Learning with Partial Forgetting in Modern Hopfield Networks

Toshihiro Ota
Tokyo Institute of Technology

Ikuro Sato
Tokyo Institute of Technology
Denso IT Lab, Inc.

Rei Kawakami
Tokyo Institute of Technology

Masayuki Tanaka
Tokyo Institute of Technology

Nakamasa Inoue
Tokyo Institute of Technology

Abstract

It has been known by neuroscience studies that partial and transient forgetting of memory often plays an important role in the brain to improve performance for certain intellectual activities. In machine learning, associative memory models such as classical and modern Hopfield networks have been proposed to express memories as attractors in the feature space of a closed recurrent network. In this work, we propose learning with partial forgetting (LwPF), where a partial forgetting functionality is designed by element-wise non-bijective projections, for memory neurons in modern Hopfield networks to improve model performance. We incorporate LwPF into the attention mechanism also, whose process has been shown to be identical to the update rule of a certain modern Hopfield network, by modifying the corresponding Lagrangian. We evaluated the effectiveness of LwPF on three diverse tasks such as bit-pattern classification, immune repertoire classification for computational biology, and image classification for computer vision, and confirmed that LwPF consistently improves the performance of existing neural networks including DeepRC and vision transformers.

1 Introduction

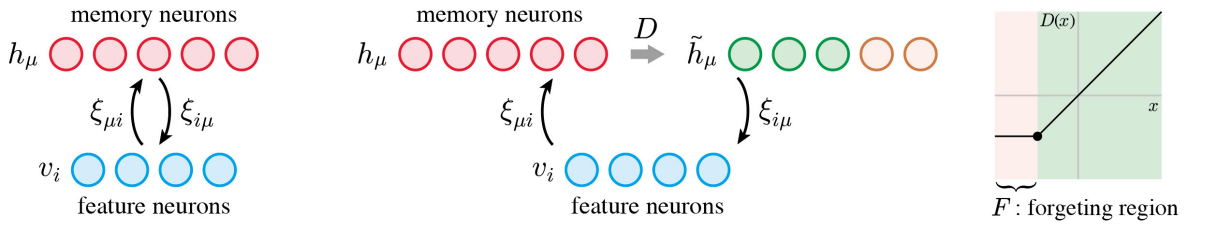
Hopfield networks (Hopfield, 1982, 1984) have been proposed to model associative memories as attractors in the feature space of a given closed neural network with an Ising-type energy function. In practice, however, this classical Hopfield network suffers a limitation of memory capacity, in which the number of distinct memories is at most proportional to the dimension of the feature space. To address this issue, models with significantly increased memory capacities (*e.g.*, exponential memory storage with respect to the feature dimension) have been recently proposed (Krotov and Hopfield, 2016; Demircigil et al., 2017; Krotov and Hopfield, 2018; Barra et al., 2018; Agliari and De Marzo, 2020), though there is a criticism about having many-body interactions which is absent in the brain (Krotov and Hopfield, 2021).

Nearly at the same time that these modern Hopfield networks were proposed, Ramsauer *et al*. clarified that each of the attention modules in the transformer (Vaswani et al., 2017) is essentially equivalent to the process of the update rule of a modern continuous Hopfield network (Ramsauer et al., 2021), and their algorithm has been applied to various tasks (Widrich et al., 2020; Fürst et al., 2021; Widrich et al., 2021; Schäfl et al., 2021; Salvatori et al., 2021; Salem, 2021; Seidl et al., 2022; Millidge et al., 2022). Based on their method and earlier works, Krotov and Hopfield developed a more general associative memory model (Krotov and Hopfield, 2021), which we refer to as the large associative memory model. In this model, two-body interactions between feature and memory neurons controlled by the corresponding pair of Lagrangian functions can express some of existing associative memory models with and without exponential memory capacity. As the model of Ramsauer *et al*. can be reproduced by choosing a certain pair of Lagrangians (Krotov and Hopfield, 2021), there is still a room to generate new models that possess preferable properties in addition to large memory capacity by designing new Lagrangians.

In neuroscience, it has been known that associative memo-

Proceedings of the 26th International Conference on Artificial Intelligence and Statistics (AISTATS) 2023, Valencia, Spain. PMLR: Volume 206.

The first author is currently with CyberAgent, Inc.



(a) Clean interaction (b) Partially degraded interaction (c) Partial forgetting function

Figure 1: Memory neurons h_μ and feature neurons v_i. (a) Clean interaction $\xi_{\mu i}$ between h_μ and v_i in the large associative memory model (Krotov and Hopfield, 2021). (b) Proposed partially degraded interaction where some memory neurons are degraded. (c) Partial forgetting function D for degrading memory neurons. D is characterized by forgetting region F, where memory neurons forget what the output should be. An example of $D : \mathbb{R} \to \mathbb{R}$ is visualized.

ries are encoded in sets of neurons and their interconnections (Miry et al., 2021; Josselyn and Tonegawa, 2020; Poo et al., 2016). In addition, the dynamics and stability of memory in human brains have been modeled by attractor dynamics of networks of the neurons (Pereira and Brunel, 2018; Wu et al., 2018; Seeholzer et al., 2019; Spalla et al., 2021). On the other hand, the brain experiences *forgetting*; *i.e.*, partial information losses occur to memories. Forgetting might be just seen as a process to somehow spoil memories at first glance, but recent studies indicate that partial and transient forgetting plays a crucial role in the brain to improve performance for certain intellectual activities (Sabandal et al., 2021; Hirashima and Nozaki, 2012). Such findings from neuroscience motivate us to model partial forgetting functionality with modern Hopfield networks and examine how the performance of a model trained with partial forgetting changes. Interestingly, we experimentally confirmed that models with the proposed partial forgetting functionality consistently improve the performance of existing neural network models.

Our contributions are three-fold:

1. We propose learning with partial forgetting (LwPF) for modern Hopfield networks by introducing a *partial forgetting function* to the Lagrangian to partially eliminate information from memory neurons.

2. We modify the attention mechanism with the proposed partial forgetting functionality. From the large associative memory model (Krotov and Hopfield, 2021), we derive the expression for *partially forgetting attention*, which only slightly modifies the argument of the softmax with the partial forgetting function.

3. We demonstrate the effectiveness of LwPF on three diverse tasks, namely bit-pattern classification, immune repertoire classification for computational biology, and image classification for computer vision. We show that LwPF improves the performance of existing models, including a modern Hopfield network DeepRC (Widrich et al., 2020) and vision transformers (Dosovitskiy et al., 2021; Touvron et al., 2021).

2 Background

To fix notations, we here provide a derivation of the attention mechanism in transformers from a continuous Hopfield network. The update rule of the neurons and the corresponding energy function will coincide with the modern continuous Hopfield network discussed by Ramsauer et al. (2021), in which the modern Hopfield network is identified with the attention modules in transformers. We will utilize this fact in Sec. 3.2.

