Meta-Learning Deep Energy-Based Memory Models

Sergey Bartunov, Jack W Rae, Simon Osindero, Timothy P Lillicrap
DeepMind
London, United Kingdom
{bartunov, jwrae, osindero, countzero}@google.com

Abstract

Attractor networks provide a sound model of associative memory. In such models the dynamics are often implemented as an optimization procedure that minimizes an energy function, such as in the classical Hopfield network. In general it is difficult to derive a writing rule for a given dynamics and energy that is both compressive and fast. Thus, most research in energy-based memory has been limited either to tractable energy models not expressive enough to handle complex high-dimensional objects such as natural images, or to models that do not offer fast writing. We present a novel meta-learning approach to energy-based memory models (EBMM) that allows to use an arbitrary neural architecture as an energy model and quickly store patterns in its weights. We demonstrate experimentally that our EBMM approach can build compressed memories for synthetic and natural data, and is capable of associative retrieval that outperforms existing memory systems in terms of the reconstruction error and compression rate.

1 Introduction

Associative memory has long been of interest to neuroscience and machine learning communities [27, 15, 16]. This interest has generated many proposals for associative memory models, both biological and synthetic. These models address the problem of storing a set of patterns in such a way that a stored pattern can be retrieved based on a partially known or distorted version. This kind of retrieval from memory is known as auto-association.

Attractor networks provide one well-grounded foundation for associative memory models [1]. Patterns are stored in such a way that they become attractors of the update dynamics defined by the network. Then, if a query pattern that preserves sufficient information for association lies in the basin of attraction for the original stored pattern, a trajectory initialized by the query will converge to the stored pattern.

A variety of implementations of the general attractor principle have been proposed. The classical Hopfield network [15], for example, defines a simple quadratic energy function whose parameters serve as a memory. The update dynamics in Hopfield networks iteratively minimize the energy until it converges to a minima. The goal of the writing process is to find parameter values such that the stored patterns become attractors for the optimization process and such that, ideally, no spurious attractors are created. Unfortunately, simple Hopfield memory models have two fundamental limitations: (1) It is not possible to add capacity for more stored patterns by increasing the number of parameters since the number of parameters in a Hopfield network is quadratic in the dimensionality of the patterns. (2) The model lacks a means of modelling the higher-order dependencies that exist in real-world data.

A variety of energy-based memory models have been proposed since the original Hopfield network to mitigate its limitations [13, 7]. Restricted Boltzmann Machines (RBMs) [12] add capacity to the model by introducing latent variables, and deep variants of RBMs [13, 24] afford more expressive energy functions. Unfortunately, training Boltzmann machines remains challenging, and while recent

3rd Workshop on Meta-Learning at NeurIPS 2019, Vancouver, Canada.

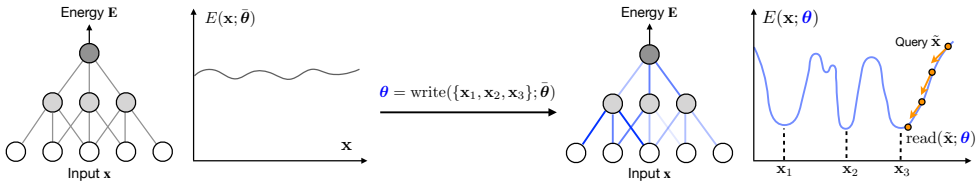

Figure 1: A schematic illustration of EBMM. The energy function is modelled by a neural network. The writing rule is then implemented as a weight update, producing parameters $\boldsymbol{\theta}$ from the initialization $\bar{\boldsymbol{\theta}}$, such that the stored patterns $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$ become local minima of the energy (see section 3). Local minima are attractors for gradient descent which implements associative retrieval starting from a query $\tilde{\mathbf{x}}$, in this case a distorted version of $\mathbf{x}_3$.

probabilistic models such as variational auto-encoders [17, 23] are easier to train, they nevertheless pay the price for expressivity in the form of slow writing. Another class of models rely on more conventional, slot-based memory architectures [26, 9, 28]. Representations stored in the slots make use of deep non-linear encoders, but since memories are only expressed as linear combinations of the slots, compression capabilities of such models are limited. Finally, hybrid approaches such as [4, 20] combine Hopfield-like memory representations with deep encoders which naturally enables fast writing, but, as we show experimentally, does not allow strong compression strategies to emerge.

In this paper we propose a novel approach that leverages recent developments in meta-learning to enable fast storage of patterns into the weights of arbitrarily structured neural networks. Relying on the gradient-based reading dynamics, we meta-learn a writing rule in the form of truncated gradient descent over the parameters defining the energy function. We show that the proposed approach enables efficient utilization of network weights and high compression, as well as fast-convergent attractor dynamics.

