Sparse Coding in a Dual Memory System for Lifelong Learning

Fahad Sarfraz*, **Elahe Arani***, **Bahram Zonooz**

[1]Advanced Research Lab, NavInfo Europe, The Netherlands
[2]Department of Mathematics and Computer Science, Eindhoven University of Technology, The Netherlands
fahad.sarfraz@navinfo.eu, elahe.arani@tue.nl, bahram.zonooz@gmail.com

Abstract

Efficient continual learning in humans is enabled by a rich set of neurophysiological mechanisms and interactions between multiple memory systems. The brain efficiently encodes information in non-overlapping sparse codes, which facilitates the learning of new associations faster with controlled interference with previous associations. To mimic sparse coding in DNNs, we enforce activation sparsity along with a dropout mechanism which encourages the model to activate similar units for semantically similar inputs and have less overlap with activation patterns of semantically dissimilar inputs. This provides us with an efficient mechanism for balancing the reusability and interference of features, depending on the similarity of classes across tasks. Furthermore, we employ sparse coding in a multiple-memory replay mechanism. Our method maintains an additional long-term semantic memory that aggregates and consolidates information encoded in the synaptic weights of the working model. Our extensive evaluation and characteristics analysis show that equipped with these biologically inspired mechanisms, the model can further mitigate forgetting. Code available at https://github.com/NeurAI-Lab/SCoMMER.

Introduction

The ability to continually acquire, consolidate, and retain knowledge is a hallmark of intelligence. Particularly, as we look to deploy deep neural networks (DNNs) in the real world, it is essential that learning agents continuously interact and adapt to the ever-changing environment. However, standard DNNs are not designed for lifelong learning and exhibit catastrophic forgetting of previously learned knowledge when required to learn tasks sequentially from a stream of data (McCloskey and Cohen 1989).

The core challenge in continual learning (CL) in DNNs is maintaining an optimal balance between plasticity and the stability of the model. Ideally, the model should be stable enough to retain previous knowledge while also plastic enough to acquire and consolidate new knowledge. Catastrophic forgetting in DNNs can be attributed to the lack of stability, and multiple approaches have been proposed to address it. Among them, *Rehearsal-based* methods, (Riemer et al. 2018; Aljundi et al. 2019b) which aim to reduce forgetting by continual rehearsal of previously seen tasks, have proven to be an effective approach in challenging CL tasks (Farquhar and Gal 2018). They attempt to approximate the joint distribution of all the observed tasks by saving samples from previous tasks in a memory buffer and intertwine the training of the new task with samples from memory. However, due to the limited buffer size, it is difficult to approximate the joint distribution with the samples alone. There is an inherent imbalance between the samples of previous tasks and the current task. This results in the network update being biased towards the current task, leading to forgetting and recency bias in predictions. Therefore, more information from the previous state of the model is needed to better approximate the joint distribution and constrain the update of the model to preserve the learned knowledge. However, it is still an open question what the optimal information is for replay and how to extract and preserve it.

The human brain provides an existence proof for successful CL in complex dynamic environments without intransigence or forgetting. Therefore, it can provide insight into the design principles and mechanisms that can enable CL in DNNs. The human brain maintains a delicate balance between stability and plasticity through a complex set of neurophysiological mechanisms (Parisi et al. 2019; Zenke, Poole, and Ganguli 2017) and the effective use of multiple memory systems (Hassabis et al. 2017). In particular, evidence suggests that the brain employs *Sparse Coding,* whereby the neural code is characterized by strong activations of a relatively small set of neurons. The efficient utilization of sparsity for information representation enables learning new associations faster with controlled interference with previous associations while maintaining sufficient representation capacity. In addition, complementary learning systems (CLS) theory posits that effective learning requires two complementary learning systems. The hippocampus rapidly encodes episodic information into non-overlapping representations, which are then gradually consolidated into the structural knowledge representation in the neocortex through the replay of neural activities.

Inspired by these mechanisms in the brain, we hypothesize that employing a mechanism to encourage sparse coding in DNNs and mimic the interplay of multiple memory systems can be effective in maintaining a balance between

*These authors contributed equally.