2.1 Overview of Large Associative Memory Model

Let us first briefly review the large associative memory model proposed by Krotov and Hopfield (2021). In this system, dynamical variables are composed of N_v visible feature neurons and N_h hidden memory neurons both continuous,

$$v : U(\subset \mathbb{R}) \to \mathbb{R}^{N_v}, \tag{1}$$

$$h : U(\subset \mathbb{R}) \to \mathbb{R}^{N_h}. \tag{2}$$

The interaction matrix

$$\xi \in \mathbb{R}^{N_h \times N_v} \tag{3}$$

governs the way that visible and hidden neurons change over time as illustrated in Fig. 1a. With the relaxing time constants of the two groups of neurons τ_v and τ_h, the model is described by the following differential equations,

$$\tau_v \frac{dv_i(t)}{dt} = \sum_{\mu=1}^{N_h} \xi_{i\mu} f_\mu(h(t)) - v_i(t), \tag{4}$$

$$\tau_h \frac{dh_\mu(t)}{dt} = \sum_{i=1}^{N_v} \xi_{\mu i} g_i(v(t)) - h_\mu(t), \tag{5}$$

where the argument t is thought of as "time", and the activation functions f and g are determined by Lagrangians

$L_h : \mathbb{R}^{N_h} \to \mathbb{R}$ and $L_v : \mathbb{R}^{N_v} \to \mathbb{R}$, such that

$$f_\mu(h) = \frac{\partial L_h(h)}{\partial h_\mu}, \quad g_i(v) = \frac{\partial L_v(v)}{\partial v_i}. \tag{6}$$

The canonical energy function for this system is given as

$$E(v,h) = \sum_{i=1}^{N_v} v_i g_i(v) - L_v(v) + \sum_{\mu=1}^{N_h} h_\mu f_\mu(h) - L_h(h) - \sum_{\mu,i} f_\mu \xi_{\mu i} g_i. \tag{7}$$

One can easily confirm that this energy function monotonically decreases, *i.e.*,

$$\frac{dE(v(t),h(t))}{dt} \leq 0, \tag{8}$$

along the trajectory of the dynamical equations, provided that the Hessians of the Lagrangians are positive semi-definite.

In addition to this, if the overall energy function is bounded from below, the dynamical equations are guaranteed to converge to a fixed point attractor state, which corresponds to one of the local minima of the energy function. The formulation of networks in terms of Lagrangians and an associated energy function enables us to easily experiment with different choices of the activation functions and different architectural arrangements of neurons.

2.2 Attention Mechanism

Suppose we have a fixed interaction matrix $\xi_{\mu i}$, then the system is defined by the choice of Lagrangians L_h and L_v. Krotov and Hopfield demonstrated that the specific choice of Lagrangians called "model B" in their paper (Krotov and Hopfield, 2021) reproduces the attention mechanism in transformers (Vaswani et al., 2017). This model is given by the following Lagrangians:

$$L_h(h) = \log \sum_\mu e^{h_\mu}, \quad L_v(v) = \frac{1}{2}\sum_i v_i^2. \tag{9}$$

For these Lagrangians, the activation functions are

$$f_\mu(h) = \frac{\partial L_h}{\partial h_\mu} = \frac{e^{h_\mu}}{\sum_\nu e^{h_\nu}} = \mathrm{softmax}(h_\mu), \tag{10}$$

$$g_i(v) = \frac{\partial L_v}{\partial v_i} = v_i. \tag{11}$$

Now we assume the adiabatic limit, $\tau_v \gg \tau_h$, which means that the dynamics of hidden memory neurons is much faster than that of visible feature neurons, *i.e.*, we can take $\tau_h \to 0$:

$$\text{Eq. (5)} \quad \rightsquigarrow \quad h_\mu(t) = \sum_{i=1}^{N_v} \xi_{\mu i} v_i(t). \tag{12}$$

Substituting the above expression into the other dynamical equation and discretizing it by taking $\Delta t = \tau_v$, then we obtain the update rule for feature neurons,

$$v_i(t+1) = \sum_{\mu=1}^{N_h} \xi_{i\mu}\, \mathrm{softmax}\left(\sum_{j=1}^{N_v} \xi_{\mu j} v_j(t)\right). \tag{13}$$

The energy function is also determined by the Lagrangians:

$$E = \frac{1}{2}\sum_{i=1}^{N_v} v_i^2 - \log\left(\sum_\mu \exp\left(\sum_i \xi_{\mu i} v_i\right)\right). \tag{14}$$

This update rule and the energy function coincide (up to some constants) with Ramsauer et al. (2021). Ramsauer *et al.* showed that a single update of v_i with the update rule in Eq. (13) is identical to the process of an attention module up to a linear projection when one regards v_j and $\xi_{\mu j}$ to be query and key matrices, respectively. That is, the linear transformations of a feature X in attention mechanism to query and key correspond to v and ξ in this paper. According to Ramsauer et al. (2021), another linear transformation of ξ can be identified with the value matrix V. More specifically, we can explicitly write down the identifications as follows. (Note that the notations are bit different from those by Ramsauer et al. (2021).) In Eq. (13), we have

$$\vec{v}(t) \in \mathbb{R}^{N_v}, \quad \xi = (\vec{\xi}_1, \ldots, \vec{\xi}_{N_h})^\top \in \mathbb{R}^{N_h \times N_v}. \tag{15}$$

Let us rewrite N_h to N and N_v to D. Suppose we have a set of N features $X := (\vec{x}_1, \ldots, \vec{x}_N)^\top \in \mathbb{R}^{N\times F}$ and for certain W_Q, $W_K \in \mathbb{R}^{F\times D}$ the linear projections give

$$\vec{v}_\nu = \vec{x}_\nu W_Q, \quad \vec{\xi}_\mu = \vec{x}_\mu W_K. \tag{16}$$

With $v := (\vec{v}_1, \ldots, \vec{v}_N)^\top \in \mathbb{R}^{N\times D}$ and another matrix $W'_V \in \mathbb{R}^{D\times D}$, we have an identification

$$Q := v = XW_Q, \tag{17}$$

$$K := \xi = XW_K, \tag{18}$$

$$V := \xi W'_V = XW_K W'_V. \tag{19}$$

By letting $W_V = W_K W'_V$, one obtains the formulation of conventional attention mechanism.

3 Learning with Partial Forgetting

In this section, we propose learning with partial forgetting (LwPF) that add a novel partial forgetting mechanism to the large associative memory model. In LwPF, the partial forgetting functionality is defined in a specific way so that one can control the amount of information carried by the memory neurons being lost along the interaction with the feature neurons.

Partial forgetting. Over the hidden memory neurons h_μ, we introduce *partial forgetting function* D and define *degraded memory neurons* $\tilde{h}_\mu$ as

$$\tilde{h}_\mu := D(h_\mu). \tag{20}$$

We designed D to be an element-wise function to model partial forgetting functionality for the memory neurons. The partial forgetting function D is identity everywhere except for *forgetting region* $F \subset \mathbb{R}$, where the value carried by the memory neuron cannot be reconstructed,

$$D(x; F, b) = \begin{cases} b & (x \in F) \\ x & (\text{otherwise}) \end{cases}, \tag{21}$$

where b is a constant bias. As this is a non-bijective function (as long as $F \neq \emptyset$), neurons cannot retrieve the value x fallen into the forgetting region F. An example of the forgetting function is shown in Fig. 1c. We will later give specific definitions of F and b, including a stochastic version, where F and b are randomly sampled from a p.d.f. depending on minibatch statistics at each training iteration, motivated by the partial and transient forgetting in the brain.

