ATLAS: Universal Function Approximator for Memory Retention

Anonymous Author(s)
Affiliation
Address
email

Abstract

Artificial neural networks (ANNs), despite their universal function approximation capability and practical success, are subject to catastrophic forgetting. Catastrophic forgetting refers to the abrupt unlearning of a previous task when a new task is learned. It is an emergent phenomenon that plagues ANNs and hinders continual learning. Existing universal function approximation theorems for ANNs guarantee function approximation ability, but seldom touch on the model details and do not predict catastrophic forgetting. This paper presents a novel universal approximation theorem for multi-variable functions using only single-variable functions and exponential functions. Furthermore, we present *ATLAS*—a novel ANN architecture based on the exponential approximation theorem and B-splines. It is shown that ATLAS is a universal function approximator capable of memory retention and, therefore, continual learning. The memory retention of ATLAS is imperfect, with some off-target effects during continual learning, but it is well-behaved and predictable. An efficient implementation of ATLAS is provided. Experiments are conducted to evaluate both the function approximation and memory retention capabilities of ATLAS.

1 Introduction

Catastrophic forgetting [7, 13, 22] is an emergent phenomenon where an ANN learns a new task, and the subsequent parameter updates interfere with the model's performance on previously learned tasks. Catastrophic forgetting is also called catastrophic interference [19]. If an ANN cannot effectively learn many tasks, it has limited utility in the context of continual learning [9, 12]. Catastrophic forgetting is like learning to pick up a cup, but simultaneously forgetting how to breathe. Even linear functions are susceptible to catastrophic forgetting, as illustrated in Figure 1

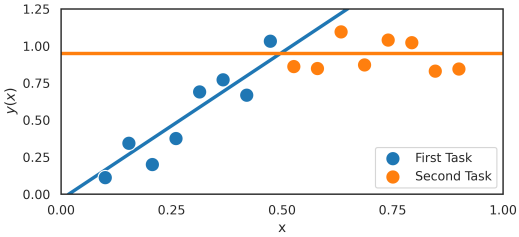


Figure 1: A linear function is susceptible to catastrophic forgetting.

The simple example of a linear regression model being susceptible to catastrophic forgetting might be due to the non-linearity of the target function, noise in the data, or parameter sharing across the

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

input. Avoiding parameter sharing is easily accomplsihed with piece-wise defined functions such as splines [26].

Universal function approximation theorems are a cornerstone of machine learning, and prove that artificial neural networks (ANNs) can approximate any given continuous target function [10, 11, 15]. The theorems do not specify how to find an ANN with sufficient performance for problems in practice. Gradient descent optimisation is the standard for training neural networks, but other optimisation and learning procedures exist. The numerical stability of gradient descent methods is of practical importance. Gradient update vectors that are bounded and orthogonal to each other improve numerical stability and can mitigate catastrophic forgetting.

This paper introduces ATLAS—a novel universal function approximator based on B-splines that has intrinsic memory retention. ATLAS has well-behaved parameter gradients that do not vanish or explode. ATLAS has methods to expand model capacity in a systematic way without loosing the previously learned information. The accompanying representation and universal approximation theorems are also provided.

2 Relevant Studies

It is conjectured that overlapping representations in ANNs lead to catastrophic forgetting [12]. Catastrophic forgetting occurs when parameters necessary for one task change while training to meet the objectives of another task [14, 20]. Strategies to mitigate catastrophic forgetting can include shielding important weights from changes, or regularisation to keep weights close to their initial values. The least desirable option is retraining a model over all tasks. Data augmentation approaches such as rehearsal and pseudo-rehearsal have also been employed [22].

Other ideas from optimal control theory in combination with dynamic programming have also been applied to counteract catastrophic forgetting, with a cost functional similar in form to the action integral from physics and Lagrangian mechanics [16].

Orthogonal Gradient Descent (OGD) is a training augmentation or optimisation technique that modifies the gradient updates of subsequent tasks to be orthogonal to previous tasks [6, 1]. One can describe data in terms of a distribution defined over the input space, target values, and time (the order of data or tasks that are presented during training). OGD attempts to make gradient updates orthogonal to each other over time. ATLAS, in contrast, possesses distal orthogonality, meaning that if two inputs are far enough from each other in the input space, then corresponding gradient updates will be orthogonal. A corollary of this is that if the data distribution between tasks shifts in the input space, then the subsequent gradient updates will tend to be orthogonal. ATLAS does not use external memory like OGD. Extensions of OGD include PCA-OGD, which compresses gradient updates into principal components to reduce memory requirements [4]. The Neural Tangent Kernel (NTK) overlap matrices, as discussed by Doan et al. [4], could be a useful tool for analysing ATLAS architectures.

The survey by Delange et al. [3] gives an extensive overview of continual learning to address catastrophic forgetting. ATLAS is a model that implements parameter isolation, because of its use of piece-wise defined splines. Particularly relevant to ATLAS is the work on scale of initialisation and extreme memorisation [21]. Increasing the density of basis functions in ATLAS can lead to better memorisation, and increases the scale of some parameters in ATLAS.

It is not necessary to expand the capacity of an ATLAS model to learn new tasks, as with some other approaches [23]. ATLAS does in practice demonstrate something akin to "graceful forgetting" as discussed in Golkar et al. [8].

