Minimum Description Length Hopfield Networks

Matan Abudy[1], Nur Lan[1,2], Emmanuel Chemla[2], Roni Katzir[1]
[1]Tel Aviv University, [2]Ecole Normale Supérieure
`matan.abudy@gmail.com`
`{nur.lan,emmanuel.chemla}@ens.psl.eu`
`rkatzir@tauex.tau.ac.il`

Abstract

Associative memory architectures are designed for memorization but also offer, through their retrieval method, a form of generalization to unseen inputs: stored memories can be seen as prototypes from this point of view. Focusing on Modern Hopfield Networks (MHN), we show that a large memorization capacity undermines the generalization opportunity. We offer a solution to better optimize this tradeoff. It relies on Minimum Description Length (MDL) to determine during training which memories to store, as well as how many of them.

1 Introduction

Generalization requires going beyond the training data. Hopfield Networks (HNs), both classical and modern, involve a retrieval component that can be seen as a form of generalization: an input image that is fed into a trained network can lead to a stored image even if the input itself is quite different from anything that the network saw during training. This form of generalization, however, is limited by the fact that the training stage of the network incentivizes rote memorization.

The present paper explores the generalization capacity of HNs using a very simple setup. Our goals are twofold. We wish to both sharpen the implications of the memorization in HNs and offer a remedy that incorporates generalization into the storage phase. For our first goal, we note that rote memorization in the storage phase can lead to the storage of a wrong number of memories. For our second goal, we show how incorporating a well-studied principle of generalization — specifically, the principle of Minimum Description Length (MDL; Solomonoff 1964, Rissanen 1978) — into the storage phase can help address the problem that we note for rote memorization. MDL balances the complexity of a hypothesis against its fit to the data, and by doing so it leads to hypotheses that are relatively simple, and therefore generalize beyond the data, but not too simple.

We start, in section 2, with relevant background. We assume familiarity with HNs and focus on MDL and its use in the context of neural networks. Section 3 presents MDL HNs. In section 4 we illustrate the problem of incorrect number of memories, and how solving it can lead to good generalization. In section 5 we show how MDL does this. Throughout, we will use modern HNs, but as far as we can tell our essential point holds in Classical HNs and other familiar architectures for associative memory. Section 6 concludes and outlines remaining issues.

2 Background on simplicity and MDL

Consider a hypothesis space $\mathcal{G}$ of possible networks, and a corpus of input data D. In our case, $\mathcal{G}$ is the set of all possible HNs expressible using our representations, and D is a set of input images. For any $G \in \mathcal{G}$, we may consider the ways in which we can encode the data D, given that G. The MDL principle (Rissanen, 1978), a computable approximation of Kolmogorov Complexity (Solomonoff, 1964, Kolmogorov, 1965, Chaitin, 1966), aims at the G that minimizes $|G| + |D : G|$,

where $|G|$ is the size of G and $|D : G|$ is the length of the shortest encoding of D given G (with both components typically measured in bits). Minimizing $|G|$ favors simple, general networks that often fit the data poorly. Minimizing $|D : G|$ favors complex, overly specific networks that overfit the data. By minimizing the sum, MDL aims at an intermediate level of generalization: reasonably simple networks that fit the data reasonably well.

Simplicity criteria have been applied to artificial neural networks, primarily for feed-forward architectures, as early as Hinton and Van Camp [1993], who minimize the encoding length of a network's weights alongside its error, and to Zhang and Mühlenbein [1993, 1995], who use a simplicity metric that is essentially the same as the MDL metric that we use in the present work. Schmidhuber [1997] presents an algorithm for discovering networks that optimize a simplicity metric that is closely related to MDL. Simplicity criteria have been used in a range of works on neural networks, including recent contributions (e.g., Ahmadizar et al., 2015 and Gaier and Ha, 2019). More recently, Lan et al. [2022, 2023] applied MDL to recurrent neural networks (RNNs) and showed that this allows the networks to find provably correct solutions to a range of tasks that are only approximated using standard methods.