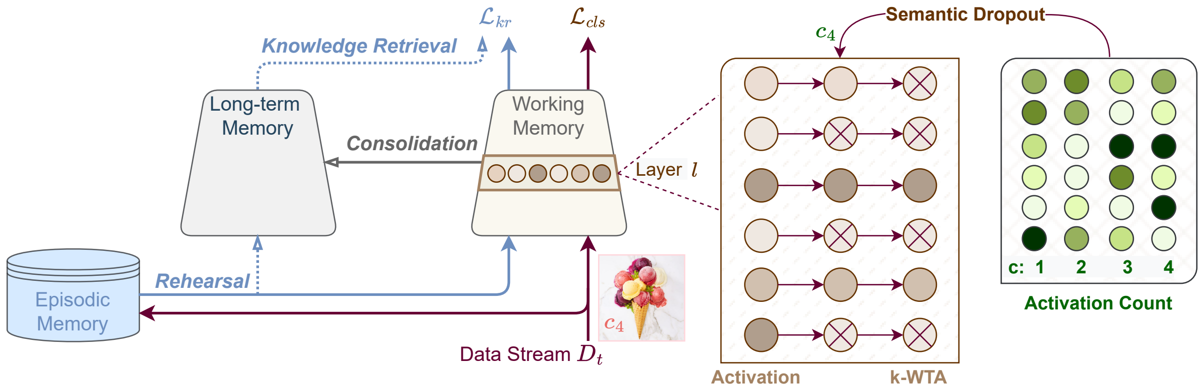

Figure 1: SCoMMER employs sparse coding in a multi-memory experience replay mechanism. In addition to the instance-based episodic memory, we maintain a long-term memory that consolidates the learned knowledge in the working memory throughout training. The long-term memory interacts with the episodic memory to enforce consistency in the functional space of working memory through the knowledge retrieval loss. To mimic sparse coding in the brain, we enforce activation sparsity along with semantic dropout, whereby the model tracks the class-wise activations during training and utilizes them to enforce sparse code, which encourages the model to activate similar units for semantically similar inputs. Schematic shows how the activations from layer l are propagated to the next layer. Darker shades indicate higher values. Given a sample from class 4, semantic dropout retains the units with higher activation counts for the class, and top-k remaining (here 2) units with higher activations are propagated to the next layer. This enables the network to form semantically conditioned subnetworks and mitigate forgetting.

stability and plasticity. To this end, we propose a multi-memory experience replay mechanism that employs sparse coding, SCoMMER. We enforce activation sparsity along with a complementary dropout mechanism, which encourages the model to activate similar units for semantically similar inputs while reducing the overlap with activation patterns of semantically dissimilar inputs. The proposed semantic dropout provides us with an efficient mechanism to balance the reusability and interference of features depending on the similarity of classes across tasks. Furthermore, we maintain an additional long-term semantic memory that aggregates the information encoded in the synaptic weights of the working memory. Long-term memory interacts with episodic memory to retrieve structural knowledge from previous tasks and facilitates information consolidation by enforcing consistency in functional space.

Our empirical evaluation on challenging CL settings and characteristic analysis show that equipping the model with these biologically inspired mechanisms can further mitigate forgetting and effectively consolidate information across the tasks. Furthermore, sparse activations in conjunction with semantic dropout in SCoMMER leads to the emergence of subnetworks, enables efficient utilization of semantic memory, and reduces the bias towards recent tasks.

Related Work

The different approaches to addressing catastrophic forgetting in CL can be broadly divided into three categories: *Regularization-based* methods regularize the model in the parameter space (Farajtabar et al. 2020; Kirkpatrick et al. 2017; Ritter, Botev, and Barber 2018; Zenke, Poole, and Ganguli 2017) or the functional space (Rannen et al. 2017; Li and Hoiem 2017), *Dynamic architecture* expands the network to dedicate a distinct set of parameters to each task, and *Rehearsal-based* methods (Riemer et al. 2018; Aljundi et al. 2019b) mitigate forgetting by maintaining an episodic memory buffer and continual rehearsal of samples from previous tasks. Among these, our method focuses on rehearsal-based methods, as it has been proven to be an effective approach in challenging CL scenarios (Farquhar and Gal 2018). The base method, Experience Replay (ER) (Riemer et al. 2018) interleaves the training of the current task with the memory samples to train the model on the approximate joint distribution of tasks. Several studies focus on the different aspects of rehearsal: memory sample selection (Isele and Cosgun 2018), sample retrieval from memory (Aljundi et al. 2019a), and what information to extract and replay (Ebrahimi et al. 2020; Bhat, Zonooz, and Arani 2022).

Dark Experience Replay (DER++) samples the output logits along with the samples in the memory buffer throughout the training trajectory and applies a consistency loss on the update of the model. Recently, CLS theory has inspired a number of approaches that utilize multiple memory systems (Wang et al. 2022a,b; Pham, Liu, and Hoi 2021) and show the benefits of multiple systems in CL. CLS-ER (Arani, Sarfraz, and Zonooz 2022) mimics the interplay between fast and slow learning systems by maintaining two additional semantic memories that aggregate the weights of the working model at different timescales using an exponential moving average. Our method enforces sparse coding for efficient representation and utilization of multiple memories.

Methodology

We first provide an overview of the motivation of biological systems before formally introducing the different components of the proposed approach.

Continual Learning in the Biological System

Effective CL in the brain is facilitated by a complex set of mechanisms and multiple memory systems. Information in the brain is represented by neural activation patterns, which form a neural code (Foldiak and Endres 2008). Specifically, evidence suggests that the brain employs *Sparse Coding,* in which sensory events are represented by strong activations of a relatively small set of neurons. A different subset of neurons is utilized for each stimulus (Foldiak 2003; Barth and Poulet 2012). There is a correlation between these sparse codes (Lehky, Tanaka, and Sereno 2021) that could capture the similarity between different stimuli. Sparse codes provide several advantages: they enable faster learning of new associations with controlled interference with previous associations and allow efficient maintenance of associative memory while retaining sufficient representational capacity.

Another salient feature of the brain is the strong differentiation and specialization of the nervous systems (Hadsell et al. 2020). There is evidence for modularity in biological systems, which supports functional specialization of brain regions (Kelkar and Medaglia 2018) and reduces interference between different tasks. Furthermore, the brain is believed to utilize multiple memory systems (Atkinson and Shiffrin 1968; McClelland, McNaughton, and O'Reilly 1995). Complementary learning systems (CLS) theory states that efficient learning requires at least two complementary systems. The instance-based hippocampal system rapidly encodes new episodic events into non-overlapping representations, which are then gradually consolidated into the structured knowledge representation in the parametric neocortical system. The consolidation of information is accompanied by replay of the neural activities.

