inspired strategy on *SoftHebb* to improve its performance in different continual learning scenarios.

2. Methods

As previously stated, we opted for the utilization of *SoftHebb* [1], which is a deep Convolutional Neural Network trained in a completely unsupervised manner, only by leveraging the correlation between data samples to acquire information. The model architecture consists of a feature extractor and a classification layer, which for simplicity we are going to call *head* from now on. The feature extractor is the section of the network that selects the most significant feature for an image, thus playing a key role in the performance of a model. For each layer of the feature extractor, a kernel (a small matrix of learnable weights) slides over the input image and is convolved with the input by computing an element-wise multiplication between the kernel and that patch, then summing all the products. This produces a single scalar value for that particular spatial location. The resulting value is typically passed through a non-linear activation function. The output of this function is what we call the activation value. This value represents how strongly the kernel has detected a specific feature at that location in the input. As the kernel slides over all spatial positions of the input, the activation values computed at each position form a feature map. This map highlights the presence and strength of the feature the kernel is designed to detect.
The head is instead the last layer of the network, which outputs the final prediction of the model, receiving a flattened feature map as input and providing the image classification. If the head's weights get overwritten, the output of the model is also going to be different, regardless of the features received in input. For these reasons, we have designed two different solutions for continual learning, one implemented on the feature extractor and one implemented on the head of the model.

2.1. Kernel Plasticity neuromodulation

The solution applied to the feature extractor takes inspiration from the natural neurotransmitters present in the brain, such as dopamine, used to cement newly learned information and preserve it in time [3].
Kernel Plasticity neuromodulation is an approach composed of two main steps:

- study the average weight update for the kernels during training on a task;
- rank the kernels based on their activation value during training on a task;

The first step aims to understand how the kernels change during training on a task. The technique keeps track of the weights every i batches and simply computes how they have changed with respect to the previous step by calculating the difference. Then, by cumulatively summing all the cells of this weight difference matrix, a single weight change value is computed and associated to each kernel. For a kernel j, let the weight at the beginning of the ith interval be $w_j^{(i-1)}$ and at the end be $w_j^{(i)}$. Then, the weight change during interval i is given by:

$$\Delta w_j^{(i)} = w_j^{(i)} - w_j^{(i-1)}.$$

The average weight change over N intervals can be defined using the cumulative change:

$$\overline{\Delta w_j} = \frac{1}{N}\sum_{i=1}^{N} |\Delta w_j^{(i)}|,$$

At the end of this process, a vector containing the average weight change per every kernel in every layer is stored in the model.
The second step calculates a cumulative sum of the activation values for each kernel every i batches of data samples. After the training phase is completed, the network calculates a final activation value to associate with each kernel and then sorts them based on this key value, having the kernels with the highest value at the top of the list. Finally, it considers only the first *top k* kernels in this list as important for the just seen task and stores them permanently in a vector. Let the activation value of kernel j at interval i be $a_j^{(i)}$ (an average or sum over spatial dimensions). The cumulative activation for kernel j is:

$$A_j = \sum_{i=1}^{N} a_j^{(i)}.$$

The kernels are then ranked based on A_j and the indices corresponding to the top K kernels are stored:

$$\mathcal{K} = \{j : A_j \text{ is among the top } K \text{ values}\}.$$

The two vectors *average kernel weight update* and *top k kernels* per task are crucial to apply the neuromodulation. The process is made up of the following steps, which need to be followed during training on a new task:

- check if the incoming weight update surpasses the average kernel weight change stored in the tensor for previous training sessions;
- check if the current kernel is among the *top k* kernels through the stored tensor.

If both conditions above are met, the incoming weight update is increased on kernels that are not among the *top k* and decreased among the *top k* kernels that break the threshold of the previously stored average weight update. So, for an incoming weight update Δw_j^{new}, first a global modulation condition is defined that checks whether at least one kernel within the top-k set K surpasses its average weight-update threshold:

$$\text{condition} = \exists\, k \in K : \|\Delta w_k^{new}\| > \Delta w_k$$

Then, the neuromodulation of kernel weight update is:

$$\Delta w_j^{mod} = \begin{cases} \alpha\, \Delta w_j^{new}, & \text{if (cond=True) and } j \notin K, \\ \beta\, \Delta w_j^{new}, & \text{if } j \in K \text{ and } \|\Delta w_j^{new}\| > \Delta w_j, \\ \Delta w_j^{new}, & \text{otherwise.} \end{cases}$$

where $\alpha > 1$ enhances plasticity for less-critical kernels, and $0 < \beta < 1$ protects the stability of important kernels. Thus, the increased plasticity for kernels not included in the top-k set ($j \notin K$) is triggered *only* when at least one kernel from the top-k kernels surpasses its average threshold, signaling active learning in important kernels.