Outside of neural networks, MDL — and the closely related Bayesian approach to induction — have been used in a wide range of tasks in which one is required to generalize from very limited data (see Horning, 1969, Berwick, 1982, Stolcke, 1994, Grünwald, 1996, and de Marcken, 1996, and more recently Rasin and Katzir, 2016 and Rasin et al., 2021.

3 MDL Hopfield Networks

Implementing MDL within a given formalism involves specifying how to measure both $|G|$, the encoding length of a hypothesis G, and $|D : G|$, the encoding length of the training data D given the hypothesis G. Moreover, in order to turn MDL into a practical objective function, we should specify how we search the hypothesis space for the choice that optimizes $|G| + |D : G|$. We describe our choices for each of these three decision points below, and discuss more possibilities in section 6. [1]

3.1 Encoding G

MHN can be described as a set of patterns stored in memory slots. An encoding of these memories is thus an encoding of the 'grammar' G (which itself is the basis for an encoding of any data point). To encode such a set of patterns, and get $|G|$ for the MDL objective, one needs to encode the pixel values of these patterns, and concatenate these encodings. Here we used two simple encoding schemes for pixel values in $[0, 1]$ (which produced similar results, that we will therefore not distinguish). **Fixed-Length Encoding:** We encode each pixel value of each pattern with a constant number of bits, say one, resulting in a total encoding length equal to the number of patterns multiplied by the pattern size. **Prefer-Extreme-Value Encoding:** This encoding favors extreme values: we assign one bit to encode 0 and 1, while the numbers in between are encoded with more bits, with the middle point 0.5 receiving the most complex encoding. For concreteness we chose $[1 + 10 \cdot (value \cdot (1 - value))]^2$.

3.2 Encoding D:G

To encode the data according to the given grammar, we need to specify how each training exemplar can be reconstructed using the grammar (network). For each training exemplar, we first need to encode which memory we should start from to reconstruct it. To do this, we encode the index of the closest memory in the memory slots. We need $\lceil \log_2 \text{num\_memory\_slots} \rceil$ bits for this encoding.

Next we need to encode the differences between the exemplar and its associated memory. We used the L1-distance between the memory and the training image, multiplied by $\lceil \log_2 \text{pattern\_size} \rceil$. If all images are bitmaps, then this amounts to encoding where the non-null differences are.

3.3 Training

In order to search the hypothesis space for the network that optimizes the MDL objective function we use Simulated Annealing (SA; Kirkpatrick et al. 1983). Starting from an initial hypothesis, SA repeatedly considers switching to a random neighbor, depending on (a) how the neighbor compares to

---

[1]The source code used in this paper is available at `https://github.com/matanabudy/mdl-hn`.

the current hypothesis, and (b) a temperature parameter. If the neighbor is better (in our case, lower $|G| + |D : G|$), SA switches to it. Otherwise, the worse the neighbor the less likely SA is to switch to it. The temperature parameter determines how risk taking SA is at a given moment: the higher the temperature, the likelier the switch to a worse neighbor. During the search the temperature is slowly lowered, and when it is close to zero the search is effectively greedy.

A random neighbor is derived from the current hypothesis using one of the following operations: 1. Removing a random memory, 2. Adding a random training set image to the memories, 3. Changing an existing memory: Selecting a random memory and applying Gaussian noise to random indices in it, 4. Crossover: Creating a new memory by averaging two random memories, replacing the originals.

4 Experiment 1: The risk with excessive memory in Hopfield Networks

Much work in the literature has been dedicated to equip Hopfield networks with a large memory capacity. Here we illustrate why the use of these large memory resources should be constrained.

Data. We created a 9×9 bitmap of each digit. For each of these 'golden' bitmaps, we created several noisy, continuous exemplars of it: each pixel was incremented with Gaussian noise with low or high variance, and the result was clipped to $[0, 1]$. Noisy, discrete exemplars were also created, by rounding the continuous exemplars to 0 or 1. **Setup.** HNs were implemented with a version of the HAMUX library [Hoover et al., 2022], updated in `https://github.com/bhoov/eqx-hamux`.

Experimental conditions. The different training sets used to train the networks were defined by: the number of golden digits represented (1 digit, 2 digits, ..., all 10 digits), the number of exemplars per digit (1, 5, 10, 30), their type (discrete, continuous), their level of noise (low or medium variance). Additionally, the networks were equipped with a memory capacity either corresponding to the cardinality of the training set (unconstrained), or to the cardinality of *golden* digits represented in the training set (golden memories).