The encoding of information into efficient sparse codes, the modular and dynamic processing of information, and the interplay of multiple memory systems might play a crucial role in enabling effective CL in the brain. Therefore, our method aims to incorporate these components into ANNs.

Sparse Coding in DNNs

The sparse neural codes in the brain are in stark contrast to the highly dense connections and overlapping representations in standard DNNs, which are prone to interference. Particularly for CL, sparse representations can reduce interference between tasks and therefore reduce forgetting, as there will be fewer task-sensitive parameters or fewer effective changes to the parameters (Abbasi et al. 2022; Iyer et al. 2021). Activation sparsity can also lead to the natural emergence of modules without explicit architectural constraints (Hadsell et al. 2020). Therefore, to mimic sparse coding, we enforce activation sparsity with a complementary semantic dropout mechanism which encourages the model to activate similar units for semantically similar samples.

Sparse Activations: To enforce the sparsity in activations, we use the k-winner-take-all (k-WTA) activation function (Maass 2000). k-WTA only retains the top-k largest values of an $N \times 1$ input vector and sets all the others to zero before propagating the vector to the next network layer. Importantly, we deviate from the common implementation of k-WTA in convolutional neural networks (CNNs) whereby the activation map of a layer ($C \times H \times W$ tensor where C is the number of channels and H and W are the spatial dimensions) is flattened into a long $CHW \times 1$ vector input and the k-WTA activation is applied similar to the fully connected network (Xiao, Zhong, and Zheng 2019; Ahmad and Scheinkman 2019). We believe that this implementation does not take into account the functional integrity of an individual convolution filter as an independent feature extractor and does not lend itself to the formation of task-specific subnetworks with specialized feature extractors. Instead, we assign an activation score to each filter in the layer by taking the absolute sum of the corresponding activation map and select the top-k filters to propagate to the next layer.

Given the activation map, we flatten the last two dimensions and assign a score to each filter by taking the absolute sum of the activations. Based on the sparsity ratio for each layer, the activation maps of the filters with higher scores are propagated to the next layers, and the others are set to zero. This enforces global sparsity, whereby each stimulus is processed by only a selected set of filters in each layer, which can be considered as a subnetwork. We also consider each layer's role when setting the sparsity ratio. The earlier layers have a lower sparsity ratio as they learn general features, enabling higher reusability, and forward transfer to subsequent tasks uses a higher sparsity for later layers to reduce the interference between task-specific features.

Semantic Dropout: While the k-WTA activation function enforces the sparsity of activation for each stimulus, it does not encourage semantically similar inputs to have similar activation patterns and reduce overlap with semantically dissimilar inputs. To this end, we employ a complementary *Semantic Dropout* mechanism, which controls the degree of overlap between activations of samples belonging to different tasks while also encouraging same class samples to utilize a similar set of units. We utilize two sets of activation trackers: *global activity counter*, $\mathcal{A}_g \in \mathbb{R}^N$, counts the number of times each unit has been activated throughout training, whereas *class-wise activity counter*, $\mathcal{A}_s \in \mathbb{R}^{C \times N}$, tracks the number of times each unit has been active for samples belonging to a particular class. N and C denote the total number of units and classes, respectively. For each subsequent task, we first employ Heterogeneous Dropout (Abbasi et al. 2022) to encourage the model to learn the new classes by using neurons that have been less active for previously seen classes by setting the probability of a neuron being dropped to be inversely proportional to its activation counts. Concretely, let $[\mathcal{A}_g^l]_j$ denote the number of times that the unit j in layer l has been activated after learning t sequential tasks. For learning the new classes in task t+1, the probability of retaining this unit is given by:

$$[P_h^l]_j = exp(\frac{-[\mathcal{A}_g^l]_j}{\max_i [\mathcal{A}_g^l]_i} \pi_h) \tag{1}$$

where π_h controls the strength of dropout with larger values leading to less overlap between representations. We then allow the network to learn with the new task with heterogeneous dropout in place of a fixed number of epochs, $\mathcal{E}_h$.

Algorithm 1: SCoMMER Algorithm for Sparse Coding in Multiple Memory Experience Replay System

Input: data stream $\mathcal{D}$; learning rate η; consistency weight γ; update rate r and decay parameter α, dropout rates π_h and π_s
Initialize: $\theta_s = \theta_w$
$\mathcal{M} \leftarrow \{\}$
for $\mathcal{D}_t \in \mathcal{D}$ **do**
  while Training **do**
    Sample training data: $(x_t, y_t) \sim \mathcal{D}_t$ and $(x_m, y_m) \sim \mathcal{M}$, and interleave $x \leftarrow (x_t, x_m)$
    Retrieve structural knowledge: $\mathcal{Z}_s \leftarrow f(x_m; \theta_s)$
    Evaluate overall loss loss: $\mathcal{L} = \mathcal{L}_{ce}(f(x; \theta_w), y) + \gamma \mathcal{L}_{kr}(f(x_m; \theta_w), \mathcal{Z}_s)$ (Eq. 4)
    Update working memory: $\theta_w \leftarrow \theta_w - \eta \nabla_{\theta_w} \mathcal{L}$
    Aggregate knowledge: $\theta_s \leftarrow \alpha \theta_s + (1 - \alpha)\, \theta_w, \quad if \;\; r > a \sim U(0, 1)$ (Eq. 3)
    Update episodic memory: $\mathcal{M} \leftarrow$ Reservoir$(\mathcal{M}, (x_t, y_t))$
    After $\mathcal{E}_h$ epochs, update semantic dropout probabilities at the end of each epoch: P_s (Eq. 2)
  Update heterogeneous dropout probabilities: P_h (Eq. 1)
return θ_s

