Retrieving k-Nearest Memories with Modern Hopfield Networks

Alexander Davydov[1,*], **Sean Jaffe**[1,2], **Ambuj K. Singh**[2], **and Francesco Bullo**[1]

[1] Center for Control, Dynamical Systems and Computation, University of California, Santa Barbara.
[2] Department of Computer Science, University of California, Santa Barbara.

Abstract

Modern continuous Hopfield networks (MCHNs) are a variant of Hopfield networks that have greater storage capacity and have been shown to have connections to the attention mechanism in transformers. In this paper, we propose a variant of MCHNs, which we call k-Hopfield layers, which is the first Hopfield-type network that retrieves the k-nearest memories to a given input. k-Hopfield layers are differentiable and may serve as (i) a soft approach to k-nearest neighbors, (ii) an augmented form of memory in deep learning architectures and (iii) an alternative to multihead attention in transformers. We empirically demonstrate that increasing k aids in correctly reconstructing a corrupted input. We show that using a k-Hopfield layer as a replacement to multihead attention demonstrates comparable performance in small vision transformers while requiring fewer parameters.

1 Introduction

Hopfield networks [Hopfield, 1982, 1984] are a class of neural networks with an underlying energy function and are typically used as a form of associative memory. Given a partial or noise-corrupted memory, the network updates its state to minimize its energy, converges to a fixed point, and retrieves the full memory. Hopfield networks have garnered interest thanks to recent developments on dense associative memories and modern Hopfield networks. It has been demonstrated that these newer variants possess greater storage capacity compared to their classical counterparts [Krotov and Hopfield, 2016, Demircigil et al., 2017, Ramsauer et al., 2021, Krotov and Hopfield, 2021]. Further, [Ramsauer et al., 2021] shows a connection between modern continuous Hopfield networks (MCHNs) and the attention mechanism in transformers [Vaswani et al., 2017]. Building upon these breakthroughs in dense associative memories and MCHNs, recent interesting extensions include (i) universal Hopfield networks [Millidge et al., 2022], (ii) kernel interpretations of MCHNs [Iatropoulos et al., 2022], and (iii) Hopfield networks with setwise connections [Burns and Fukai, 2023]. Finally, Hopfield networks can be used to design new transformer architectures [Hoover et al., 2023].

In this paper, we propose a variant of MCHN to retrieve the k-nearest memories to an input. We call this variant a *k-Hopfield layer*. In short, given a collection of memories $\{\xi_i\}_{i=1}^N$, and input x_0, the k-Hopfield layer retrieves the k memories $\xi_{i_1}, \dots, \xi_{i_k}$ that are most similar to x_0. We provide a pictorial representation of this procedure in Figure 1. Our k-Hopfield layer is built upon a soft top-k operator [Amos et al., 2019], akin to how MCHNs use $\mathrm{softmax}$ as a soft approximation for $\arg\max$. In this way, our k-Hopfield layer is differentiable and can be seamlessly integrated into deep learning architectures trained via gradient descent. To the best of our knowledge, the k-Hopfield layer is the first Hopfield-type network to retrieve more than one memory. We demonstrate the k-Hopfield layer's ability to reconstruct obscured memories. We additionally demonstrate the utility of our approach as a multi-head attention substitute that uses fewer trainable parameters with comparable performance on small-scale vision transformers [Dosovitskiy et al., 2021].

*Corresponding author: davydov@ucsb.edu

Associative Memory & Hopfield Networks in 2023. NeurIPS 2023 workshop.

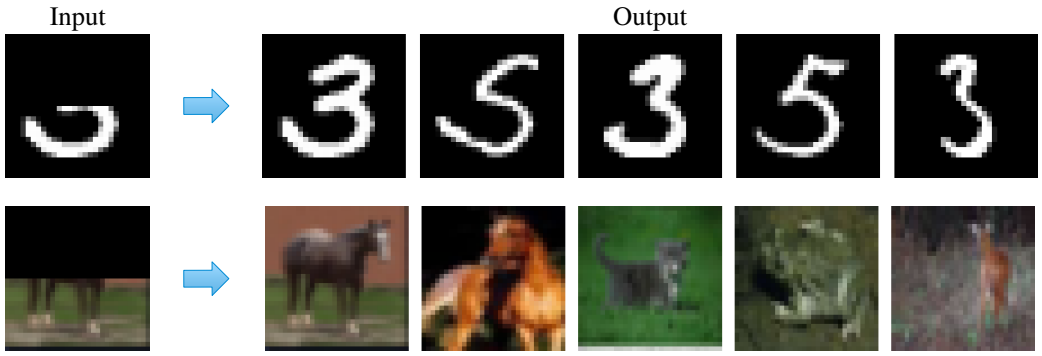

Figure 1: Pictorial representation of the k-Hopfield layer with $k = 5$. The left images are the occluded MNIST and CIFAR-10 inputs to the k-Hopfield layer and the right images are the outputs of the k-Hopfield layer, i.e., the k closest stored memories to the occluded input.