2.2. Multi-Head solution

The last fully connected layer needs to be protected from catastrophic forgetting as well. Thus, the project develops a continual learning solution to be applied to the head of the model by creating a multi-head architecture. When the model encounters a new task T_{new}, it proceeds as follows:

1. **Resume the Feature Extractor:** The feature extractor ϕ is loaded with the weights learned so far.
2. **Instantiate a New Head:** A new head H_{new} is created. At the beginning of training on task T_{new}, the network output is given by:

$$y = H_{\text{new}}\big(\phi(x)\big), \quad \text{with } x \in R^d.$$

3. **Train and Save the Head:** After training, the new head H_{new} is detached from the feature extractor and stored in the set of heads:

$$\mathcal{H} = \mathcal{H} \cup \{H_{\text{new}}\}.$$

During inference, the task label is not available. Instead, the model selects the most appropriate head based on the outputs over a dataset X (without target labels). Let $X \in R^{B\times d}$ be a batch of B samples. For each stored head $H_t \in \mathcal{H}$, the output is computed as:

$$Y_t = H_t\big(\phi(X)\big) \in R^{B\times N},$$

where N is the number of classes for the task corresponding to H_t.
For each head t, define a cumulative score S_t by first taking the maximum response in the class dimension for each sample and then summing these maxima over the batch:

$$S_t = \sum_{i=1}^{B} \max_{j=1,\dots,N} \{(Y_t)_{ij}\}.$$

The head selection is then defined as choosing the head with the highest cumulative score:

$$t^* = \arg\max_{t\in\mathcal{H}} S_t.$$

The final prediction for an input x is obtained using the selected head:

$$y = H_{t^*}\big(\phi(x)\big).$$

3. Results

The two implemented solutions are validated through a set of experiments conducted on CIFAR-10 and CIFAR-100 datasets.

3.1. Training and setup

The experiments are carried out to test the performance of the model in an incremental learning scenario. Thus, they are organized as follows: the different tasks consist in the classification of disjoint subsets of classes of the same dataset and they are learned sequentially. Once the training phase is completed, the experiments run an evaluation phase to study the effects of the catastrophic forgetting and the efficacy of the solutions implemented against it. The number of tasks per experiment is variable, and so is the number of classes to categorize associated with each single task. Though, an important characteristic is that the number of classes per task stays consistent throughout the same experiment, but may vary among different experiments.
The experiments compare the following four different models:

- **V-model**: the vanilla model with no continual learning solution implemented;
- **M-model**: the model which has only the multi-head solution;
- **KP-model**: the model which has only the kernel plasticity neuromodulation used;
- **KPM-model**: the model with both kernel plasticity neuromodulation and multi-head.

In the following paragraphs, three different analyses are presented, studying the impact of the number of tasks proposed to the model, the number of layers in the feature extractor, and the number of classes per task in the performance of the four models.

3.1.1 Incremental Learning results with regard to tasks number

The experiments were conducted keeping the number of classes per task fixed to two. Thus, experiments with 2, 4, 6, 8, and 10 tasks were executed both on CIFAR-100 and CIFAR-10 [2]. The experiments ran with incremental learning of two tasks did not show any notable performance improvement between the KPM-model and the M-model, suggesting that the network is using almost all the information from the previous task for the evaluation of the next one. However, there is a big performance increase between the V-model and the M-model, where the latter outperformed the former of almost 30% in evaluation of task 0.
Nevertheless, an interesting behavior is observed when we increase the task number to four (Figure 1).

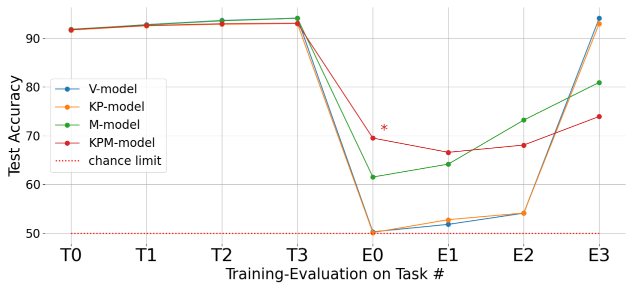

Figure 1: Accuracy results on incremental learning of 4 tasks with 2 classes per task belonging to CIFAR-100 on 80 experiments. The * highlights the tasks where the p-value is below 0.05 (Wilcoxon Signed-Rank Test comparing KPM vs M models).