During this period, we let the class-wise activations emerge and then employ *Semantic Dropout*. It encourages the model to utilize the same set of units by setting the probability of retention of a unit for each class c as proportional to the number of times it has been activated for that class so far:

$$[P_s^l]_{c,j} = 1 - exp(\frac{-[\mathcal{A}_s^l]_{c,j}}{\max_i\,[\mathcal{A}_s^l]_{c,i}} \pi_s) \tag{2}$$

where π_s controls the strength of dropout. The probabilities for semantic dropout are updated at the end of each epoch to enforce the emerging pattern. This provides us with an efficient mechanism for controlling the degree of overlap in representations as well as enabling context-specific processing of information, which facilitates the formation of semantically conditioned subnetworks. Activation sparsity, together with semantic dropout, also provides an efficient mechanism for balancing the reusability and interference of features depending on the similarity of classes across the tasks.

Multiple Memory Systems

Inspired by the interaction of multiple memory systems in the brain, in addition to a fixed-size instance-based episodic memory, our method builds a long-term memory that aggregates the learned information in the working memory.

Episodic Memory: Information consolidation in the brain is facilitated by replaying the neural activation patterns that accompanied the learning event. To mimic this mechanism, we employ a fixed-size episodic memory buffer, which can be thought of as a very primitive hippocampus. The memory buffer is maintained with *Reservoir Sampling* (Vitter 1985), which aims to match the distribution of the data stream by assigning an equal probability to each incoming sample.

Long-Term Memory: We aim to build a long-term semantic memory that can consolidate and accumulate the structural knowledge learned in the working memory throughout the training trajectory. The knowledge acquired in DNNs resides in the learned synaptic weights (Krishnan et al. 2019). Hence, progressively aggregating the weights of the working memory (θ_w) as it sequentially learns tasks allows us to consolidate the information efficiently. To this end, we build long-term memory (θ_s) by taking the exponential moving average of the working memory weights in a stochastic manner (which is more biologically plausible (Arani, Sarfraz, and Zonooz 2021)), similar to (Arani, Sarfraz, and Zonooz 2022):

$$\theta_s \leftarrow \alpha \theta_s + (1 - \alpha)\, \theta_w, \quad if \;\; r > a \sim U(0, 1) \tag{3}$$

where α is the decay parameter and r is the update rate.

Long-term memory builds structural representations for generalization and mimics the slow acquisition of structured knowledge in the neocortex, which can generalize well across tasks. Long-term memory then interacts with instance-level episodic memory to retrieve structural relational knowledge (Sarfraz, Arani, and Zonooz 2021) for the previous tasks encoded in the output logits. Consolidated logits are then utilized to enforce consistency in the functional space of the working model. This facilitates the consolidation of information by encouraging the acquisition of new knowledge while maintaining the functional relation of previous knowledge and aligning the decision boundary of working memory with long-term memory.

Overall Formulation

Given a continuous data stream $\mathcal{D}$ containing a sequence of tasks $(\mathcal{D}_1, \mathcal{D}_2, .., \mathcal{D}_T)$, the CL task is to learn the joint distribution of all the observed tasks without the availability of task labels at test time. Our proposed method, SCoMMER, involves training a working memory θ_w, and maintains an additional long-term memory θ_s and an episodic memory $\mathcal{M}$. The long-term memory is initialized with the same parameters as the working memory and has the same sparsity constraints. Therefore, long-term memory aggregates the weights of working memory. We initialize heterogeneous dropout probabilities π_h randomly to set the probability of retention of a fraction of units to 1 and others to 0 so that the first task is learned using a few, but sufficient units and the remaining can be utilized to learn subsequent tasks.

During each training step, we interleave the batch of samples from the current task $x_t \sim \mathcal{D}_t$, with a random batch of exemplars from episodic memory $x_m \sim \mathcal{M}$. Working memory is trained with a combination of cross-entropy loss in

Buffer	Method	S-CIFAR10		S-CIFAR100		GCIL	
		Class-IL	Task-IL	Class-IL	Task-IL	Unif	Longtail
–	JOINT	92.20±0.15	98.31±0.12	70.62±0.64	86.19±0.43	58.36±1.02	56.94±1.56
	SGD	19.62±0.05	61.02±3.33	17.58±0.04	40.46±0.99	12.67±0.24	22.88±0.53
200	ER	44.79±1.86	91.19±0.94	21.40±0.22	61.36±0.39	16.40±0.37	19.27±0.77
	DER++	64.88±1.17	91.92±0.60	29.60±1.14	62.49±0.78	18.84±0.60	26.94±1.27
	CLS-ER	66.19±0.75	**93.90**±0.60	35.23±0.86	67.34±0.79	25.06±0.81	28.54±0.87
	SCoMMER	**69.19**±0.61	93.20±0.10	**40.25**±0.05	**69.39**±0.43	**30.84**±0.80	**29.08**±0.31
500	ER	57.74±0.27	93.61±0.27	28.02±0.31	68.23±0.16	28.21±0.69	20.30±0.63
	DER++	72.70±1.36	93.88±0.50	41.40±0.96	70.61±0.11	32.92±0.74	25.82±0.83
	CLS-ER	**75.22**±0.71	**94.94**±0.53	47.63±0.61	73.78±0.86	36.34±0.59	28.63±0.68
	SCoMMER	74.97±1.05	94.36±0.06	**49.63**±1.43	**75.49**±0.43	**36.87**±0.36	**35.20**±0.21

Table 1: Comparison on different CL settings. The baseline results are from (Arani, Sarfraz, and Zonooz 2022).