2 Methods

Motivation and ksoftmax: Recall the standard update rule for the modern continuous Hopfield networks (MCHNs)

$$x^+ = \Xi \, \mathrm{softmax}(\beta \Xi^\top x), \tag{1}$$

where $\Xi = [\xi_1, \dots, \xi_N] \in \mathbb{R}^{n \times N}$ with $\{\xi_i\}_{i=1}^N$ being a collection of memories, and $\beta > 0$ is an inverse temperature parameter. It was shown in [Ramsauer et al., 2021] that when the memories are sufficiently well separated, the update rule (1) in one step retrieves the memory closest to x_0.

We motivate our proposed approach for modifying the update rule (1) to retrieve k nearest memories via an intuitive explanation for the effectiveness of (1) in memory retrieval. In the limit as $\beta \to \infty$, $\mathrm{softmax}(\beta \Xi^\top x)$ converges to the binary vector with unity in the entry corresponding to the largest entry of the vector $\Xi^\top x$. Then $\Xi \, \mathrm{softmax}(\beta \Xi^\top x)$ outputs the single memory which is closest to x.

Following the intuitive explanation above, we aim to define a new operation, $\mathrm{ksoftmax}$, which has the property that in the limit as $\beta \to \infty$, $\mathrm{ksoftmax}(\beta \Xi^\top x)$ converges to a binary $n \times k$ matrix where the i-th column corresponds to one of the i-th closest memories with unity in the entry of the vector corresponding to the i-th largest entry of $\Xi^\top x$. Specifically, $\mathrm{ksoftmax}$ serves as a smooth and soft approximation for the nonsmooth operator that outputs this binary matrix.

The key observation is the following: finding the index corresponding to the i-th largest entry of a vector may be computed by taking the top-i operator of the vector minus its top-$(i-1)$ operator, where we recall that the top-k operator of a vector x returns a binary vector of the same size as x, but with unity in the entries corresponding to the k-largest entries of x. Since the top-k operator is not differentiable, we replace it by a soft version. We discuss alternative approaches to retrieving k-nearest memories in Appendix A. In this work, we adopt the approach of [Amos et al., 2019], which they refer to as the *limited multi-label projection layer*. We will refer to this layer as the *sum-softmax function* in analogy with the classical softmax operation.

Definition 1. *The sum-softmax function,* $\mathrm{ssm} : \mathbb{R}^n \times \{1, \dots, n\} \to \mathbb{R}^n$ *is defined by*

$$\mathrm{ssm}(x; k) = \operatorname*{arg\,min}_{y \in [0,1]^n} \; -x^\top y - H_{\mathrm{b}}(y), \qquad \textit{s.t.} \quad \mathbb{1}_n^\top y = k, \tag{2}$$

where $H_{\mathrm{b}}(y) = -\sum_{i=1}^n (y_i \log(y_i) + (1 - y_i) \log(1 - y_i))$ *is the binary entropy function.*

The sum-softmax function and softmax share the following similarity: if $k = 1$ and H_{b} in the objective function (2) is replaced with the entropy function $H(y) = -\sum_{i=1}^n y_i \log(y_i)$, then the solution to the minimization problem is equal to the output of the softmax function. The choice of using H_{b} rather than H in the objective function encourages less sparse gradients [Amos et al., 2019]. We establish in Appendix B that the ssm function converges to the top-k operator as $\beta \to \infty$. The forward and backward passes for the ssm function were implemented in Pytorch in [Amos et al., 2019] and can be directly applied for the problem under consideration.

With the ssm function in hand, we are ready to define the $\mathrm{ksoftmax}$ function.

Definition 2. *For $k \leq n$, define the k-softmax function* $\mathrm{ksoftmax} : \mathbb{R}^n \to \mathbb{R}^{n \times k}$ *one column at a time, for $i = 1, \ldots, k$, by:*

$$\mathrm{ksoftmax}_i(x) = \begin{cases} \mathrm{ssm}(x; 1), & \text{if } i = 1, \\ \mathrm{ssm}(x; i) - \mathrm{ssm}(x; i-1), & \text{otherwise} \end{cases} \tag{3}$$

Note that since $\mathrm{ssm}(x; i)$ provides a smooth approximation for the binary vector with unity in entries corresponding to the top-i entries of x, then $\mathrm{ksoftmax}_i(x)$ is an approximation to the binary vector with unity in the entry corresponding to the i-th largest entry of x. Moreover, this vector always has nonnegative entries (we prove this intuitive fact in Appendix B).

k-Hopfield Layer and Practical Considerations: Given an input $x_0 \in \mathbb{R}^n$, we propose the following single-step update, which we refer to as a *k-Hopfield layer*:

$$X = \Xi \, \mathrm{ksoftmax}(\beta \Xi^\top x_0). \tag{4}$$

Note that although the input x_0 is a vector, the output, $X \in \mathbb{R}^{n \times k}$ is a matrix where each column corresponds to a memory. The column X_i corresponds to an approximation to the i-th closest memory to the input x_0. Although the update rule (4) is a map from $\mathbb{R}^n$ to $\mathbb{R}^{n \times k}$, we provide an energy interpretation of the update rule when studied one column at a time in Appendix C (i.e., the dynamical system $x^+ = \Xi \, \mathrm{ksoftmax}_i(\beta \Xi^\top x)$, which takes the i-th column from ksoftmax).