Results and discussion. Unconstrained MHNs very closely memorized the full input: in the examples inspected, unsurprisingly, memories retrieved from exemplars were diverse, each capturing few exemplars, with distances between an exemplar and its retrieved memory being low, potentially lower than to the underlying golden digit. On the other hand, with MHNs constrained *a priori* to use no more than the golden number of memories, the learned memories were close to the actual golden digits. (Results became similar to those in Fig. 2). MHNs are thus capable of excellent generalization in principle, but only if the memory resources are kept under control. The next experiment shows how to achieve this in a non-supervised manner (i.e. non-*ad-hoc*), through the MDL objective.

5 Experiment 2: MDL delivers the golden memories (unsupervised)

Setup. The Minimum Description Length (MDL) principle was incorporated as described in section 3 above to train MHNs, on the same training sets described for Exp. 1.

Results. Fig. 1 shows the outcome of one simulation. We see that the memories learned through MDL are identical to the target 'golden' digits, aside from two grey pixels in digits 8 and 9. This illustrates the two main results. First, the final number of memory slots (which is now adjusted during training), aligns well with the golden number of digits in the training set, see Fig. 2 and Fig. 4 in Appx. A for continuous noise). This holds strictly true in most cases, except most notably for the high noise conditions, with many exemplars for each digit – conditions under which an image may be sampled very far from its golden digit, which may reasonably justify the creation of a

Figure 1: Memory slots learned through MDL training (top row) and exemplars that were matched to each memory.

new category. We come back to these issues in the follow-up below. Second, the learned memories strongly resemble the golden digits: the golden digits were at an average L2 distance of 0.449 for the low noise simulations (1.181 for medium noise) from their associated memories.
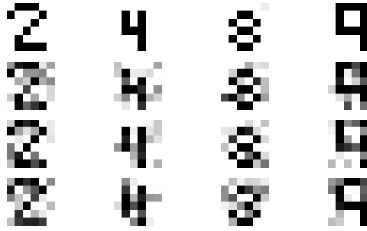
Follow-up experiment. We conjectured that some deviance from the correct number of memory slots (Fig. 2) may actually be legitimate. A noisy version of a digit could be so far from its source that it would justify creating a new category. As a conservative criterion, we thus consider that an exemplar that ends up closest to a digit that is not its source digit is too noisy. Indeed, it could lead to a new category and an overestimation of categories. Or it could lead to an underestimation, since a golden digit would become redundant as a category if all its exemplars end up closer to another digit. We therefore re-ran the Exp .2 with a new dataset, in which we disregarded such exemplars. We found that MDLHNs were even better than before at aligning their number of memories with the (now, more meaningful) golden number, see Fig. 3 and Fig. 5 in Appx. A.
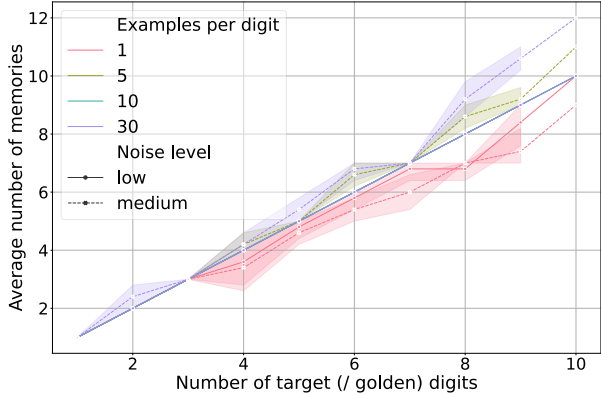

Figure 2: Average number of memory slots found through MDL training on discrete noisy exemplars, compared to the number of golden digits.

6 Discussion

We noted that generalization in HNs is limited to the retrieval stage, while storage amounts to rote memorization. We illustrated the problem that this limitation poses when the training data are imperfect, including variants, possibly noisy, of a hidden number of archetypes. In such cases, standard HNs memorize the surface, noisy versions and do not recover the intuitively correct underlying collection of images. We showed how this problem can be remedied by incorporating the generalization principle of MDL into the storage phase: by considering the encoding length of the network (the lower the encoding length, the less capable it is to memorize the data) as a counterweight to the fit to the data, MDLHNs overcame the challenge of generalization and recovered both the correct number of archetypes and the archetypes themselves.