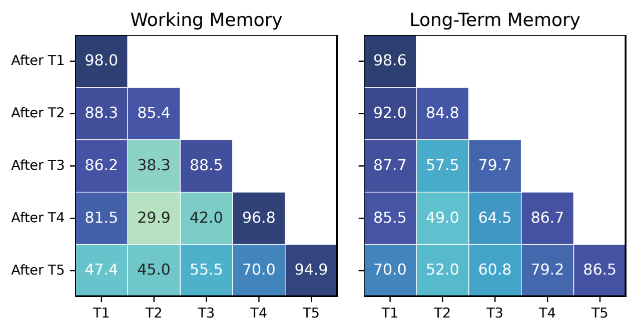

Figure 2: Task-wise performance of working memory and the long-term memory. The long-term memory effectively aggregates knowledge and generalizes well across the tasks.

the interleaved batch $x \leftarrow (x_t, x_b)$, and knowledge retrieval loss on the exemplars. Thus, the overall loss is given by:

$$\mathcal{L} = \mathcal{L}_{ce}(f(x;\theta_w), y) + \gamma \mathcal{L}_{kr}(f(x_m;\theta_w), f(x_m;\theta_s)) \quad (4)$$

where γ controls the strength of consistency enforcement, and the mean squared error loss is used for $\mathcal{L}_{kr}$. The training step is followed by stochastically updating the long-term memory (Eq. 3). The semantic dropout and heterogeneous dropout probabilities are updated at the end of each epoch and task, respectively (using Eqs. 1 and 3). We use long-term memory for inference, as it aggregates knowledge and generalizes well across tasks (cf. Figure 2). Algorithm 1 provides further training details.

Evaluation Protocol

To gauge the effectiveness of SCoMMER in tackling the challenges faced by a lifelong learning agent, we consider multiple CL settings that test different aspects of the model.

Class-IL presents a challenging CL scenario where each task presents a new set of disjoint classes, and the model must learn to distinguish between all the classes seen so far without the availability of task labels at the test time. It requires the model to effectively consolidate information across tasks and learn generalizable features that can be reused to acquire new knowledge. **Generalized Class-IL (GCIL)** (Mi et al. 2020) extends the Class-IL setting to more realistic scenarios where the agent has to learn an object over multiple recurrences spread across tasks and tackle the challenges of class imbalance and varying number of classes in each task. GCIL utilizes probabilistic modeling to sample the number of classes, the classes appearing, and their sample sizes. Details of the datasets used in each setting are provided in Appendix. Although our method does not utilize separate classification heads or subnets, for completion, we also evaluate performance under the Task-IL setting, where the model has access to the task labels at inference. In this setting, we use the task label to select the subset of output logits to select from.

Empirical Evaluation

We compare SCoMMER with state-of-the-art rehearsal-based methods across different CL settings under uniform experimental settings (details provided in Appendix). *SGD* provides the lower bound with standard training on sequential tasks, and *JOINT* gives the upper bound on performance when the model is trained on the joint distribution.

Table 1 shows that SCoMMER provides performance gains in the majority of the cases and demonstrates the effectiveness of our approach under varying challenging CL settings. In particular, it provides considerable improvement under low buffer size settings, which suggests that our method is able to mitigate forgetting with fewer samples from previous tasks. The performance gains over CLS-ER, which employs two semantic memories, show that the sparse coding in our method enables the effective utilization of a single semantic memory. In particular, the gains in the GCIL setting where the agent has to face the challenges of class imbalance and learn over multiple occurrences of objects allude to several advantages of our method. Our proposed semantic dropout in conjunction with sparse activations enables the model to reuse the sparse code associated with the recurring object and learn better representations with the additional samples by adapting the corresponding

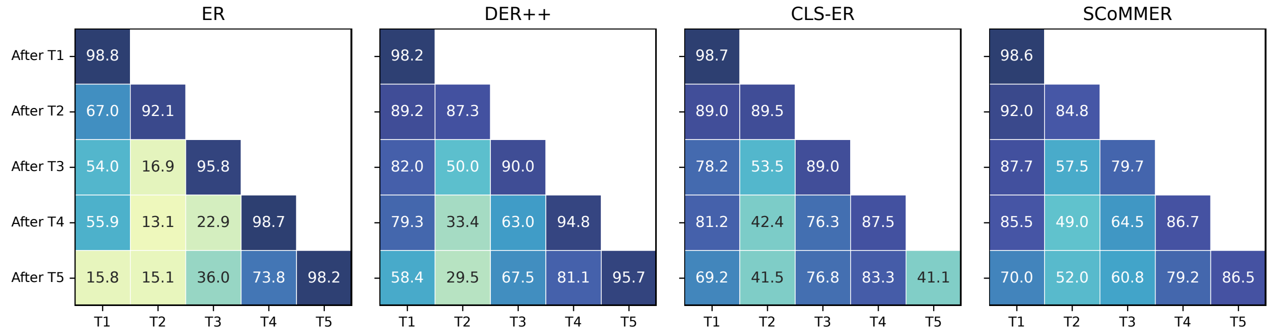

Figure 3: Task-wise performance of different methods. The heatmaps provide the test set accuracy of each task (x-axis) evaluated at the end of each sequential learning task (y-axis). SCoMMER retains the performance of earlier tasks better without compromising on the current task.