2 Retrieval in energy-based models

We focus on deterministic, fixed-point attractor networks as a basis for associative memory. Such networks define update dynamics for iterative evolution of the input pattern: $\mathbf{x}^{(k+1)} = f(\mathbf{x}^{(k)})$, where a fixed-point attractor is a point $\mathbf{x}$ for which the dynamics converge, i.e. $\mathbf{x} = f(\mathbf{x})$. Learning the associative memory is then equivalent to learning the dynamics f such that its fixed-point attractors are the stored patterns and the corresponding basins of attraction are sufficiently wide for retrieval.

An energy-based attractor network is defined by the energy function $E(\mathbf{x})$ mapping an input object $\mathbf{x} \in \mathcal{X}$ to a real scalar value. A particular model may then impose additional requirements on the energy function. In this paper we consider a class of energy functions that are differentiable on $\mathcal{X} \subseteq \mathbb{R}^d$, bounded from below and rely on gradient optimization as attractor dynamics. More precisely, we define the update dynamics over $k = 1, \ldots, K$ steps via gradient descent:

$$\textbf{read}(\tilde{\mathbf{x}}; \boldsymbol{\theta}) = \mathbf{x}^{(K)}, \quad \mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \gamma^{(k)} \nabla_{\mathbf{x}} E(\mathbf{x}^{(k)}), \quad \mathbf{x}^{(0)} = \tilde{\mathbf{x}}. \tag{1}$$

With appropriately set step sizes $\{\gamma^{(k)}\}_{k=0}^{K}$ this procedure asymptotically converges to a local minimum of energy $E(\mathbf{x})$ [21]. Since asymptotic convergence may be not enough for practical applications, we truncate the optimization procedure (1) at K steps and treat $\mathbf{x}^{(K)}$ as a result of the retrieval. While vanilla gradient descent (1) is sufficient to implement retrieval, in our experiments we employ a number of extensions, such as the use of Nesterov momentum and projected gradients, which are thoroughly described in Appendix A.

Relying on the generic optimization procedure allows us to translate the problem of designing update dynamics with desirable properties to constructing an appropriate energy function, which in general is equally difficult. In the next section we discuss how to tackle this difficulty.

3 Meta-learning gradient-based writing rules

For the energy model we assume a parametric family $E(\mathbf{x}; \boldsymbol{\theta})$ differentiable in both $\mathbf{x}$ and $\boldsymbol{\theta}$, and bounded from below as a function of $\mathbf{x}$. These are mild assumptions that are often met in the existing

Table 1: Number of error bits in retrieved binary patterns.

METHOD \ # PATTERNS	16	32	48	64	96
HOPFIELD NETWORK, HEBB RULE	0.4	5.0	9.8	13.0	16.5
HOPFIELD NETWORK, STORKEY RULE	0.0	0.9	6.3	11.3	17.1
HOPFIELD NETWORK, PSEUDO-INVERSE RULE	0.0	0.0	0.3	4.3	22.5
DIFFERENTIABLE PLASTICITY [20]	3.0	13.2	20.8	26.3	34.9
MANN [25]	0.1	0.2	1.8	4.25	9.6
LSTM [14]	30	58	63	64	64
MEMORY NETWORKS [26]	**0.0**	**0.0**	**0.0**	**0.0**	10.5
EBMM (ours)	**0.0**	**0.0**	**0.1**	0.5	**4.2**

neural architectures with an appropriate choice of activation functions, e.g. tanh. The writing rule then compresses input patterns $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ into parameters $\boldsymbol{\theta}$ such that each of the stored patterns becomes a local minimum of $E(\mathbf{x}; \boldsymbol{\theta})$ or, equivalently, creates a basin of attraction for gradient descent in the pattern space. This property can be practically quantified by the reconstruction error, e.g. mean squared error, between the stored pattern $\mathbf{x}$ and the one retrieved:

$$\mathcal{L}(X, \boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}\left[||\mathbf{x}_i - \textbf{read}(\tilde{\mathbf{x}}_i; \boldsymbol{\theta})||_2^2\right]. \tag{2}$$

Here we assume a known distortion model $\tilde{\cdot}$ such as randomly erasing certain number of dimensions, salt and pepper noise etc. While one can call minimization of this loss with a conventional optimization method a writing rule, it will require too many optimization steps to obtain a satisfactory solution and thus will not be applicable in many memory applications [25].