Our proposal is robust to architecture changes. While we illustrated the use of MDL with MHNs, MDL can just as easily be incorporated into classical HNs or other architectures for associative memory. Indeed, as mentioned in section 2, MDL has been explored for sequential neural network architectures as well as for entirely different formalisms such as those of formal language theory, and it has also been argued to underlie various mechanisms in human cognition. In this regard, our solution is a principled and very general one to the problem that we noted.

While general, MDL is very tightly connected to representations, which highlights interesting future directions for MDLHNs. Our present illustration relied on simple encoding schemes both for $|G|$ and for $|D : G|$. For both, however, one can consider more sophisticated choices. For instance, in a more realistic setting than ours, the system might have a preference for encoding certain values over others. And for $|D : G|$, our distance function was a convenient choice for the illustration, but it is one that is suitable for static noise that flips random bits. It is not designed to capture, for example, translation, rotation, stretches, reflection, or other operators that are arguably needed for a system that captures human intuitions about visual similarity. Incorporating such operators into $|D : G|$ would be an interesting next step. One simple way to illustrate the relevance of better representations is this. Using our current, naive encodings, a network trained on a single image will have no reason not to memorize that image. With more sophisticated primitives, on the other hand, a reconstruction of an archetype may well be preferable even for a single image.

Our focus in the present paper has been entirely on the objective function for HNs. For the optimization of the storage phase we simply adopted a general search algorithm (specifically, SA), which supported the recovery of memories in our examples. For setups that are more complex than the one used here, this might prove inadequate, and other optimization algorithms may be more helpful. Lan et al. [2022], for example, report using a genetic algorithm for optimizing MDL in the domain of RNNs. And other choices might be more helpful still.

References

[1] Ray J. Solomonoff. A formal theory of inductive inference, parts I and II. *Information and Control*, 7(1 & 2):1–22, 224–254, 1964.

[2] Jorma Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.

[3] Andrei Nikolaevic Kolmogorov. Three approaches to the quantitative definition of information. *Problems of Information Transmission (Problemy Peredachi Informatsii)*, 1:1–7, 1965.

[4] Gregory J. Chaitin. On the length of programs for computing finite binary sequences. *Journal of the ACM*, 13:547–569, 1966.

[5] Geoffrey E. Hinton and Drew Van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the sixth annual conference on Computational learning theory*, pages 5–13, 1993.

[6] Byoung-Tak Zhang and Heinz Mühlenbein. Evolving optimal neural networks using genetic algorithms with Occam's Razor. *Complex Systems*, 7(3):199–220, 1993.

[7] Byoung-Tak Zhang and Heinz Mühlenbein. Balancing accuracy and parsimony in genetic programming. *Evolutionary Computation*, 3(1):17–38, 2020/07/11 1995.

[8] Jürgen Schmidhuber. Discovering neural nets with low Kolmogorov complexity and high generalization capability. *Neural Networks*, 10(5):857–873, 1997.

[9] Fardin Ahmadizar, Khabat Soltanian, Fardin AkhlaghianTab, and Ioannis Tsoulos. Artificial neural network development by means of a novel combination of grammatical evolution and genetic algorithm. *Engineering Applications of Artificial Intelligence*, 39:1–13, 2015.

[10] Adam Gaier and David Ha. Weight agnostic neural networks. *CoRR*, abs/1906.04358, 2019.

[11] Nur Lan, Michal Geyer, Emmanuel Chemla, and Roni Katzir. Minimum description length recurrent neural networks. *Transactions of the Association for Computational Linguistics*, 10: 785–799, 8/2/2022 2022.

[12] Nur Lan, Emmanuel Chemla, and Roni Katzir. Benchmarking neural network generalization for grammar induction. In *Proceedings of the 2023 CLASP Conference on Learning with Small Data (LSD)*, pages 131–140, Gothenburg, Sweden, September 2023. Association for Computational Linguistics.