As was studied in [Millidge et al., 2022], similarity measures other than the inner product may be desirable depending on the application domain. To this end, we can modify our update rule to accommodate more general similarity measures:

$$X = \Xi \, \mathrm{ksoftmax}(\beta \mathrm{sim}(\Xi, x)), \qquad \text{where } \; \mathrm{sim}(\Xi, x)_i = \mathrm{sim}(\xi_i, x), \tag{5}$$

where $\mathrm{sim}(\xi_i, x)$ denotes a measure of similarity between the vectors ξ_i and x and is "large" when ξ_i and x are close to one another. Other choices of similarity are the negative squared Euclidean distance, and the negative Manhattan distance, as reported in [Millidge et al., 2022].

Our update rule (4) or (5) can also readily be used as a first step in a larger dense associative memory. Specifically, given an input $x_0 \in \mathbb{R}^n$, one can consider the combined update

$$X = \Xi \, \mathrm{ksoftmax}(\beta_1 \Xi^\top x_0), \qquad X^+ = \Xi \, \mathrm{softmax}(\beta_2 \Xi^\top X), \tag{6}$$

where the softmax is applied to each column independently. Intuitively, the k-Hopfield update finds k new inputs which are close to the k memories closest to x_0 and then MCHN update retrieves these nearest memories in one step, as was shown in [Ramsauer et al., 2021].

3 Numerical Experiments

Image Reconstruction: In line with the experiments run in [Millidge et al., 2022] and [Burns and Fukai, 2023], we study the ability of the k-Hopfield layer to reconstruct corrupted memories as a function of both the number of stored memories and k. Specifically, we say that a k-Hopfield layer correctly reconstructs a memory if, given a corrupted input, any of the columns of the output matrix X are sufficiently close to the uncorrupted input. We say that a reconstruction is sufficiently close to the uncorrupted input if the sum-of-squares difference between the two is less than 50.

We embed data from MNIST, CIFAR-10, and Tiny ImageNet as memories into our k-Hopfield layer. We corrupt inputs as follows: if the dataset is (i) either MNIST or Tiny ImageNet, we occlude the top half of the input image, (ii) CIFAR-10, then we occlude the top $10/32$ rows of the input image. We use $\beta = 3$ and the negative Manhattan distance as similarity function in all experiments. We report the results for the k-Hopfield layer in additional to the MCHN in Figure 2.

We observe in all experiments that using $k = 1$ yields similar performance to using a standard MCHN for reconstruction. Additionally, we can empirically observe that using $k > 1$ provides improved ability to reconstruct the uncorrupted memory. This result makes intuitive sense since increasing k allows the layer to reconstruct multiple memories which are close to the input and check if any of these reconstructed memories are close to the uncorrupted memory.

ksoftmax Multihead Attention: Next, we assess using ksoftmax as a replacement for multihead attention. In this context, k represents the number of heads and the k-Hopfield attention update is

$$\mathrm{kHopfieldAttn(X)} = V \, \mathrm{ksoftmax}(\beta Q^\top K), \tag{7}$$

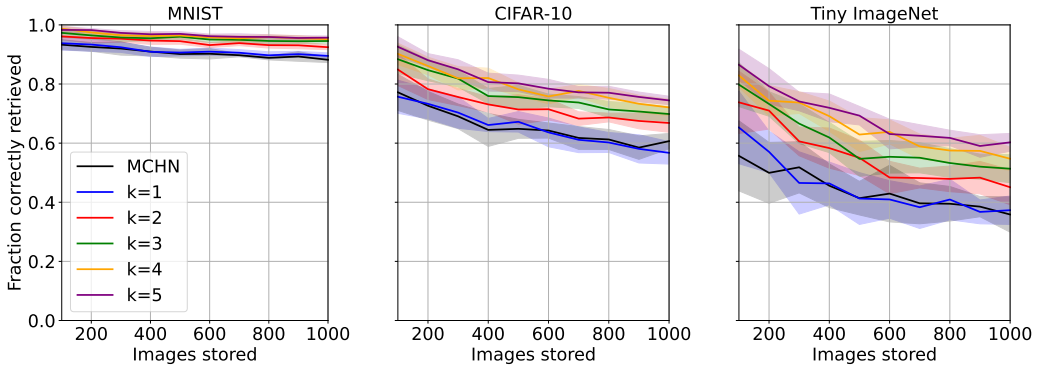

Figure 2: Storage capacity of the k-Hopfield layer with varying k and a MCHN [Ramsauer et al., 2021] as measured by increasing the number of memories. We average the performance over 10 trials with different memories and report the mean and one standard deviation. We observe that $k = 1$ performs comparably to MCHN and increasing k improves storage capacity (higher is better).

where $K = U_K X, Q = U_Q X$, and $V = U_V X$, denote the keys, values, and queries, respectively, and $\mathrm{ksoftmax}$ is applied to each column individually. Compared to standard multihead attention which runs k self-attention updates in parallel and concatenates them, the update (7) directly provides k different outputs. Thus, the update (7) serves as a substitute for multihead attention, while only requiring the parameters of a single head.

To test the k-Hopfield attention update, we train small vision transformers (ViT) [Dosovitskiy et al., 2021] with standard multihead attention and with k-Hopfield attention on MNIST (Figure 3) and CIFAR-10 (Appendix D). We provide training details in Appendix E. We observe that both multihead attention and k-Hopfield attention perform comparably.