Sparse Activations	Long-Term Memory	Semantic Dropout	Accuracy
✓	✓	✓	**69.19**$_{\pm 0.61}$
✓	✓	✗	67.38$_{\pm 1.51}$
✗	✓	✗	61.88$_{\pm 2.43}$
✓	✗	✗	49.44$_{\pm 5.43}$
✗	✗	✗	44.79$_{\pm 1.86}$

Table 2: Ablation Study: Effect of systematically removing different components of SCoMMER on the performance in S-CIFAR10. All components contribute to the gain.

subset of filters. Furthermore, compared to the dense activations in CLS-ER, the sparse coding in SCoMMER leads to the emergence of subnetworks that provide modularity and protection to other parts of the network, since the entire network is not updated for each input image. This increases the robustness of the model to class imbalance.

Overall, our method provides an effective approach to employ sparse coding in DNN and allows better long-term memory utilization, which can effectively consolidate information across tasks and further mitigate forgetting.

Ablation Study

To gain further insight into the contribution of each component of our method, we systematically remove them and evaluate the performance of the model in Table 2. The results show that all components of SCoMMER contribute to the performance gains. The drop in performance from removing semantic dropout suggests that it is effective in enforcing sparse coding on the representations of the model, which reduces the interference between tasks and allows semantically similar classes to share information. We also observe the benefits of multiple memory systems in CL. Additional long-term memory provides considerable performance improvement and suggests that the EMA of the learned synaptic weights can effectively consolidate knowledge across tasks. Furthermore, we observe that sparsity is a critical component for enabling CL in DNNs. Sparse activations alone significantly improve ER performance and also enable efficient utilization of semantic memory. We highlight that these individual components complement each other and that the combined effect leads to the observed performance improvement in our method.

Characteristics Analysis

We look at the model characteristics to understand the performance gains. Subsequent analysis is performed on models trained on S-CIFAR10 with a 200 buffer size.

Stability-Plasticity Dilemma

To better understand how well different methods maintain a balance between stability and plasticity, we look at how task-wise performance evolves as the model learns tasks sequentially. The diagonal of the heatmap shows the plasticity of the model as it learns the new task, whereas the difference between the accuracy of the task when it was first learned and at the end of the training indicates the stability of the model. Figure 3 shows that SCoMMER is able to maintain a better balance and provides a more uniform performance on tasks compared to baselines. While CLS-ER provides better stability than DER++, it comes at the cost of the model's performance on the last task, which could be due to the lower update rate of the stable model. SCoMMER, on the other hand, retains performance on the earlier tasks (T1 and T2) and provides good performance on the recent task. We also compare the long-term semantic and working memory performance in Figure 2. Long-term memory effectively aggregates the learned knowledge into the synaptic weights of working memory and generalizes well across tasks.

Emergence of Subnetworks

To evaluate the effectiveness of activation sparsity and semantic dropout to enforce sparse coding in the model, we look at the average activity of the units in the penultimate layer. The emerging sparse code for each class is tracked during training using the class-wise activity counter and enforced using semantic dropout probabilities (Equation 2). Given a test sample from class c, ideally, we would want

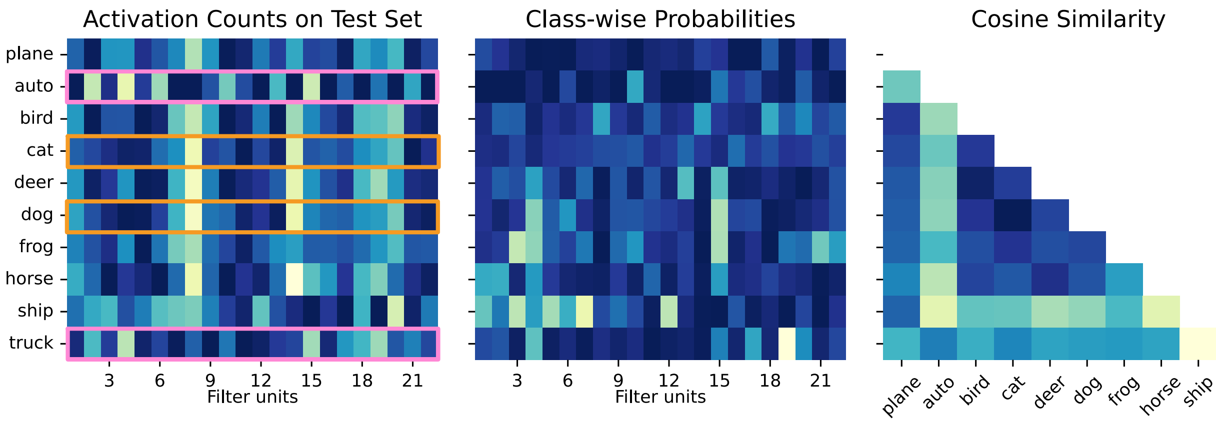

Figure 4: Class-wise activation counts of the filters in the penultimate layer of the model trained on S-CIFAR10 with 200 buffer size. Comparison of the activation counts on the test set with the learned class-wise probabilities, P_s, during training shows the effectiveness of semantic dropout in enforcing sparse coding. Right plot shows the cosine similarities between the activation counts of different classes. Semantically similar classes have higher correlation in activations. Darker color shows higher values.