Model. Associated with the degradation of memory neurons (*i.e.*, partial forgetting function D), the Hopfield model of our interest (Fig. 1b) is then described by the following differential equations,

$$\tau_v \frac{dv_i(t)}{dt} = \sum_{\mu=1}^{N_h} \xi_{i\mu} \tilde{f}_\mu(h(t)) - v_i(t), \tag{22}$$

$$\tau_h \frac{dh_\mu(t)}{dt} = \sum_{i=1}^{N_v} \xi_{\mu i} g_i(v(t)) - h_\mu(t), \tag{23}$$

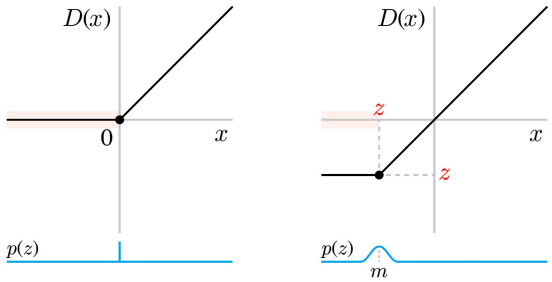
where the activation function g_i for the feature neurons is kept intact, and $\tilde{f}_\mu$ is our degraded activation function defined through Lagrangian L_h and $\tilde{h}$ as

$$\tilde{f}_\mu(h) = \frac{\partial L_h(\tilde{h})}{\partial h_\mu} = \frac{\partial L_h(D(h))}{\partial h_\mu}. \tag{24}$$

If there is no overlap between F and the values that h takes, then D practically becomes the identity function, and this model turns out to be equivalent to the large associative memory model reviewed in the previous section.

3.1 Characteristics of Degradation: Forgetting Regions

Our partial forgetting function is characterized by two components: forgetting region F and constant bias b. In the brain, forgetting would be likely a stochastic process in which information loss occurs with some probability, and we mimic such a property by introducing stochastic forgetting region, which is randomly defined iteration by iteration.



(a) ReLU (b) PFU

Figure 2: Partial forgetting functions. (a) ReLU as partial forgetting function, where forgetting region F is the negative real numbers $\mathbb{R}_{<0}$. (b) PFU, where z is randomly sampled from the Gaussian distribution $\mathcal{N}(m, \sigma^2)$.

In particular, we introduce specific ways of forgetting depending on different degrees of stochasticity as described below, and we will provide comparative experiments in the next section.

ReLU. With a fixed forgetting region $F = \{x : x < 0\}$ and the zero bias $b = 0$, the partial forgetting function turns out to be the ReLU function: $D(x; F, b) = \mathrm{ReLU}(x)$ (see Fig. 2a). This is one of the simplest examples of partial forgetting where memory neurons do not respond to negative inputs. The forgetting region F and the bias b is fixed here, which means that the partial forgetting occurs in a deterministic way for this function.

Partial Forgetting Unit (PFU). To incorporate randomness into partial forgetting, we propose PFU. It employs a forgetting region $F = \{x : x < z\}$ and a bias $b = z$ where z is a random variable of Gaussian distribution $\mathcal{N}(m, \sigma^2)$ (see Fig. 2b). The choice of m is based on the distribution of the features, that is, it is basically designed to take the median of the features, $m = \bar{x}_{\text{median}}$, where $\bar{x}_{\text{median}}$ is the median of inputs dynamically computed from a minibatch of a dataset. We left σ as a hyperparameter. In the testing phase, we set $z = m$ to make the inference deterministic. We empirically found that $m = \bar{x}_{\text{median}}$ performs well in some tasks (see Sec. 4.2), and $m = 0$ in some other tasks (see Sec. 4.3 for details) as a special case.

Let us make some remarks in order. As we will see, in our formulation the outputs of memory neurons in the forgetting region still supply positive values in the denominator of softmax in Eq. (25) when b is finite, so it may be better to take $b = -\infty$ as a model of artificial neural network. If we define $b = -\infty$, contributions to the denominator from neurons in the forgetting region completely vanish so that the $\chi(h_\mu)\mathrm{softmax}(D(h_\mu))$ is normalized to 1, which might look more meaningful. In this paper, however, we do not do so by the following two reasons. The first reason is that it becomes difficult to distinguish whether forgetting has occurred or not. If one takes $b = -\infty$, the output of the modified softmax is still normalized to one. Therefore, we

cannot distinguish whether the forgetting has occurred or just the softmax has quite small entries. The second reason is that if $b = -\infty$, the output gets to be indefinite when all the memory neurons falls into the forgetting region. With a finite b, the output is always well-defined, even when all the entries of an input happen to be forgotten. For such a case, the output will become zero vector, which clearly means the input is completely forgotten. Actually, this situation really occurs in Sec. 4.2.

To summarize, we model partial forgetting functionality by introducing an element-wise non-bijective function (partial forgetting function with forgetting region) as follows: When a memory neuron takes a value in the forgetting region, the degraded activation will replace the value with the cutoff value b. If memory neurons do not fall into the forgetting region, they are kept intact. Motivated by the fact that the brain experiences transient forgetting, we introduced randomness in PFU, aside from simple ReLU. When the forgetting region is fixed (deterministic) and set to be the entire negative values, the forgetting function turns out to be ReLU, and we propose PFU as an extension of ReLU to deal with stochastic behavior of forgetting functionality and distributions of features with non-zero median. Our model can be viewed as an extended version of the original large associative memory model (Krotov and Hopfield, 2021) because the symmetric relation of the interaction weight matrices will be effectively broken in our model. This extension seems reasonable in the sense that it incorporates not only memory retrieval but also forgetting functionality.

3.2 Partially Forgetting Attention

We now introduce a new attention mechanism with LwPF. As reviewed in Sec. 2.2, to obtain the attention mechanism from the large associative memory model, one needs to use the Lagrangians in Eq. (9). We modify the activation function for the degraded memory neuron $\tilde{h}$ as

$$\tilde{f}_\mu(h) = \frac{\partial L_h(\tilde{h})}{\partial h_\mu} = \chi(h_\mu)\mathrm{softmax}(D(h_\mu)), \tag{25}$$

where χ is the differential of D and given by the indicator function about $F^c = \mathbb{R} \setminus F$:

$$\chi(x) = \begin{cases} 0, & x \in F, \\ 1, & x \in F^c. \end{cases} \tag{26}$$

By this indicator function χ, the values of the hidden neurons fallen into the forgetting region are completely lost, *i.e.*, $\tilde{f}_\mu(h) = 0$ for $h_\mu \in F$.