[13] James Horning. *A Study of Grammatical Inference*. PhD thesis, Stanford, 1969.

[14] Robert C. Berwick. *Locality Principles and the Acquisition of Syntactic Knowledge*. PhD thesis, MIT, Cambridge, MA, 1982.

[15] Andreas Stolcke. *Bayesian Learning of Probabilistic Language Models*. PhD thesis, University of California at Berkeley, Berkeley, California, 1994.

[16] Peter Grünwald. A minimum description length approach to grammar inference. In Stefan Wermter, Ellen Riloff, and Gabriele Scheler, editors, *Connectionist, Statistical and Symbolic Approaches to Learning for Natural Language Processing*, Springer Lecture Notes in Artificial Intelligence, pages 203–216. Springer, 1996.

[17] Carl de Marcken. *Unsupervised Language Acquisition*. PhD thesis, MIT, Cambridge, MA, 1996.

[18] Ezer Rasin and Roni Katzir. On evaluation metrics in Optimality Theory. *Linguistic Inquiry*, 47 (2):235–282, 2016.

[19] Ezer Rasin, Iddo Berger, Nur Lan, Itamar Shefi, and Roni Katzir. Approaching explanatory adequacy in phonology using Minimum Description Length. *Journal of Language Modelling*, 9 (1):17–66, 2021.

[20] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by Simulated Annealing. *Science*, 220(4598):671–680, May 1983. ISSN 0036-8075, 1095-9203. doi: 10.1126/science.220.4598. 671. URL `https://www.science.org/doi/10.1126/science.220.4598.671`.

[21] Benjamin Hoover, Duen Horng Chau, Hendrik Strobelt, and Dmitry Krotov. A universal abstraction for hierarchical hopfield networks. In *The Symbiosis of Deep Learning and Differential Equations II*, 2022. URL `https://openreview.net/forum?id=SAv3nhzNWhw`.
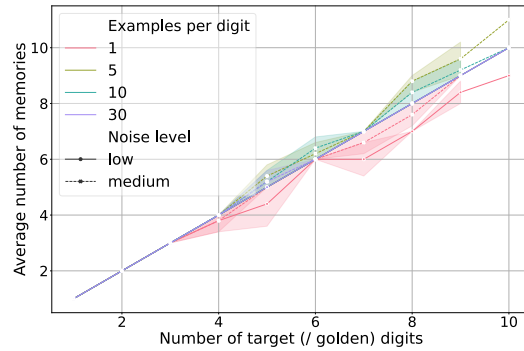
A   Supplementary Figures

Figure 3: Follow-up to Exp. 1 - average number of memory slots found using MDL, compared to the number of target 'golden' digits in the discrete noise training set, while making sure noised exemplars aren't ambiguous between classes.
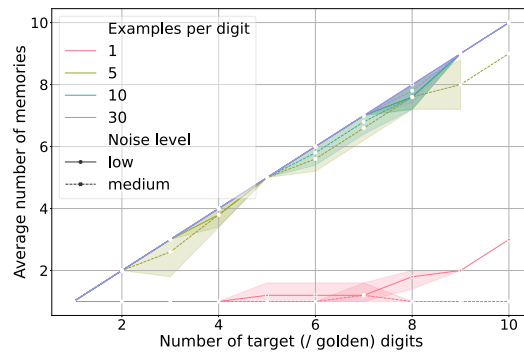

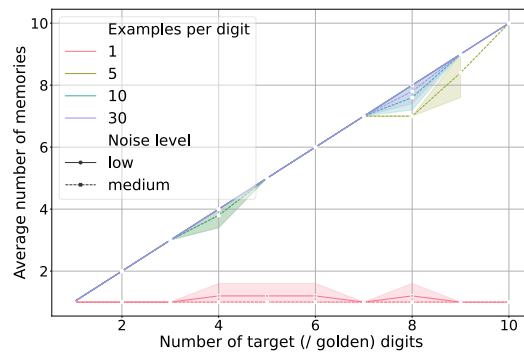Figure 4: Running Exp. 1 with continuous noise exemplars in the training set

Figure 5: Follow-up experiment for the continuous noise training set