4 Conclusion and Future Directions

We propose the k-Hopfield layer which can retrieve k-closest memories to an obscured input in a differentiable manner. We provide a mathematical treatment for the layer and provide an energy-based interpretation. The layer performs well when embedded in small vision transformers while requiring fewer parameters.

Future theoretical work will entail deciphering the provided energy function and convergence guarantees for the update rule. Future experimental work will entail testing on larger scale transformers.

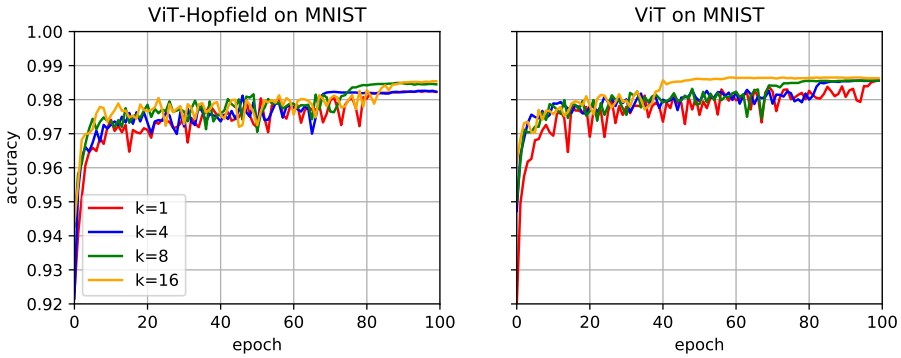

Figure 3: ViT training curves on MNIST with k-Hopfield attention (ViT-Hopfield) and standard multi-headed attention (ViT). k corresponds to the number of attention heads. We observe that performance of both methods is comparable, even though our method requires fewer parameters.

Acknowledgements: This work was supported in part by the NSF Graduate Research Fellowship under grant 2139319, the ARO under grant W911NF-22-1-0233, and by AFOSR award FA9550-22-1-0059.

References

R. Abraham, J. E. Marsden, and T. S. Ratiu. *Manifolds, Tensor Analysis, and Applications*, volume 75 of *Applied Mathematical Sciences*. Springer, 2 edition, 1988. ISBN 0387967907.

B. Amos, V. Koltun, and J. Z. Kolter. The limited multi-label projection layer. *arXiv preprint: 1906.08707*, 2019. URL https://arxiv.org/abs/1906.08707.

T. F. Burns and T. Fukai. Simplicial Hopfield networks. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=_QLsH8gatwx.

M. Demircigil, J. Heusel, M. Löwe, S. Upgang, and F. Vermet. On a model of associative memory with huge storage capacity. *Journal of Statistical Physics*, 168:288–299, 2017. doi:10.1007/s10955-017-1806-y.

A. Dosovitskiy, A. K. L. Beyer, 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 Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

B. Hoover, Y. Liang, B. Pham, R. Panda, H. Strobelt, D. H. Chau, M. J. Zaki, and D. Krotov. Energy transformer. *arXiv preprint arXiv:2302.07253*, 2023. URL https://arxiv.org/abs/2302.07253.

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. doi:10.1073/pnas.79.8.2554.

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. doi:10.1073/pnas.81.10.3088.

G. Iatropoulos, J. Brea, and W. Gerstner. Kernel memory networks: A unifying framework for memory modeling. In *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=px87A_nzK-T.

D. Krotov and J. J. Hopfield. Dense associative memory for pattern recognition. In *Advances in Neural Information Processing Systems*, 2016. URL https://arxiv.org/abs/1606.01164.

D. Krotov and J. J. Hopfield. Large associative memory problem in neurobiology and machine learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=X4y_10OX-hX.

Y. Liu, E. Sangineto, W. Bi, N. Sebe, B. Lepri, and M. D. Nadai. Efficient training of visual transformers with small datasets. In *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=SCN8UaetXx.

C. Malaviya, P. Ferreira, and A. F. T. Martins. Sparse and constrained attention for neural machine translation. *arXiv preprint arXiv:1805.08241*, 2018. URL https://arxiv.org/abs/1805.08241.

B. Millidge, T. Salvatori, Y. Song, T. Lukasiewicz, and R. Bogacz. Universal Hopfield networks: A general framework for single-shot associative memory models. In *International Conference on Machine Learning*, 2022. URL https://proceedings.mlr.press/v162/millidge22a.html.

H. Ramsauer, B. Schäfl, J. Lehner, P. Seidl, M. Widrich, L. Gruber, M. Holzleitner, T. Adler, D. Kreil, M. K. Kopp, G. Klambauer, J. Brandstetter, and S. Hochreiter. Hopfield networks is all you need. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=tL89RnzIiCd.

M. E. Sander, J. Puigcerver, J. Djolonga, G. Peyré, and M. Blondel. Fast, differentiable and sparse top-k: A convex analysis perspective. In *International Conference on Machine Learning*, 2023. URL https://proceedings.mlr.press/v202/sander23a.html.

M. J. Todd. On max-k-sums. *Mathematical Programming*, 171:489–517, 2018. doi:10.1007/s10107-017-1201-0.

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*, 2021. URL https://proceedings.mlr.press/v139/touvron21a.html.