We again assume the adiabatic limit, $\tau_v \gg \tau_h$, and integrate out the hidden memory neurons. Then, from Eq. (22) we have

$$\tau_v \frac{dv_i}{dt} = \sum_{\mu=1}^{N_h} \xi_{i\mu}\chi(h_\mu)\mathrm{softmax}(D(h_\mu)) - v_i, \tag{27}$$

with $h_\mu = \sum_k \xi_{\mu k} v_k$. The energy function of this system is given as:

$$E = \frac{1}{2}\sum_{i=1}^{N_v} v_i^2 - \log\left(\sum_\mu \exp\left(D\Big(\sum_i \xi_{\mu i} v_i\Big)\right)\right). \tag{28}$$

By defining an index set $M := \{\mu : \chi(h_\mu) = 1\}$, which picks up the undegraded memory neurons, Eq. (27) simplifies to

$$\tau_v \frac{dv_i}{dt} = \sum_{\mu \in M} \xi_{i\mu}\mathrm{softmax}\left(D\left(\sum_k \xi_{\mu k} v_k\right)\right) - v_i. \tag{29}$$

This means that all the interactions except for those in the forgetting region can be written in the almost same way as in the previous section, and now the partial forgetting function is involved in the argument of the softmax. Discretizing this differential equation by taking $\Delta t = \tau_v$, we eventually obtain the update rule for the feature neurons, which is essentially the attention mechanism in transformers, but with slight modification. More specifically, the former attention

$$Z_{ij} = \sum_\mu V_{i\mu}\mathrm{softmax}(\beta \sum_k K_{\mu k} Q_{kj}), \tag{30}$$

where β is a scaling factor, and $Q_{ij}, K_{i\mu}, V_{i\mu}$ are query, key, and value matrices, respectively, becomes

$$Z_{ij} = \sum_{\mu \in M_j} V_{i\mu}\mathrm{softmax}(D(\beta \sum_k K_{\mu k} Q_{kj})), \tag{31}$$

where $M_j = \{\mu : \chi(\beta \sum_i K_{\mu i} Q_{ij}) = 1\}$ in our model. This expression may imply that the non-bijective projections by the partial forgetting function D remove information from features lying below a cutoff value, which is either deterministic or stochastic aiming that the learned network roughly identifies more relevant and less relevant features for a given sample.

4 Experiments

In this section, we embed the partial forgetting functionality in modern Hopfield networks to investigate how the model performance changes in various data domains. We demonstrate the effectiveness of LwPF for three diverse tasks, namely (1) bit pattern classification, (2) immune repertoire classification, and (3) image classification. The first one is a toy problem. We apply the proposed LwPF to small Hopfield networks (Ramsauer et al., 2021) to provide an evidence that LwPF improves training performance. The second one is a real-world problem from computational biology. We apply LwPF to DeepRC (Widrich et al., 2020), a medium-sized modern Hopfield network. The third one

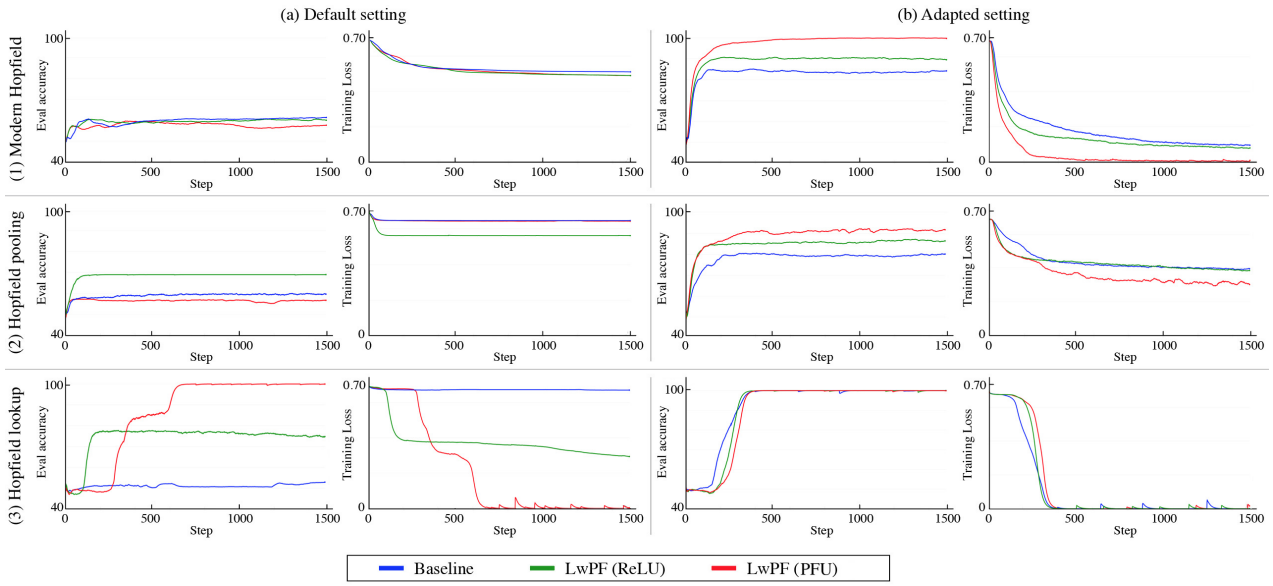


Figure 3: Demonstration of bit-pattern classification. Evaluation accuracy (%) and training loss are shown. The proposed LwPF was applied to Hopfield networks, each with modern Hopfield layers, Hopfield pooling layers, or Hopfield lookup layers. Results are for default and adapted hyperparameter settings.

is a real-world problem from computer vision. We introduce LwPF to vision transformers (Dosovitskiy et al., 2021; Touvron et al., 2021), large state-of-the-art neural networks. All the baseline models are a form of the large associative memory model by Krotov and Hopfield. The model of Ramsauer et al. (2021), the baseline of Sec. 4.1, the model of Widrich et al. (2020), the baseline of Sec. 4.2, and the vision transformer considered in Sec. 4.3 are all neural network architectures that consist of attention mechanism as a component. All the attention blocks in these models are regarded as realizations of the model B of large associative memory model by Krotov and Hopfield (Sec. 2.2). In each experiment, we chose a specific optimizer that matches the one used in the corresponding baseline so that the comparison is as fair as possible. Other experimental settings, such as learning rate, weight decay, optimizer-specific hyperparameters, etc., are also the same as the corresponding baselines. One exception is the batch size used in the image classification task. We reduce the batch size because we encountered a memory issue when the original batch size is used in our environment. We present and discuss the main results below. Some details are presented in the supplementary materials.

4.1 Bit pattern classification

Motivation. The goal of this subsection is to provide the first evidence that LwPF improves the training performance of Hopfield networks. We use a simple toy dataset and small network architectures.

Task and measure. Bit pattern classification is a binary classification task in the domain of multiple instance learning. We use the bit pattern dataset provided by Ramsauer et al. (2021). It consists of a collection of bit pattern instances, each of which is a sequence of zeros and ones. The positive class has specific bit patterns, which are absent in the negative one. All dataset parameters are default ones, that is, the dataset has 8-bit sequences, 2,048 bags (1,536 for training and 512 for evaluation), 16 instances per bag, 8 unique instances indicative for the positive class, and one signal implanted into one bag of the positive class. We report training loss and evaluation accuracy.