The neuromodulation technique applied to the feature extractor combined with the multi-head solution (KPM-model) effectively mitigates catastrophic forgetting, especially on earlier tasks (Task 0 and Task 1), outperforming both the M-model and V-model. The results strongly suggest that kernel plasticity neuromodulation is essential for preserving knowledge of initial tasks, with significant statistical evidence for improved retention on Task 0. Although the performance on Task 1 also improved, statistical significance was not achieved. On later tasks, however, the unconstrained plasticity of the M-model provided better adaptability, making it preferable for these newer tasks. Overall, the KPM-model achieves a beneficial balance between retaining earlier knowledge and learning new tasks, thus enhancing stability, accuracy balance across tasks, and generalizability.

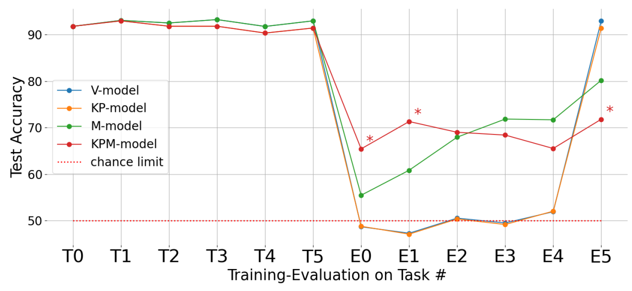

Figure 2: Accuracy results on incremental learning of 6 tasks with 2 classes per task belonging to CIFAR-100 on 80 experiments.

We then increased the task number to six (Figure 2).The results confirm the hypothesis that earlier tasks' performances are preserved by the KPM-model in comparison to the other configurations. In particular, it outperforms all the other models on the first three tasks learned, but the statistical testing suggests that the model is preferable only the first two tasks. However, it is important to remember that the overall objective of the KPM-model is not to outperform the other models entirely, but rather to solve or mitigate the catastrophic forgetting effect of incrementally learned tasks and this objective is well reached. Indeed, the behavior shows a redistribution of the performance, which is reduced in later tasks and increased in earlier tasks. Differently, the M-model, the KP-model, and the V-model show much higher performance on later tasks, which is exactly the behavior caused by catastrophic forgetting, where newer tasks tend to overwrite all the previously learned weights, completely destroying the acquired information on previous tasks. In support of this hypothesis, the following are the overall performance metrics of the four models tested:

Num. of Tasks	V-model	KP-model
2	72.6	72.8
4	62.55	62.5
6	56.81	56.48
8	56.9	56.6
10	55.8	55.76

Table 1: Overall performance of V-model and KP-model on incremental learning of tasks based on classes of CIFAR-100.

Num. of Tasks	M-model	KPM-model
2	80.15	79.7
4	69.9	69.5
6	68.99	68.57
8	66.3	64.5
10	65	63

Table 2: Overall performance of KPM-model and M-model on incremental learning of tasks based on classes of CIFAR-100.

The same testing scenarios were implemented on CIFAR-10 as well, to confirm that the advantages of the neuromodulation technique were not only relative to the CIFAR-100 dataset. While the accuracy performance on CIFAR-10 is worse than the one on CIFAR-100, the following results all show that the KPM-model allows to successfully retain the earlier learned tasks after the training phase finishes, outperforming all the other models (Figure 3).

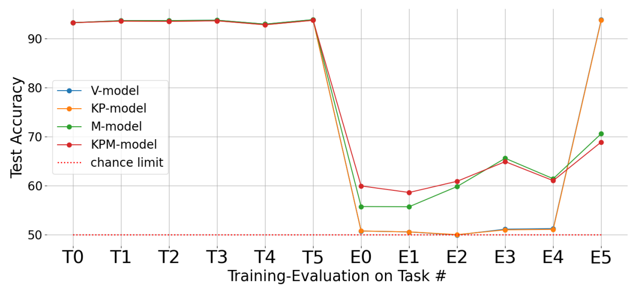

Figure 3: Accuracy results on incremental learning of 6 tasks with 2 classes per task belonging to CIFAR-10 on 250 experiments.

3.1.2 Incremental Learning with regard to different layer numbers

The *SoftHebb* model [1] with three convolutional layers and one fully connected layer showed minimal improvements in continual learning scenarios, with significant performance gain only on the first task. Overall, the KPM-model's benefits over the baseline (M-model) were limited, suggesting that the network size may be too small to effectively address catastrophic forgetting. The large accuracy gap between early and later tasks indicates poor stability, limited generalization, and excessive plasticity compared to memory retention.
The next architecture tested consists of five convolutional Hebbian layers and one fully connected classifier head, significantly improving performance compared to the four-layer *SoftHebb* model. The larger network enhances stability, generalization, and the plasticity-memory trade-off, showing a better redistribution of accuracy across tasks. The KPM-model notably outperforms the M-model on the first two tasks, with statistical confirmation, and achieves overall accuracy gains of approximately 3.5%.
The last architecture tested is composed of eleven layers: ten Hebbian convolutional layers and one fully connected layer for the head of the classifier. The performances are overall much worse than the six-layer architecture, where there is no real evidence that using the KPM-model is better than using the M-model. This result is also in accordance with the work [1] where increasing the layers too much does not bring any real performance improvement and actually seems to damage the overall accuracy of the model.
Thus, having done these different testing scenarios with different model architectures, all the tests used the six-layer deep neural network.