A. Trockman and J. Z. Kolter. Mimetic initialization of self-attention layers. In *International Conference on Machine Learning*, 2023. URL https://openreview.net/forum?id=HxN8K1esES.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017. URL https://arxiv.org/abs/1706.03762.

Y. Xie, H. Dai, M. Chen, B. Dai, T. Zhao, H. Zha, W. Wei, and T. Pfister. Differentiable top-k with optimal transport. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://arxiv.org/abs/2002.06504.

A Comparative Discussion to Alternative Approaches

One alternative to retrieving k-nearest memories, for example, would be to retrieve one memory, and then remove that memory and retrieve again. Our method is better than this one in two respects. First, our method is inherently differentiable, while iteratively removing memories is not. Second, a key function of Hopfield networks is the ability to retrieve superpositions of memories, or "in-between" memories, referred to as spin-glass states [Hopfield, 1982] or metastable states [Ramsauer et al., 2021]. The retrieval of superpositions is what makes Hopfield networks suitable for continuous learning tasks. There is no intuitive way to remove a superposition from memory if it was retrieved.

An alternative naive approach requires increasing the dimension of the state space to be $n \times k$ and defining new "memories" which are combinations of the original ones. This approach requires combinatorially many new memories compared to original ones and becomes quickly intractible.

Finally, there is a growing literature on soft top-k operators [Malaviya et al., 2018, Amos et al., 2019, Xie et al., 2020, Sander et al., 2023]. While we adopt the approach of [Amos et al., 2019], an interesting future direction of research would evaluate the performance of alternative soft top-k operators in k-Hopfield layers.

B Mathematical Analysis

Proposition 3. *Let $x \in \mathbb{R}^n$ and suppose $x_{[k]} > x_{[k+1]}$, where $x_{[i]}$ denotes the i-th largest entry of x. Then*

$$\lim_{\beta \to \infty} \mathrm{ssm}(\beta x; k) = \mathrm{topk}(x), \qquad \textit{where } \mathrm{topk}(x)_i = \begin{cases} 1, & \textit{if } x_i \in \{x_{[1]}, \dots, x_{[k]}\} \\ 0, & \textit{otherwise} \end{cases}. \tag{8}$$

Proof. Since the optimization problem is permutation invariant, we assume, without loss of generality, that $x_1 \geq x_2 \geq \cdots \geq x_n$. Following the analysis from [Amos et al., 2019], the Lagrangian for the minimization problem arising from $\mathrm{ksoftmax}$ is given by

$$L(y, \lambda; \beta) = -\beta x^\top y - H_{\mathrm{b}}(y) + \lambda(k - \mathbb{1}_n^\top y).$$

The first-order optimality condition is $\nabla_y L(y^\star, \lambda^\star; \beta) = \mathbb{0}_n$ and yields the conditions $-\beta x_i + \log(y_i^\star/(1-y_i^\star)) - \lambda^\star = 0$ for all $i \in \{1, \dots, n\}$. These conditions can be expressed succinctly as

$$y^\star = \sigma(\beta x + \lambda^\star \mathbb{1}_n), \tag{9}$$

where σ denotes the logistic function $\sigma(z) = 1/(1+e^{-z})$, which acts elementwise. Moreover, to find the optimal dual variable, we substitute (9) into the constraint $\mathbb{1}_n^\top y^\star = k$ to find the condition

$$\sum_{i=1}^n \sigma(\beta x_i + \lambda^\star) = k. \tag{10}$$

Note that the map $\lambda \mapsto \sum_{i=1}^n \sigma(\beta x_i + \lambda)$ is continuous and monotonically increasing so that, if there is an interval $[\underline{\lambda}, \overline{\lambda}]$ such that $\sum_{i=1}^n \sigma(\beta x_i + \underline{\lambda}) \le k$ and $\sum_{i=1}^n \sigma(\beta x_i + \overline{\lambda}) \ge k$, then there exists $\lambda^\star \in [\underline{\lambda}, \overline{\lambda}]$ satisfying (10).

We now aim to show that as $\beta \to \infty$, $y^\star \to \mathrm{topk}(x)$. To this end, let $\varepsilon > 0$ be given. We will prove that there exists $\beta > 0$ sufficiently large such that $y_i^\star \ge 1-\varepsilon$ for $i \in \{1, \dots, k\}$ and $y_i^\star \le \varepsilon$ for $i \in \{k+1, \dots, n\}$[2]. To this end, let $M = \sigma^{-1}(1-\varepsilon), m = \sigma^{-1}(\varepsilon)$ and pick[3]

$$\beta \ge \max\left\{0, \frac{M-m}{x_k - x_{k+1}}, \frac{M - \sigma^{-1}\left(\frac{\varepsilon}{n-k}\right)}{x_k - x_{k+1}}, \frac{\sigma^{-1}(1-\varepsilon/k) - m}{x_k - x_{k+1}}\right\}.$$