Network architecture. We use three Hopfield networks provided in the official `hflayers` implementation by Ramsauer et al. (2021). They consist of (1) a modern Hopfield layer, (2) a Hopfield pooling layer, and (3) a Hopfield lookup layer, respectively, each followed by a linear projection for binary classification. Two architecture parameter settings, namely the default settings (DS) and the adapted settings (AS), are used for evaluation. DS uses the default parameters implemented with `hflayers`. AS adapts three Hopfield update steps, an increased number of heads, and separated weights for lookup. Details are presented in Appendix A. LwPF is introduced to the attention mechanism in the Hopfield update steps with $m = \bar{x}_{\text{median}}, \sigma = 0.01$ and $b = z$.

Training. The AdamW optimizer (Loshchilov and Hutter, 2019) with a learning rate of 10^{-3}, a weight decay of 0.01 and $(\beta_1, \beta_2) = (0.9, 0.999)$ is used for 1,500 steps. The objective function to be minimized is binary cross-entropy loss.

Results. Figure 3 shows the results. It can be seen that

Table 1: Immune repertoire classification results for CMV dataset (Emerson et al., 2017). The area under the curve (multiplied by 100) is reported with the standard deviation across five cross-validation folds. n_c: kernel size. n_k: number of kernels. Baseline uses DeepRC (Widrich et al., 2020) (the official implementation is used). LwPF applies the proposed partial forgetting function to the baseline.

Method	$n_c = 7$			$n_c = 9$		
	$n_k = 8$	$n_k = 16$	$n_k = 32$	$n_k = 8$	$n_k = 16$	$n_k = 32$
Baseline	75.01±2.33	73.45±1.13	79.13±0.85	80.39±0.34	76.67±3.52	82.06±0.47
LwPF (ReLU)	**79.64±0.97**	74.62±1.11	50.00±0.00	80.36±0.89	50.00±0.00	80.86±0.59
LwPF (PFU)	76.90±1.07	**75.64±1.32**	**82.20±0.90**	**80.92±0.50**	**82.81±0.55**	**83.14±0.38**

LwPF with ReLU and PFU achieve a lower training loss than the baseline in all settings. This shows that LwPF improves the training performance of the modern Hopfield networks.

The best performing method transitions from the baseline to LwPF with ReLU and then LwPF with PFU as the network size increases, *i.e.*, the baseline performs the best for the smallest case (DS with modern Hopfield layer), ReLU performs the best for the second smallest case (DS with Hopfield pooling layer), and PFU performs the best for the other cases. This suggests that the optimal partial forgetting function depends on the number of network parameters. These results indicate that LwPF with ReLU should be used for small networks, and PFU should be used for relatively large networks.

4.2 Immune repertoire classification

Motivation. This demonstration aims to show the effectiveness of LwPF for a real-world task with a medium-sized network (∼10k parameters). We apply LwPF to DeepRC (Widrich et al., 2020), a modern Hopfield network.

Task and measure. Immune repertoire classification is a problem of multiple instance learning in computational biology. We follow the problem settings by Widrich et al. (2020), in which a neural network predicts the immune status based on the input immune repertoire represented by bags of immune receptor sequences. The real-world CMV dataset (Emerson et al., 2017), which consists of T-cell repertoire with known cytomegalovirus serostatus, is used for evaluation. The number of subjects is 666. The average number of sequences per immune repertoire is about 300k. We report the area under the curve (AUC) across five-fold cross-validation.

Network architecture. We use DeepRC, a modern Hopfield network architecture proposed by Widrich et al. (2020). It consists of an input layer that concatenates amino-acid features and position features, a 1D convolutional block with a head to take the maximum value over sequence positions, a sequence of attention blocks with a head to take the summation, and a linear layer to perform binary classification. More details are presented in Appendix B. Experiments are conducted with all combinations of $n_c = 7, 9$ and $n_k = 8, 16, 32$, where n_c and n_k are the kernel size and the number of kernels for convolution, respectively. LwPF is introduced to the sequence of the attention block with $m = \bar{x}_{\text{median}}, \sigma = 0.01$ and $b = z$.

Training. We use the official implementation of DeepRC and the recommended training hyper-parameters, *i.e.*, the Adam optimizer (Kingma and Ba, 2015) is used with learning rate of 10^{-4} without weight decay, $(\beta_1, \beta_2) = (0.9, 0.999)$, and batch size of 4 for 10^5 steps. The objective function to be minimized is cross-entropy loss.

Results. Table 1 summarizes the results. It can be seen that LwPF with PFU always outperforms the baseline. ReLU performs the best with the smallest settings ($n_c = 7, n_k = 8$), and PFU performs the best with the others. This tendency is the same as that for bit pattern classification in Sec. 4.1.

A limitation of ReLU can also be seen. The results of LwPF with ReLU are indeed very sensitive to the experimental setting and ReLU fails in training with some settings (e.g., $n_c = 7, n_k = 32$), in which $M = \emptyset$, *i.e.*, all the inputs fall into the forgetting region, during training. This limitation is avoided for PFU by stochastic forgetting region and taking the dynamic median of the inputs as $m = \bar{x}_{\text{median}}$. The results above indicate that PFU is stable and thus recommended for medium-sized networks in practice.

4.3 Image classification

Motivation. In this subsection we demonstrate the effectiveness of LwPF for a real-world problem from computer vision. We incorporate LwPF into vision transformers, which are large state-of-the-art neural networks in computer vision with ∼100M parameters.

Task and measure. In image classification, a fundamental problem in computer vision, the aim is to classify images into a pre-defined finite number of categories. We use the CIFAR-10 and CIFAR-100 datasets (Krizhevsky, 2009), each of which consists of 60,000 natural images. The image size is 32×32. The ground-truth object category labels are attached to each image. The number of categories is

Table 2: Image classification accuracy (%).

Method	ViT-B	
	CIFAR-10	CIFAR-100
Baseline	89.40	65.45
LwPF (ReLU)	**90.39**	65.39
LwPF (PFU)	90.17	**65.82**

10 for CIFAR-10 and 100 for CIFAR-100. We report the classification accuracy (%) for each dataset.

Network architecture. We use the vision transformer (ViT), the transformer network for image classification proposed by Dosovitskiy et al. (2021); Touvron et al. (2021). We use Vit-Base (B), whose number of parameters and the number of attention heads are 86.6M and 12 heads. More details are presented in Appendix C. LwPF is introduced to all attention blocks of the transformer encoders with $m = 0, \sigma = 1.0$ and $b = 0$. We fixed m to zero in this task to efficiently run experiments with multiple GPUs without synchronization at each partial forgetting function.

Training. We follow the training settings by Touvron et al. (2021). Images are resized to 224×224. The SGD optimizer is used with a learning rate of 0.01 and a weight decay of 10^{-4} for 1,000 epochs. The batch size is 192. The objective function to be minimized is cross-entropy loss. The implementation of PyTorch Image Models `timm` (Wightman, 2019) is used. All experiments are conducted with four NVIDIA V100 GPUs.