3.1.3 Incremental Learning about the number of classes per task

The final part of the study revolves around studying the behavior of the model based on the number of classes associated to a task.
Increasing the number of classes per task means that the network has to learn much more information per task and means that it has to be able to protect and use the previously stored information across tasks as much as possible. Indeed, a general decrease in accuracy is the first clear indicator of this situation. Increasing the number of tasks from two to four causes more than an average 14.5% accuracy drop on the KPM-model and even worse on the other models (Figure 4). Having said that, the KPM-model, still manages to outperform the other models, including the M-model. This is confirmed by the statistical testing, with p-value < 0.05 for the first two tasks and the confidence intervals which do not cross zero.

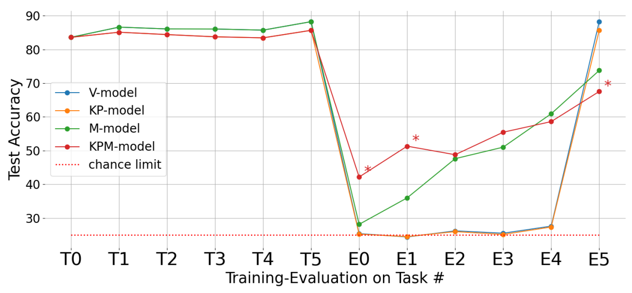

Figure 4: Accuracy results on incremental learning of 6 tasks with 4 classes per task belonging to CIFAR-100 on 80 experiments.

Finally, increasing the class number to six per task brings the performance down even more and the KPM-model is not able to protect the earlier tasks from the effects of catastrophic forgetting more than the M-model (Figure 5). This confirms the initial considerations that increasing the number of classes per task is going to increase the effects of catastrophic forgetting due to the increased amount of information that has to be retained.

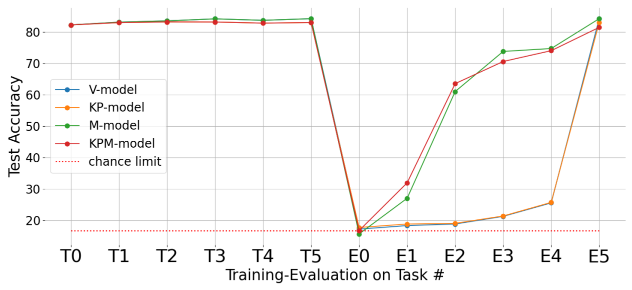

Figure 5: Accuracy results on incremental learning of 6 tasks with 6 classes per task belonging to CIFAR-100 on 80 experiments.

4. Conclusions

This study addressed catastrophic forgetting in Hebbian deep neural networks using bio-inspired neuro-modulation techniques, aiming to show that biologically plausible solutions can be good alternatives to more traditional algorithms. The focus was on incremental image classification tasks, balancing memory stability with plasticity. Results indicate that applying brain-inspired features both to the feature extractor (inspired by neuromodulation) and to the classifier head (multi-head model inspired by brain submodules) achieves around 70% accuracy across incremental tasks. Performance remains satisfactory even with increased task complexity, significantly outperforming random guessing. Additionally, the bio-inspired approach greatly reduces training epochs compared to traditional methods. In fact, by leveraging local Hebbian learning rules all models based on *SoftHebb* converge within a single unsupervised epoch. demonstrating strong potential for continual learning scenarios. To the best of our knowledge, this is the first kind of work that uses a Hebbian deep neural network in a continual learning scenario and that successfully overcomes catastrophic forgetting.

References

[1] Adrien Journé, Hector Garcia Rodriguez, Qinghai Guo, and Timoleon Moraitis. Hebbian Deep Learning Without Feedback. Technical report.

[2] Alex Krizhevsky. Learning Multiple Layers of Features from Tiny Images. Technical report, 2009.

[3] Arjun Magotra and Juntae Kim. Neuromodulated dopamine plastic networks for heterogeneous transfer learning with hebbian principle. *Symmetry*, 13(8), 8 2021.

[4] Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A Comprehensive Survey of Continual Learning: Theory, Method and Application. 1 2023.