Note that (1) for $y_i^\star \ge 1-\varepsilon$ with $i \in \{1, \dots, k\}$, it is sufficient that $\lambda^\star \ge M - \beta x_k$ and (2) for $y_i^\star \le \varepsilon$ with $i \in \{k+1, \dots, n\}$, it is sufficient that $\lambda^\star \le m - \beta x_{k+1}$. In other words, to prove the result, we need to establish that there exists $\lambda^\star \in [M - \beta x_k, m - \beta x_{k+1}]$ satisfying the condition (10). First note that the interval $[M - \beta x_k, m - \beta x_{k+1}]$ is nonempty because $\beta \ge (M-m)/(x_k - x_{k+1})$. To establish the existence of $\lambda^\star$, we leverage the continuity and monotonicity of the map $\lambda \mapsto \sum_{i=1}^n \sigma(\beta x_i + \lambda)$. First, we evaluate this map at the left endpoint of our interval to obtain

$$\begin{aligned}
\sum_{i=1}^n \sigma(\beta x_i + M - \beta x_k) &= \sum_{i=1}^{k-1} \sigma(\beta(x_i - x_k) + M) + \sigma(M) + \sum_{i=k+1}^n \sigma(\beta(x_i - x_k) + M) \\
&\overset{(a)}{\le} k - \varepsilon + \sum_{i=k+1}^n \sigma(\beta(x_{k+1} - x_k) + M) \\
&= k - \varepsilon + (n-k)\sigma(\beta(x_{k+1} - x_k) + M) \\
&\overset{(b)}{\le} k - \varepsilon + \varepsilon = k,
\end{aligned}$$

where inequality (a) holds because $\sigma(z) \le 1$ for all z, because $\sigma(M) = 1-\varepsilon$, and because $\beta(x_i - x_k) \le \beta(x_{k+1} - x_k)$ for all $i \in \{k+1, \dots, n\}$ since $\beta \ge 0$. Inequality (b) holds because $\beta \ge (M - \sigma^{-1}(\varepsilon/(n-k)))/(x_k - x_{k+1})$.

Regarding the evaluation of the map at the right endpoint of the interval, we compute

$$\begin{aligned}
\sum_{i=1}^n \sigma(\beta x_i + m - \beta x_{k+1}) &= \sum_{i=1}^{k} \sigma(\beta(x_i - x_{k+1}) + m) + \sigma(m) + \sum_{i=k+2}^n \sigma(\beta(x_i - x_{k+1}) + m) \\
&\overset{(c)}{\ge} \sum_{i=1}^{k} \sigma(\beta(x_k - x_{k+1}) + m) + \varepsilon \\
&= k\sigma(\beta(x_k - x_{k+1}) + m) + \varepsilon \\
&\overset{(d)}{\ge} k\left(1 - \frac{\varepsilon}{k}\right) + \varepsilon = k,
\end{aligned}$$

where inequality (c) holds because $\beta(x_i - x_{k+1}) \ge \beta(x_k - x_{k+1})$ for all $i \in \{1, \dots, k\}$ since $\beta \ge 0$, $\sigma(m) = \varepsilon$, and because $\sigma(z) \ge 0$ for all z. Inequality (d) holds because $\beta \ge \frac{\sigma^{-1}(1-\varepsilon/k) - m}{x_k - x_{k+1}}$.

[2] If $\epsilon \ge 1$, any choice of $\beta \ge 0$ will suffice, so we further assume $\epsilon < 1$.

[3] Note that since σ is the logistic function, σ^{-1} exists on the domain $]0,1[$ and is given by $\sigma^{-1}(x) = \log(x/(1-x))$.

Therefore, we conclude that there exists a $\lambda^\star \in [M - \beta x_k, m - \beta x_{k+1}]$ satisfying condition (10). Since $\lambda^\star \geq M - \beta x_k$, by (9) we conclude that $y_i^\star \geq 1 - \varepsilon$ for $i \in \{1, \ldots, k\}$ and since $\lambda^\star \leq m - \beta x_{k-1}$, we conclude that $y_i^\star \leq \varepsilon$ for $i \in \{k+1, \ldots, n\}$. Since $\varepsilon > 0$ was arbitrarily small, the proof is completed. □

Proposition 4. *For all $x \in \mathbb{R}^n$, $k, j \in \{1, \ldots, n\}$, with $k \geq j$, the following inequality holds elementwise:*

$$\mathrm{ssm}(x; k) \geq \mathrm{ssm}(x; j). \tag{11}$$

Proof. Recall from the proof of Proposition 3 that $\mathrm{ssm}(x; k) =: y^{\star,k}$ is given by $y^{\star,k} = \sigma(x + \lambda^{\star,k})$, where $\lambda^{\star,k}$ is the optimal dual variable and satisfies

$$\sum_{i=1}^{n} \sigma(x_i + \lambda^{\star,k}) = k.$$

Note, however that if $\lambda^{\star,k} \geq \lambda^{\star,j}$, then the result is proved in view of the formulas for $y^{\star,k}$ and $y^{\star,j}$ in view of monotonicity of σ. However, it is straightforward to see that $\lambda^{\star,k} \geq \lambda^{\star,j}$ since the mapping $\lambda \mapsto \sum_{i=1}^{n} \sigma(x_i + \lambda)$ is monotone and $k \geq j$. In other words, for the optimal dual variable to satisfy the constraint, it is necessary that $\lambda^{\star,k} \geq \lambda^{\star,j}$. Thus, the result is proved. □

As a corollary to Proposition 4, we can see that for any $x \in \mathbb{R}^n$, $\mathrm{ksoftmax}(x) \geq 0$, where the inequality is understood elementwise.