the model to use the subset of neurons that had higher activity for the training samples from class c without providing any task information. Concretely, we track the class-wise activity on the test set and plot the normalized activation counts for a set of neurons next to their class-wise probabilities at the end of training. Figure 4 shows a high correlation between the test set activation counts and the semantic dropout probabilities at the end of training, particularly for recent classes. The activation counts also hint at the natural emergence of semantically conditioned subnetworks, as the model utilizes a different set of units for different classes. Furthermore, we observe that semantically similar classes have a higher degree of correlation between their activation patterns. For instance, cat and dog share the most active neurons, a similar pattern is observed between horse and deer, and car and truck. The cosine similarities between the activation counts of the different classes further supports the observation. This is even more remarkable given that these classes are observed in different tasks, particularly for cars and trucks, which are observed in the first and last tasks.

Task Recency Bias

A major challenge in CL is the recency bias, in which the predictions of the model are biased toward the recent task task (Wu et al. 2019). This leads to considerable forgetting of earlier tasks. To compare the degree to which SCoMMER tackles this issue, we evaluate the probabilities of predicting each task by aggregating the softmax output of samples from the test set of all seen tasks and averaging the probabilities of classes in each task. Figure 5 shows that SCoMMER provides more uniform probabilities to predict each task. CLS-ER is able to mitigate the bias towards the last task, which can be attributed to the aggregation of knowledge in the semantic memories; however, CLS-ER reduces the probability of predicting the last task. SCoMMER effectively mitigates recency bias and provides uniform prediction probabilities across tasks without any explicit regularization.

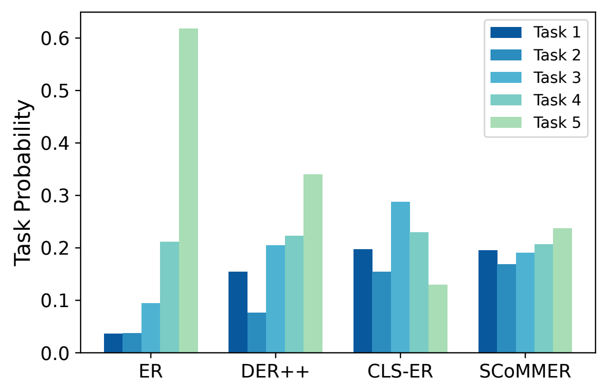

Figure 5: Average probabilities of predicting classes from each tasks at the end of training. SCoMMER provides more uniform probabilities across the tasks.

Conclusion

Motivated by the mechanisms for the representation and utilization of multiple memory systems in the brain, we proposed a novel approach to employ sparse coding in multiple memory systems. SCoMMER enforces activation sparsity along with a complementary semantic dropout mechanism, which encourages the model to activate similar units for semantically similar inputs and reduce overlap with dissimilar inputs. In addition, it maintains long-term memory, which consolidates the learned knowledge in the working memory. Our empirical evaluation shows the effectiveness of the approach in mitigating forgetting in challenging CL scenarios. Furthermore, sparse coding enables efficient consolidation of knowledge in the long-term memory, reduces the bias towards recent tasks, and leads to the emergence of semantically conditioned subnetworks. We hope that our study inspires further research in this promising direction.

References

Abbasi, A.; Nooralinejad, P.; Braverman, V.; Pirsiavash, H.; and Kolouri, S. 2022. Sparsity and Heterogeneous Dropout for Continual Learning in the Null Space of Neural Activations. *arXiv preprint arXiv:2203.06514*.

Ahmad, S.; and Scheinkman, L. 2019. How can we be so dense? The benefits of using highly sparse representations. *arXiv preprint arXiv:1903.11257*.

Aljundi, R.; Belilovsky, E.; Tuytelaars, T.; Charlin, L.; Caccia, M.; Lin, M.; and Page-Caccia, L. 2019a. Online continual learning with maximal interfered retrieval. In *Advances in Neural Information Processing Systems*, 11849–11860.

Aljundi, R.; Lin, M.; Goujaud, B.; and Bengio, Y. 2019b. Gradient based sample selection for online continual learning. In *Advances in Neural Information Processing Systems*, 11816–11825.

Arani, E.; Sarfraz, F.; and Zonooz, B. 2021. Noise as a resource for learning in knowledge distillation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 3129–3138.

Arani, E.; Sarfraz, F.; and Zonooz, B. 2022. Learning Fast, Learning Slow: A General Continual Learning Method based on Complementary Learning System. In *International Conference on Learning Representations*.

Atkinson, R. C.; and Shiffrin, R. M. 1968. Human memory: A proposed system and its control processes. In *Psychology of learning and motivation*, volume 2, 89–195. Elsevier.

Barth, A. L.; and Poulet, J. F. 2012. Experimental evidence for sparse firing in the neocortex. *Trends in neurosciences*, 35(6): 345–355.

Bhat, P.; Zonooz, B.; and Arani, E. 2022. Consistency is the key to further mitigating catastrophic forgetting in continual learning. *arXiv preprint arXiv:2207.04998*.