Results. Table 2 shows the results. LwPF with ReLU or PFU consistently improve the performance of the vision transformer. This confirms the effectiveness of our approach using the Hopfield-based formulation for state-of-the-art transformers in computer vision.

From Table 2, one finds that LwPF with ReLU decreases the performance with CIFAR-100. This tendency is in a sense similar to the results for the bit pattern classification and immune repertoire classification. As in the previous two experiments, ReLU performs the best with the simplest settings, and PFU performs the best with the others. In addition, these empirical evaluations seem to be very interesting. Although the modification of the vision transformer is relatively simple, the Table 2 indicates that the proposed LwPF with CIFAR-10 achieves more than 90% accuracy, while in the range of hyperparameters considered in our experiments the typical results of the vision transformer on CIFAR-10 would be in the lower 80% range. These results indicate that LwPF can improve the performance of state-of-the-art vision transformers, but the partial forgetting function needs to be carefully designed. Further discussion is provided in the next section.

5 Discussion and Limitation

For improving the performance of modern Hopfield networks, we proposed LwPF which implements partial forgetting functionality for memory neurons during training. We also introduced a new attention mechanism with the partial forgetting functionality as a natural extension. We demonstrated the effectiveness of LwPF on three diverse tasks and showed that LwPF improves the performance of existing models including DeepRC and vision transformers. It is remarkable that the performance of these models gains such large improvements just by simply changing the feedforward operation.

From a neuroscientific point of view, partial and transient forgetting plays a crucial role in the brain to improve performance for certain intellectual activities. This work provides analogous evidences that modern Hopfield network models equipped with our forgetting functionality consistently outperform the counterparts without such a functionality. Training with this forgetting functionality could prioritize a type of features that are indispensable for a large portion of data in a way that the activation values become relatively large so as to gain robustness against the partial forgetting. In other words, the network would reduce the probability of losing these information by entering the stochastic forgetting region, and such an implicit mechanism may bring positive effects for testing. Below, we discuss limitations and broader impacts.

Limitations. One limitation of LwPF is that the performance depends on the choice of partial forgetting function D and its hyperparameters. Although we demonstrated an improvement in performance for three distinct tasks used artificial data, biological data, and image data, the best performing partial forgetting function depends on each task.

Another limitation of this work lies in the narrowed focus on the attention mechanism. We applied LwPF to the attention mechanism because many state-of-the-art methods rely on it. However, the framework itself can be applied to other types of Hopfield networks including modern extensions (Millidge et al., 2022; Tang and Kopp, 2021; Krotov, 2021).

Broader impacts. Transformers are the novel state-of-the-art neural network models in almost all the domains of computer science and machine learning in these days. Our work provides a variant of transformers through the update rule of modern Hopfield networks equipped with partial forgetting functionality. From this point of view, we can further study the fundamental properties of the novel attention mechanism in state-of-the-art models by mapping them to the dynamics of Hopfield networks, which are more transparent neural network models. In addition, this perspective enables us to explore more efficient and generalizable transformer models in a transparent way, not in ad hoc ways from tasks to tasks.

Since our experiments were designed to verify the formu-

lation of our proposed LwPF, we believe that our results do not directly cause harm to society, while they can be of use to develop new transformer architectures. It is expected that a more promising Hopfield model with a set of Lagrangians and an energy function will be a good starting point to pursue new state-of-the-art neural network models.

Acknowledgement

This work was an outcome of a research project, Development of Quality Foundation for Machine-Learning Applications, supported by DENSO IT LAB Recognition and Learning Algorithm Collaborative Research Chair (Tokyo Tech.). This work was also supported by JSPS KAKENHI Grant Number JP22H03642.

References

Elena Agliari and Giordano De Marzo. Tolerance versus synaptic noise in dense associative memories. *The European Physical Journal Plus*, 135(11):1–22, 2020.

Adriano Barra, Matteo Beccaria, and Alberto Fachechi. A new mechanical approach to handle generalized hopfield neural networks. *Neural Networks*, 106:205–222, 2018.

Mete Demircigil, Judith Heusel, Matthias Löwe, Sven Upgang, and Franck Vermet. On a model of associative memory with huge storage capacity. *Journal of Statistical Physics*, 168(2):288–299, 2017.

A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representation (ICLR)*, 2021.

Ryan O Emerson, William S DeWitt, Marissa Vignali, Jenna Gravley, Joyce K Hu, Edward J Osborne, Cindy Desmarais, Mark Klinger, Christopher S Carlson, John A Hansen, et al. Immunosequencing identifies signatures of cytomegalovirus exposure history and HLA-mediated effects on the T cell repertoire. *Nature Genetics*, 49(5): 659, 2017.

Andreas Fürst, Elisabeth Rumetshofer, Viet Tran, Hubert Ramsauer, Fei Tang, Johannes Lehner, David Kreil, Michael Kopp, Günter Klambauer, Angela Bitto-Nemling, et al. Cloob: Modern hopfield networks with infoloob outperform clip. *arXiv preprint arXiv:2110.11316*, 2021.

Masaya Hirashima and Daichi Nozaki. Learning with slight forgetting optimizes sensorimotor transformation in redundant motor systems. *PLoS Comput Biol*, 8(6), 2012.

J J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, 1982.

J J Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences*, 81(10): 3088–3092, 1984.

Sheena A. Josselyn and Susumu Tonegawa. Memory engrams: Recalling the past and imagining the future. *Science*, 367(6473):eaaw4325, 2020.

Diederick P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Alex Krizhevsky. Learning multiple layers of features from tiny images. In *Technical report, CIFAR*, 2009.

Dmitry Krotov. Hierarchical associative memory. *arXiv preprint arXiv:2107.06446*, 2021.

Dmitry Krotov and John Hopfield. Dense associative memory is robust to adversarial inputs. *Neural computation*, 30(12):3151–3167, 2018.

Dmitry Krotov and John J. Hopfield. Dense associative memory for pattern recognition. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

Dmitry Krotov and John J. Hopfield. Large associative memory problem in neurobiology and machine learning. In *International Conference on Learning Representations*, 2021.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Beren Millidge, Tommaso Salvatori, Yuhang Song, Thomas Lukasiewicz, and Rafal Bogacz. Universal hopfield networks: A general framework for single-shot associative memory models. *arXiv preprint arXiv:2202.04557*, 2022.

Omid Miry, Jie Li, and Lu Chen. The quest for the hippocampal memory engram: From theories to experimental evidence. *Frontiers in Behavioral Neuroscience*, 14, 2021.

Ulises Pereira and Nicolas Brunel. Attractor dynamics in networks with learning rules inferred from in vivo data. *Neuron*, 99(1):227–238.e4, 2018.

Mm Poo, Michele Pignatelli, Tomás J. Ryan, Susumu Tonegawa, Tobias Bonhoeffer, Kelsey C. Martin, Andrii Rudenko, Li-Huei Tsai, Richard W. Tsien, Gord Fishell, Caitlin Mullins, J. Tiago Gonçalves, Matthew Shtrahman, Stephen T. Johnston, Fred H. Gage, Yang Dan, John Long, György Buzsáki, and Charles Stevens. What is memory? the present state of the engram. *BMC Biol*, 14(40), 2016.

Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Lukas Gruber, Markus Holzleitner, Thomas Adler, David Kreil, Michael K Kopp, Günter Klambauer, Johannes Brandstetter, and

Sepp Hochreiter. Hopfield networks is all you need. In *International Conference on Learning Representations*, 2021.

John Martin Sabandal, Jacob A. Berry, and Ronald L. Davis. Dopamine-based mechanism for transient forgetting. *Nature*, 591(7850):426–430, 2021.

Fathi M Salem. Unbounded capacity associative memory for real-valued pattern storage and recall. In *2021 IEEE International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 1066–1069. IEEE, 2021.

Tommaso Salvatori, Yuhang Song, Yujian Hong, Lei Sha, Simon Frieder, Zhenghua Xu, Rafal Bogacz, and Thomas Lukasiewicz. Associative memories via predictive coding. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 3874–3886. Curran Associates, Inc., 2021.

Bernhard Schäfl, Lukas Gruber, Angela Bitto-Nemling, and Sepp Hochreiter. Hopular: Modern hopfield networks for tabular data. *arXiv preprint arXiv:2206.00664*, 2021.

Alexander Seeholzer, Moritz Deger, and Wulfram Gerstner. Stability of working memory in continuous attractor networks under the control of short-term plasticity. *PLoS Comput Biol*, 15(4), 2019.

Philipp Seidl, Philipp Renz, Natalia Dyubankova, Paulo Neves, Jonas Verhoeven, Jörg K Wegner, Marwin Segler, Sepp Hochreiter, and Günter Klambauer. Improving few- and zero-shot reaction template prediction using modern hopfield networks. *Journal of Chemical Information and Modeling*, 2022.

Davide Spalla, Isabel Maria Cornacchia, and Alessandro Treves. Continuous attractors for dynamic memories. *Elife*, 2021.

Fei Tang and Michael Kopp. A remark on a paper of krotov and hopfield [arxiv: 2008.06996]. *arXiv preprint arXiv:2105.15034*, 2021.

H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning (ICML)*, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Michael Widrich, Bernhard Schäfl, Milena Pavlović, Hubert Ramsauer, Lukas Gruber, Markus Holzleitner, Johannes Brandstetter, Geir Kjetil Sandve, Victor Greiff, Sepp Hochreiter, and Günter Klambauer. Modern hopfield networks and attention for immune repertoire classification. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 18832–18845. Curran Associates, Inc., 2020.

Michael Widrich, Markus Hofmarcher, Vihang Prakash Patil, Angela Bitto-Nemling, and Sepp Hochreiter. Modern hopfield networks for return decomposition for delayed rewards. In *Deep RL Workshop NeurIPS 2021*, 2021.

Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019.

Yan Wu, Gregory Wayne, Karol Gregor, and Timothy Lillicrap. Learning attractor dynamics for generative memory. *Advances in Neural Information Processing Systems*, 31, 2018.

Learning with Partial Forgetting in Modern Hopfield Networks: Supplementary Materials

Appendix A: Modern Hopfield networks for bit pattern classification

This section reviews the three Hopfield layers (Ramsauer et al., 2021) to explicitly show how LwPF is introduced to them. We also provide hyperparameter settings.

LwPF for Hopfield layers. The first Hopfield layer (`Hopfield`) is a layer for propagating a set of vectors via query patterns $\boldsymbol{R} = (\boldsymbol{r}_1, \cdots, \boldsymbol{r}_S)^\top \in \mathbb{R}^{S \times d_r}$ and key patterns $\boldsymbol{Y} = (\boldsymbol{y}_1, \cdots, \boldsymbol{y}_N)^\top \in \mathbb{R}^{N \times d_y}$ as

$$\boldsymbol{Z} = \mathrm{softmax}(\beta \boldsymbol{R}\boldsymbol{W}_Q \boldsymbol{W}_K^\top \boldsymbol{Y}^\top)\boldsymbol{Y}\boldsymbol{W}_V, \tag{32}$$

where $\boldsymbol{W}_K \in \mathbb{R}^{d_y \times d_k}, \boldsymbol{W}_Q \in \mathbb{R}^{d_r \times d_k}, \boldsymbol{W}_V \in \mathbb{R}^{d_y \times d_v}$ are matrices for Hopfield-based propagation, $\beta > 0$ is a scaling parameter, and $S, d_r, N, d_y, d_k \in \mathbb{N}_{>0}$ denote dimensions. LwPF is introduced to it as

$$Z_{ij} = \sum_{\mu \in M_j} [\mathrm{softmax}(D(\beta \boldsymbol{R}\boldsymbol{W}_Q \boldsymbol{W}_K^\top \boldsymbol{Y}^\top))]_{\mu j} [\boldsymbol{Y}\boldsymbol{W}_V]_{i\mu}, \tag{33}$$

where $M_j = \{\mu : [\chi(\beta \boldsymbol{R}\boldsymbol{W}_Q \boldsymbol{W}_K^\top \boldsymbol{Y}^\top)]_{\mu j} = 1\}$, and $[\cdot]_{ij}$ denotes the (i, j) element of a matrix.

The Hopfield pooling layer (`HopfieldPooling`) is a layer for propagating patterns via the key patterns $\boldsymbol{Y} \in \mathbb{R}^{N \times d_y}$ as

$$\boldsymbol{Z} = \mathrm{softmax}(\beta \boldsymbol{Q}\boldsymbol{W}_K^\top \boldsymbol{Y}^\top)\boldsymbol{Y}\boldsymbol{W}_V, \tag{34}$$

where $\boldsymbol{Q} \in \mathbb{R}^{S \times d_k}, \boldsymbol{W}_K \in \mathbb{R}^{d_y \times d_k}, \boldsymbol{W}_V \in \mathbb{R}^{d_y \times d_v}$. LwPF is introduced to it as

$$Z_{ij} = \sum_{\mu \in M_j} [\mathrm{softmax}(D(\beta \boldsymbol{Q}\boldsymbol{W}_K^\top \boldsymbol{Y}^\top))]_{\mu j} [\boldsymbol{Y}\boldsymbol{W}_V]_{i\mu}. \tag{35}$$

where $M_j = \{\mu : [\chi(\beta \boldsymbol{Q}\boldsymbol{W}_K^\top \boldsymbol{Y}^\top)]_{\mu j} = 1\}$.

The Hopfield lookup layer (`HopfieldLayer`) is a layer for propagating a vector or a set of vectors via query patterns $\boldsymbol{R} \in \mathbb{R}^{S \times d_r}$ as

$$\boldsymbol{Z} = \mathrm{softmax}(\beta \boldsymbol{R}\boldsymbol{W}_K^\top)\boldsymbol{W}_V, \tag{36}$$

where $\boldsymbol{W}_K \in \mathbb{R}^{d_y \times d_r}, \boldsymbol{W}_V \in \mathbb{R}^{d_y \times d_v}$. LwPF is introduced to it as

$$Z_{ij} = \sum_{\mu \in M_j} [\mathrm{softmax}(D(\beta \boldsymbol{R}\boldsymbol{W}_K^\top))]_{\mu j} [\boldsymbol{W}_V]_{i\mu}. \tag{37}$$

where $M_j = \{\mu : [\chi(\beta \boldsymbol{R}\boldsymbol{W}_K^\top)]_{\mu j} = 1\}$.