Hence, we explore a different approach to designing a *fast* writing rule (i.e. one that requires *few parameter update iterations*) inspired by recently proposed gradient-based meta-learning techniques [2, 8] which we call meta-learning energy-based memory models (EBMM). The straightforward application of gradient-based meta-learning to the loss (2) is problematic due to combinatorial number of possible distortions. Instead, we define a different *writing loss* $\mathcal{W}$, minimizing which serves as a proxy for ensuring that input patters are local minima for the energy:

$$\mathcal{W}(\mathbf{x}, \boldsymbol{\theta}) = E(\mathbf{x}; \boldsymbol{\theta}) + \alpha ||\nabla_x E(\mathbf{x}; \boldsymbol{\theta})||_2^2 + \beta ||\boldsymbol{\theta} - \bar{\boldsymbol{\theta}}||_2^2. \tag{3}$$

The writing loss (3) consists of three terms. The first term is simply the energy value which we would like to be small for stored patterns relative to non-stored patterns. The condition for $\mathbf{x}$ to be a local minimum of $E(\mathbf{x}; \boldsymbol{\theta})$ is two-fold: first, the gradient at $\mathbf{x}$ is zero, which is captured by the second term of the writing loss, and, second, the hessian is positive-definite. The latter condition is difficult to express in a form that it admits efficient optimization and we found that meta-learning using just first two terms in the writing loss is sufficient. Finally, the third term limits deviation from initial parameters $\bar{\boldsymbol{\theta}}$ which we found helpful from optimization perspective.

We use truncated gradient descent on the writing loss (3) to implement the writing rule:

$$\textbf{write}(X) = \boldsymbol{\theta}^{(T)}, \quad \boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta^{(t)} \frac{1}{N} \sum_{i=1}^{N} \nabla_{\boldsymbol{\theta}} \mathcal{W}(\mathbf{x}_i, \boldsymbol{\theta}^{(t)}), \quad \boldsymbol{\theta}^{(0)} = \bar{\boldsymbol{\theta}} \tag{4}$$

To ensure that gradient updates (4) are useful for minimization of the reconstruction error (2) we train the combination of retrieval and writing rules end-to-end, meta-learning initial parameters $\bar{\boldsymbol{\theta}}$, learning rate schedules $\mathbf{r} = (\{\gamma^{(k)}\}_{k=1}^{K}, \{\eta^{(t)}\}_{t=1}^{T})$ and meta-parameters $\tau = (\alpha, \beta)$ to perform well on random sets of patterns from the known data distribution $p_d(X)$:

$$\text{minimize } \mathbb{E}_{X \sim p_d(X)} \left[\mathcal{L}(X, \textbf{write}(X))\right] \text{ for } \bar{\boldsymbol{\theta}}, \mathbf{r}, \boldsymbol{\tau}. \tag{5}$$

While truncated gradient descent is not guaranteed to converge, the reading and writing rules are trained jointly to minimize the reconstruction error (2) and thus ensure that they converge *sufficiently* fast. This property turns this potential drawback of the method to its advantage over provably convergent, but slow models. It also relaxes the necessity of stored patterns to create too well-behaved basins of attraction. If, for example, a stored pattern creates a nuisance attractor in the dangerous proximity of the main one, the gradient descent (1) might successfully pass it with appropriately learned step sizes.

4 Experiments

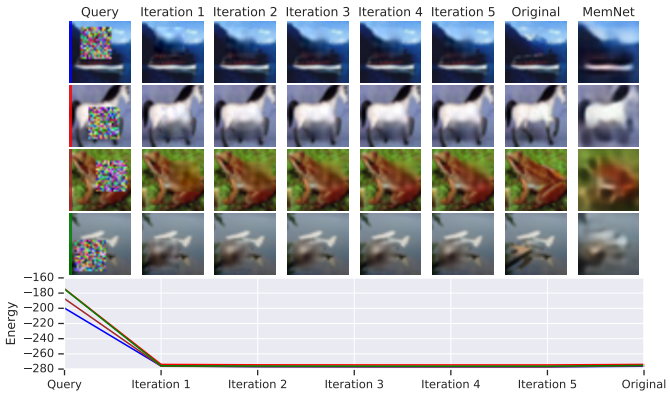

Figure 2: Visualization of gradient descent iterations during retrieval of CIFAR images. The last column contains reconstructions from Memory networks (both models use 10k memory).

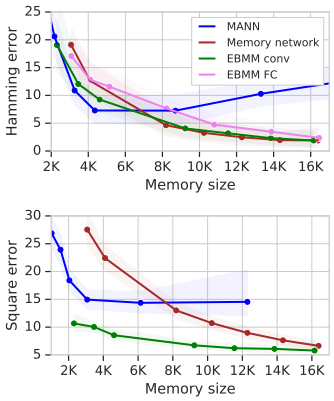

Figure 3: Distortion (reconstruction error) vs rate (memory size) analysis on batches of 64 images. Top: Omniglot, bottom: CIFAR.