C Energy Interpretation of Update Rule

Proposition 5. *For any $k \in \{1, \ldots, n\}$, the following statements hold:*

(i) There exists $V_k : \mathbb{R}^n \to \mathbb{R}$ such that $\nabla V_k(x) = \mathrm{ssm}(x; k)$. Indeed, one such V_k is given by

$$V_k(x) = \int_0^1 x^\top \mathrm{ssm}(tx; k) dt. \tag{12}$$

(ii) The function V_k in equation (12) is convex and its Hessian, $\nabla^2 V_k(x)$, is a Laplacian matrix for all $x \in \mathbb{R}^n$.

Proof. To establish that ssm is the gradient of a potential function V_k, we show that its Jacobian is symmetric. To this end, leveraging the Lagrangian formulation from Proposition 3, we know that

$$\mathrm{ssm}(x; k)_i =: y_i^\star(x, \lambda^\star(x)) = \sigma(x_i + \lambda^\star(x)) \quad \text{and} \quad \sum_{i=1}^{n} \sigma(x_i + \lambda^\star(x)) = k, \tag{13}$$

where $\lambda^\star(x)$ denotes the optimal dual variable for the optimization problem. Computing the partial derivatives for $y_i^\star$ yields

$$\frac{\partial y_i^\star}{\partial x_j}(x, \lambda^\star(x)) = \sigma'(x_i + \lambda^\star(x)) \Big(\delta_{ij} + \frac{\partial \lambda^\star}{\partial x_j}(x) \Big), \tag{14}$$

where δ_{ij} is the Kronecker delta. The evaluation of $\partial \lambda^\star / \partial x_j$ may be computed via the implicit function theorem:

$$\begin{aligned} &\sum_{i=1}^{n} \sigma'(x_i + \lambda^\star(x)) \Big(\delta_{ij} + \frac{\partial \lambda^\star(x)}{\partial x_j} \Big) = 0 \\ &\qquad \iff \qquad \frac{\partial \lambda^\star}{\partial x_j}(x) = -\frac{\sigma'(x_j + \lambda^\star(x))}{\sum_{\ell=1}^{n} \sigma'(x_\ell + \lambda^\star(x))}. \end{aligned}$$

Substituting this expression into the equation (14) yields

$$\frac{\partial y_i^\star}{\partial x_j}(x, \lambda^\star(x)) = -\frac{\sigma'(x_i + \lambda^\star(x)) \sigma'(x_j + \lambda^\star(x))}{\sum_{\ell=1}^{n} \sigma'(x_\ell + \lambda^\star(x))} + \delta_{ij} \sigma'(x_i + \lambda^\star(x_i)). \tag{15}$$

Since $\partial y_i^\star/\partial x_j = \partial y_j^\star/\partial x_i$ for all $i, j \in \{1, \dots, n\}$, we conclude that the Jacobian matrix of $y^\star$ is symmetric. The result is then implied by the Poincaré lemma from differential geometry [Abraham et al., 1988, Theorem 6.4.14]. A direct calculation shows that with V_k defined in (12) satisfies $\nabla V_k(x) = \mathrm{ssm}(x; k)$ using the symmetry of partial derivatives.

To see that V_k is convex, we will establish that the Jacobian of $y^\star(x, \lambda^\star(x))$ is positive semidefinite for all x. For $i \in \{1, \dots, n\}$, we use the shorthand $a_i := \sigma'(x_i + \lambda^\star(x))$ and note that for all x and i, $a_i \in]0, 1/4]$. Then we can see that the diagonal elements of the Jacobian are positive since

$$\frac{\partial y_i^\star}{\partial x_i}(x, \lambda^\star(x)) = -\frac{a_i^2}{\sum_{j=1}^n a_j} + a_i = a_i\Big(1 - a_i / \sum_{j=1}^n a_j\Big) > 0.$$

We can also see that the off-diagonal elements are negative. Computing row sums of the Jacobian yields

$$\begin{aligned}\frac{\partial y_i^\star}{\partial x_i}(x, \lambda^\star(x)) + \sum_{j \neq i} \frac{\partial y_i^\star}{\partial x_j}(x, \lambda^\star(x)) &= a_i - \frac{a_i^2}{\sum_{\ell=1}^n a_\ell} - \sum_{j \neq i} \frac{a_i a_j}{\sum_{\ell=1}^n a_\ell} \\ &= a_i - \frac{a_i}{\sum_{\ell=1}^n a_\ell}(a_i + \sum_{j \neq i} a_j) = a_i - a_i = 0.\end{aligned}$$

Since the diagonal entries of $\frac{\partial y^\star}{\partial x}$ are positive, the off-diagonal entries are negative, and row sums are equal to zero, we conclude that for all x, the Jacobian matrix is a Laplacian matrix. An application of Gershgorin's circle theorem establishes positive semidefiniteness for all x, which proves convexity of V_k. □