Hyperparameters. Table 3 summarizes the two hyperparameters settings, DS and AS, described in Sec. 4.1. We used the official implementation[1] for running experiments.

[1] https://github.com/ml-jku/hopfield-layers

Table 3: Hyperparameters for DS and AS

Hyperparameter	DS	AS
Size of input	8	8
Size of association space (hidden size)	8	8
Number of parallel association heads	1	8
Number of updates in one Hopfield head	0	3
Scaling parameter	1.0	0.25
Dropout parameter	0.0	0.5
Separated lookup weights	False	True
Trainable lookup targets	True	False

Appendix B: DeepRC for immune repertoire classification

This section reviews the DeepRC architecture. Table 4 shows the architecture of DeepRC, which takes amino acid features and position features as inputs and performs binary classification. LwPF is applied to its attention, that is,

$$\boldsymbol{Z} = \mathrm{softmax}(\beta \boldsymbol{Q}\boldsymbol{K}^\top)\boldsymbol{V}, \tag{38}$$

is replaced by

$$Z_{ij} = \sum_{\mu \in M_j} [\mathrm{softmax}(D(\beta \boldsymbol{Q}\boldsymbol{K}^\top))]_{\mu j} V_{i\mu}. \tag{39}$$

where $M_j = \{\mu : [\chi(\beta \boldsymbol{Q}\boldsymbol{K}^\top)]_{\mu j} = 1\}$. The detailed architecture of the 1D-CNN and SNN blocks can be found in the official implementation of DeepRC[2]. All training hyperparameters are default and not tuned with LwPF.

Table 4: Architecture of DeepRC

	Formulation	Shape
Amino acid features	$\boldsymbol{X}_a$	$(N, d, 20)$
Position features	$\boldsymbol{X}_p$	$(N, d, 3)$
Concatenation layer	$\boldsymbol{X} = [\boldsymbol{X}_p; \boldsymbol{X}_a]$	$(N, d, 23)$
1D-CNN block (1 layer, n_k kernels, n_c kernel size)	$\boldsymbol{H} = \mathrm{CNN}(\boldsymbol{X})$	(N, d, n_k)
Max pooling	$\boldsymbol{V} = \mathrm{MaxPool}(\boldsymbol{H})$	(N, n_k)
SNN block (2 layer, 32 features)	$\boldsymbol{K} = \mathrm{SNN}(\boldsymbol{V})$	$(N, 32)$
Attention	$\boldsymbol{Z} = \mathrm{softmax}(\beta \boldsymbol{Q}\boldsymbol{K}^\top)\boldsymbol{V}$	(N, n_k)
Summation	$\boldsymbol{s} = \mathrm{Sum}(\boldsymbol{Z})$	(n_k)
Linear projection	$\boldsymbol{p} = \mathrm{Linear}(s)$	(1)

Appendix C: Vision transformers for image classification

This section reviews vision transformer (ViT) architectures (Dosovitskiy et al., 2021; Touvron et al., 2021) to provide detailed experimental settings.

LwPF for vision transformers. Table 5 shows the architecture parameters for the three vision transformers, ViT-Ti, ViT-S, and ViT-B. All of them are trained with a patch size of 16×16 and a resolution of 224×224. Note that each transformer-encoder layer has an attention module. LwPF is applied to all attention modules.

We used PyTorch Image Models `timm`[3]. The ViT-Ti, ViT-S, and ViT-B architectures are specified with the options `deit_tiny_patch16_244`, `deit_small_patch16_244`, and `deit_base_patch16_244`, respectively.

[2]https://github.com/ml-jku/DeepRC

[3]https://github.com/rwightman/pytorch-image-models

Table 5: Architecture of vision transformers

Model	Embedding dim.	# heads	Dim. per head	# layers	# params
ViT-Ti	192	3	64	12	5.7M
ViT-S	384	6	64	12	22.1M
ViT-B	768	12	64	12	86.6M

Hyperparameter settings. Finally, we report the results of studies on LwPF hyperparameters. Note that all training hyperparameters are fixed to ones, as recommended in (Touvron et al., 2021), which are for training vanilla ViTs, as described in Sec. 4.3. We did not adjust the training hyperparameters to LwPF to solely evaluate whether LwPF improves generalization ability. Note that tuning the training hyperparameters with LwPF may further improve performance.

The early stopping results are reported in Tables 6 and 7. In Table 6, LwPF with ReLU and PFU outperform the baseline at epoch 250 for all the ViTs. This suggests that the partial forgetting function should be scheduled. We used a fixed definition of D during training in this work. Extending D to D_t, where t is the number of iterations or epochs, to dynamically control the strength of forgetting in training would be interesting in future work.

Table 6: Results at epochs 250, 500, and 1000 for CIFAR-10. Evaluation accuracy (Acc.; %) and training loss (Loss) are reported. Underline values indicate better performance than the baseline

Method	Epoch	ViT-Ti		ViT-S		ViT-B	
		Acc.	Loss	Acc.	Loss	Acc.	Loss
Baseline	250	63.58	1.951	71.56	1.879	73.93	1.849
	500	73.80	1.747	81.47	1.787	82.58	1.717
	1,000	85.36	1.680	88.71	1.574	89.40	1.579
LwPF (ReLU)	250	64.66	1.920	71.73	1.865	74.72	1.826
	500	74.66	1.832	80.83	1.777	84.56	1.694
	1,000	84.30	1.693	87.70	1.646	90.39	1.567
LwPF (PFU)	250	64.70	1.922	71.65	1.856	74.71	1.813
	500	74.58	1.835	80.99	1.776	84.30	1.694
	1,000	84.26	1.692	87.61	1.647	90.17	1.568

Table 7: Results at epochs 250, 500, and 1000 for CIFAR-100. Evaluation accuracy (Acc.; %) and training loss (Loss) are reported. Underline values indicate better performance than the baseline

Method	Epoch	ViT-Ti		ViT-S		ViT-B	
		Acc.	Loss	Acc.	Loss	Acc.	Loss
Baseline	250	41.35	3.883	49.65	3.578	55.72	3.537
	500	53.88	3.571	59.12	3.300	61.93	3.151
	1,000	61.93	3.243	64.17	2.936	65.45	2.764
LwPF (ReLU)	250	39.04	3.898	49.52	3.614	54.77	3.549
	500	51.00	3.614	59.08	3.388	60.70	3.062
	1,000	61.01	3.242	62.13	2.882	65.39	2.673
LwPF (PFU)	250	39.04	3.898	49.44	3.612	54.75	3.550
	500	51.00	3.614	59.20	3.388	60.71	3.063
	1,000	60.84	3.242	61.67	2.859	65.82	2.661