We evaluate our approach on different kinds of data ranging from random binary 128-bit patterns (table 1) to binary images of hand-written characters [18] to natural images from CIFAR dataset (figure 3). In each case we corrupted a random part of the pattern (64 bits for the random binary patterns and 16×16 block for images) and let the model perform association of the original pattern.

To compare EBMM to existing memory architectures we selected a number of relevant baselines models: Hopfield network using different learning rules, Differentiable Plasticity model [20], Memory-Augmented Neural Network (MANN) [25], LSTM [14] and Memory Networks [26]. All models shared the same encoder (and decoder, when applicable) network. We explored two versions of EBMM that use parts of fully-connected (FC) and convolutional (conv) layers in a 3-block ResNet [11] as writable memory. The proportion of writable weights in the layers can be varied, thus allowing to control the memory footprint. All network architectures are described in Appendix B

Our primary interest in this study is how well a memory model can retrieve a remembered pattern for a given memory size. We measure the memory size as a number of `float32` numbers used to represent a hidden state (as in LSTM), a hebbian trace (as in Differentiable plasticity) or dynamic weights (as in our model). In the binary experiment we fixed capacity of all models to $128 * (128 - 1)/2 + 128$ to allow fair comparison with Hopfield networks which have memory size strongly tied to the input dimensionality. We then varied the number of patterns written to this fixed volume. In all other experiments we varied the memory size and measured retrieval of a 64-images batch.

As one can see, EBMM performed well in all experiments, demonstrating the most significant advantage in low-capacity, compressive regimes. Kanerva machine and Differentiable plasticity models effectively failed to learn on the more complex image tasks, possible because of the strong noise and long training sequences. Memory networks appeared to be the most competitive baseline, but only when given enough memory to adequately represent high-dimensional objects.

5 Conclusion

We introduced a novel method for learning deep associative memory systems. Our method benefits from the recent progress in deep learning so that we can use a very large class of neural networks both for learning representations and for storing patterns in network weights. At the same time, we are not bound by slow gradient learning thanks to meta-learning of fast writing rules. We showed that our method is applicable in a variety of domains from non-compressible (binary strings) to highly compressible (natural images) and that the resulting memory uses available capacity efficiently. We believe that more elaborated architecture search will lead to stronger results on par with state-of-the-art generative models.

References

[1] Daniel J Amit and Daniel J Amit. *Modeling brain function: The world of attractor neural networks*. Cambridge university press, 1992.

[2] Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems*, pages 3981–3989, 2016.

[3] Antreas Antoniou, Harrison Edwards, and Amos Storkey. How to train your maml. *arXiv preprint arXiv:1810.09502*, 2018.

[4] Jimmy Ba, Geoffrey E Hinton, Volodymyr Mnih, Joel Z Leibo, and Catalin Ionescu. Using fast weights to attend to the recent past. In *Advances in Neural Information Processing Systems*, pages 4331–4339, 2016.

[5] David Belanger, Bishan Yang, and Andrew McCallum. End-to-end learning for structured prediction energy networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 429–439. JMLR. org, 2017.

[6] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[7] Yilun Du and Igor Mordatch. Implicit generation and generalization in energy-based models. *arXiv preprint arXiv:1903.08689*, 2019.

[8] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*, 2017.

[9] Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, et al. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538 (7626):471, 2016.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[12] Geoffrey E Hinton. A practical guide to training restricted boltzmann machines. In *Neural networks: Tricks of the trade*, pages 599–619. Springer, 2012.

[13] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[14] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

[15] John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

[16] Pentti Kanerva. *Sparse distributed memory*. MIT press, 1988.

[17] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[18] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

[19] Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. *arXiv preprint arXiv:1711.05101*, 2017.

[20] Thomas Miconi, Jeff Clune, and Kenneth O Stanley. Differentiable plasticity: training plastic neural networks with backpropagation. *arXiv preprint arXiv:1804.02464*, 2018.

[21] Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013.

[22] Yurii E Nesterov. A method for solving the convex programming problem with convergence rate o (1/kˆ 2). In *Dokl. akad. nauk Sssr*, volume 269, pages 543–547, 1983.

[23] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

[24] Ruslan Salakhutdinov and Hugo Larochelle. Efficient learning of deep boltzmann machines. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 693–700, 2010.

[25] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pages 1842–1850, 2016.

[26] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. *arXiv preprint arXiv:1410.3916*, 2014.

[27] David J Willshaw, O Peter Buneman, and Hugh Christopher Longuet-Higgins. Non-holographic associative memory. *Nature*, 222(5197):960, 1969.

[28] Yan Wu, Gregory Wayne, Karol Gregor, and Timothy Lillicrap. Learning attractor dynamics for generative memory. In *Advances in Neural Information Processing Systems*, pages 9401–9410, 2018.