With Proposition 5 in hand, we are ready to provide an energy interpretation to the k-Hopfield layer rule (4). To be more specific, since the rule (4) is a map from $\mathbb{R}^n$ to $\mathbb{R}^{n \times k}$, it cannot have an energy function that it is minimizing. Instead, we study the update rule one column at a time as follows:

$$x^+ = \Xi\, \mathrm{ksoftmax}_i(\beta \Xi^\top x), \tag{16}$$

where $i \in \{1, \dots, k\}$ and $\mathrm{ksoftmax}_i : \mathbb{R}^n \to \mathbb{R}^n$ outputs the ith column of $\mathrm{ksoftmax}$. In other word, in one step, the update rule (16) attempts to output the i-th closest memory to x. This update rule defines a map from $\mathbb{R}^n$ to itself. If $i \neq 1$, we consider the function $E : \mathbb{R}^n \to \mathbb{R}^n$ given by

$$E(x) = \frac{1}{2} x^\top x - \frac{1}{\beta}\left(\int_0^1 x^\top (\mathrm{ssm}(t\beta \Xi^\top x; i) - \mathrm{ssm}(t\beta \Xi^\top x; i-1)) dt\right). \tag{17}$$

Then, the update rule (16) is a gradient descent update with unit step size. To see this fact, note that

$$\nabla E(x) = x - \Xi(\mathrm{ssm}(\beta \Xi^\top x; i) - \mathrm{ssm}(\beta \Xi^\top x); i - 1)$$

and note that $\mathrm{ssm}(\beta \Xi^\top x; i) - \mathrm{ssm}(\beta \Xi^\top x; i-1) = \mathrm{ksoftmax}_i(\beta \Xi^\top x)$. A gradient descent algorithm on E with stepsize $\alpha > 0$ is

$$x^+ = x - \alpha \nabla E(x) = (1 - \alpha)x + \alpha \Xi\, \mathrm{ksoftmax}_i(\beta \Xi^\top x).$$

Picking $\alpha = 1$ yields the stated update rule (16).

If instead, $i = 1$, then $\mathrm{ksoftmax}_1(\beta \Xi^\top x) = \mathrm{ssm}(\beta \Xi^\top x; 1)$ and the underlying energy function $E : \mathbb{R}^n \to \mathbb{R}^n$ is given by

$$E(x) = \frac{1}{2} x^\top x - \frac{1}{\beta} \int_0^1 x^\top \mathrm{ssm}(t\beta \Xi^\top x; 1) dt. \tag{18}$$

Although we do not present a closed-form expression for $V_k(x) = \int_0^1 x^\top \mathrm{ssm}(tx; k) dt$, we provide the following interpretation: since $\mathrm{ssm}(x; k)$ is a smooth approximation for the top-k operator, we believe its underlying potential function V_k is a smooth approximation for the k-max sum function

$$f_k(x) = \sum_{i=1}^k x_{[i]},$$

where we recall the notation $x_{[i]}$ denotes the i-th largest element of the vector x. Smooth approximations for k-max sums were studied in [Todd, 2018] and for $k = 1$, one such approximation is the log-sum-exp function, as was studied for MCHNs in [Ramsauer et al., 2021]. We believe that different approximations can provide different architectures for associative memory systems.

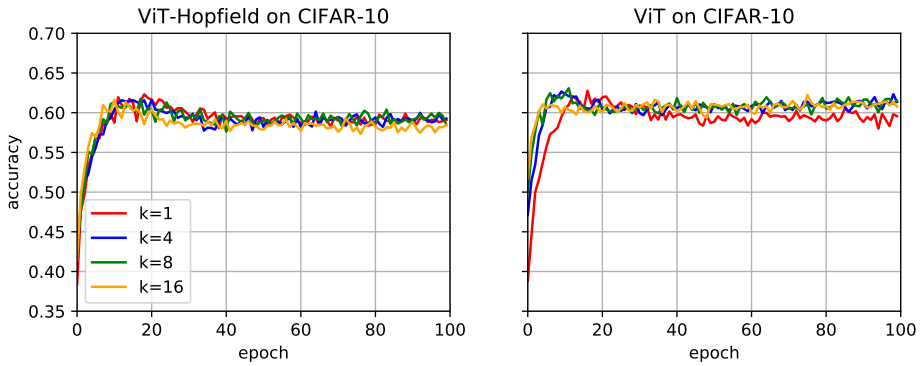

Figure 4: Vision transformer training curves on CIFAR-10 with k-Hopfield attention(ViT-Hopfield) and standard multi-headed attention(ViT). k corresponds to the number of attention heads. We observe that performance of both methods is comparable, even though our method requires fewer parameters.

D More Experiments

We further experiment with the k-Hopfield attention and vision transformer for CIFAR-10. We note that the performance of both models is poor. Vision transformers are known to be difficult to train on small datasets [Liu et al., 2021]. Successfully training ViTs requires techniques such as distillation and smart initialization [Touvron et al., 2021, Trockman and Kolter, 2023]. We do not use these techniques as we are interested in comparison of the attention mechanism.

E Training Details

For vision transformers, we use the ViT class from `https://github.com/lucidrains/vit-pytorch`. We train with patch size of 4, embedding dimension of 128, MLP dimension of 512, and depth of 2. For β, we set it equal to the usual transformer normalization constant $1/\sqrt{d}$, where d is the dimension of the head. We use an Adam optimizer with learning rate of 0.001 and no dropout. For the ssm function, we use a modified version of the implementation in `https://github.com/locuslab/lml`, where we edited the forward pass to use a Newton solver rather than a bracketing method.