Ebrahimi, S.; Petryk, S.; Gokul, A.; Gan, W.; Gonzalez, J. E.; Rohrbach, M.; et al. 2020. Remembering for the Right Reasons: Explanations Reduce Catastrophic Forgetting. In *International Conference on Learning Representations*.

Farajtabar, M.; Azizan, N.; Mott, A.; and Li, A. 2020. Orthogonal gradient descent for continual learning. In *International Conference on Artificial Intelligence and Statistics*, 3762–3773. PMLR.

Farquhar, S.; and Gal, Y. 2018. Towards robust evaluations of continual learning. *arXiv preprint arXiv:1805.09733*.

Foldiak, P. 2003. Sparse coding in the primate cortex. *The handbook of brain theory and neural networks*.

Foldiak, P.; and Endres, D. 2008. Sparse coding. *Scholarpedia*, 3(1): 2984.

Hadsell, R.; Rao, D.; Rusu, A. A.; and Pascanu, R. 2020. Embracing change: Continual learning in deep neural networks. *Trends in cognitive sciences*, 24(12): 1028–1040.

Hassabis, D.; Kumaran, D.; Summerfield, C.; and Botvinick, M. 2017. Neuroscience-inspired artificial intelligence. *Neuron*, 95(2): 245–258.

Isele, D.; and Cosgun, A. 2018. Selective experience replay for lifelong learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Iyer, A.; Grewal, K.; Velu, A.; Souza, L. O.; Forest, J.; and Ahmad, S. 2021. Avoiding Catastrophe: Active Dendrites Enable Multi-Task Learning in Dynamic Environments. *arXiv preprint arXiv:2201.00042*.

Kelkar, A.; and Medaglia, J. 2018. Evidence of brain modularity. *Encyclopedia of Evolutionary Psychological Science. Springer, Cham. https://doi. org/10.1007/978-3-319-16999-6_2422-1*.

Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N.; Veness, J.; Desjardins, G.; Rusu, A. A.; Milan, K.; Quan, J.; Ramalho, T.; Grabska-Barwinska, A.; et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13): 3521–3526.

Krishnan, G. P.; Tadros, T.; Ramyaa, R.; and Bazhenov, M. 2019. Biologically inspired sleep algorithm for artificial neural networks. *arXiv preprint arXiv:1908.02240*.

Lehky, S. R.; Tanaka, K.; and Sereno, A. B. 2021. Pseudosparse neural coding in the visual system of primates. *Communications biology*, 4(1): 1–12.

Li, Z.; and Hoiem, D. 2017. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12): 2935–2947.

Maass, W. 2000. On the computational power of winner-take-all. *Neural computation*, 12(11): 2519–2535.

McClelland, J. L.; McNaughton, B. L.; and O'Reilly, R. C. 1995. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 102(3): 419.

McCloskey, M.; and Cohen, N. J. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, 109–165. Elsevier.

Mi, F.; Kong, L.; Lin, T.; Yu, K.; and Faltings, B. 2020. Generalized Class Incremental Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 240–241.

Parisi, G. I.; Kemker, R.; Part, J. L.; Kanan, C.; and Wermter, S. 2019. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113: 54–71.

Pham, Q.; Liu, C.; and Hoi, S. 2021. Dualnet: Continual learning, fast and slow. *Advances in Neural Information Processing Systems*, 34: 16131–16144.

Rannen, A.; Aljundi, R.; Blaschko, M. B.; and Tuytelaars, T. 2017. Encoder based lifelong learning. In *Proceedings of the IEEE International Conference on Computer Vision*, 1320–1328.

Riemer, M.; Cases, I.; Ajemian, R.; Liu, M.; Rish, I.; Tu, Y.; and Tesauro, G. 2018. Learning to learn without forgetting by maximizing transfer and minimizing interference. *arXiv preprint arXiv:1810.11910*.

Ritter, H.; Botev, A.; and Barber, D. 2018. Online structured laplace approximations for overcoming catastrophic forgetting. In *Advances in Neural Information Processing Systems*, 3738–3748.

Sarfraz, F.; Arani, E.; and Zonooz, B. 2021. Knowledge distillation beyond model compression. In *2020 25th International Conference on Pattern Recognition (ICPR)*, 6136–6143. IEEE.

Vitter, J. S. 1985. Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)*, 11(1): 37–57.

Wang, Z.; Zhang, Z.; Ebrahimi, S.; Sun, R.; Zhang, H.; Lee, C.-Y.; Ren, X.; Su, G.; Perot, V.; Dy, J.; et al. 2022a. DualPrompt: Complementary Prompting for Rehearsal-free Continual Learning. *arXiv preprint arXiv:2204.04799*.

Wang, Z.; Zhang, Z.; Lee, C.-Y.; Zhang, H.; Sun, R.; Ren, X.; Su, G.; Perot, V.; Dy, J.; and Pfister, T. 2022b. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 139–149.

Wu, Y.; Chen, Y.; Wang, L.; Ye, Y.; Liu, Z.; Guo, Y.; and Fu, Y. 2019. Large scale incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 374–382.

Xiao, C.; Zhong, P.; and Zheng, C. 2019. Enhancing adversarial defense by k-winners-take-all. *arXiv preprint arXiv:1905.10510*.

Zenke, F.; Poole, B.; and Ganguli, S. 2017. Continual learning through synaptic intelligence. *Proceedings of machine learning research*, 70: 3987.