Pi-sigma neural networks use nodes that compute products instead of sums [25]. Pi-sigma neural networks reportedly have potential utility in building more efficient and expressive models than the standard ANNs. ATLAS can similarly compute products, and is more easily trained than the Pi-sigma neural networks.

B-splines, which form the basis of ATLAS, have been applied for machine learning [5]. Scardapane et al. [24] investigated trainable activation functions parameterised by splines. Uniform cubic B-splines have basis functions that are translates of one another [2]. Uniform cubic B-splines have been tested for memory retention, and ATLAS can be considered as an improvement on existing spline models [26]. B-splines, and by extension ATLAS, can be trained to fit lower frequency components,

expanded and trained again until a network is found with sufficient accuracy and generalisation, similar to other techniques [17, 18]

3 Notation

Vector quantities like $\vec{\mathbf{x}}$ are clearly indicated with a bar or arrow for legibility. Parameters, inputs, functions etc. without a bar or arrow are scalar quantities like $S(x)$. Some scalar quantities with indices are the scalar components of a vector like x_j or scalar parameters in the model like θ_i

4 Exponential Representation Theorem

Any continuous multi-variable function on a compact space can be uniformly approximated with multi-variable polynomials by the Stone-Weierstrass Theorem. Let $\mathcal{I}$ denote an index set of tuples of natural numbers including zero such that $i_j \in \mathbb{N}^0$ for all $j \in \mathbb{N}$ with $i = (i_1, .., i_n) \in \mathcal{I}$ and $a_i \in \mathbb{R}$. Multi-variable polynomials can be represented as:

$$y(\vec{\mathbf{x}}) = y(x_1, .., x_n) = \sum_{i \in \mathcal{I}} a_i x_1^{i_1} x_2^{i_2} ... x_n^{i_n} = \sum_{i \in \mathcal{I}} a_i \Pi_{j=1}^n x_j^{i_j}$$

Each monomial term $a_i \Pi_{j=1}^n x_j^{i_j}$ is a product of single-variable functions in each variable. It is desirable to rewrite products as sums. The product of single-variable functions can be rewritten as a sum using exponentials and logarithms.

Lemma 1. *For any $a_i \in \mathbb{R}$, there exists $\gamma_i > 0$ and $\beta_i > 0$, such that: $a_i = \gamma_i - \beta_i$*

Theorem 1 (Exponential representation theorem)**.** *Any multi-variable polynomial function $y(\vec{\mathbf{x}})$ of n variables over the positive orthant, can be exactly represented by continuous single-variable functions $g_{i,j}(x_j)$ and $h_{i,j}(x_j)$ in the form:*

$$y(\vec{\mathbf{x}}) = \sum_{i \in \mathcal{I}} \exp\big(\Sigma_{j=1}^n g_{i,j}(x_j)\big) - \exp\big(\Sigma_{j=1}^n h_{i,j}(x_j)\big)$$

Proof. Consider any monomial term $a_i \Pi_{j=1}^n x_j^{i_j}$ with $a_i \in \mathbb{R}$, then by Lemma 1 there exist strictly positive numbers $\gamma_i > 0$ and $\beta_i > 0$, such that:

$$\begin{aligned} a_i \Pi_{j=1}^n x_j^{i_j} &= \gamma_i \Pi_{j=1}^n x_j^{i_j} - \beta_i \Pi_{j=1}^n x_j^{i_j} \\ &= \exp\Big(\log\Big(\gamma_i \Pi_{j=1}^n x_j^{i_j}\Big)\Big) - \exp\Big(\log\Big(\beta_i \Pi_{j=1}^n x_j^{i_j}\Big)\Big) \\ &= \exp\Big(\log(\gamma_i) + \Sigma_{j=1}^n \log\Big(x_j^{i_j}\Big)\Big) - \exp\Big(\log(\beta_i) + \Sigma_{j=1}^n \log\Big(x_j^{i_j}\Big)\Big) \end{aligned}$$

The argument of each exponential function is a sum of single-variable functions and constants. Without loss of generality, a set of single-variable functions can be defined such that:

$$a_i \Pi_{j=1}^n x_j^{i_j} = \exp\big(\Sigma_{j=1}^n g_{i,j}(x_j)\big) - \exp\big(\Sigma_{j=1}^n h_{i,j}(x_j)\big)$$

Since this holds for any $a_i \Pi_{j=1}^n x_j^{i_j}$ and all $i \in \mathcal{I}$, it follows that:

$$y(\vec{\mathbf{x}}) = \sum_{i \in \mathcal{I}} \exp\big(\Sigma_{j=1}^n g_{i,j}(x_j)\big) - \exp\big(\Sigma_{j=1}^n h_{i,j}(x_j)\big)$$

□

This result is fundamental to the paper. Since every continuous function can be approximated with multi-variable polynomials, it follows that every continuous function can be approximated with positive and negative exponential functions. Some consideration must be given to single-variable function approximators, since it is the main building block for the representation theorem.

5 Single-Variable Function Approximation

Splines are piece-wise defined single-variable functions over some interval. Each sub-interval of a spline is most often locally given by a low degree polynomial, even though the global structure is not a low degree polynomial. B-splines are polynomial splines that are defined in a way that resembles other basis function formulations [2]. Each single-variable function in ATLAS is approximated with uniform cubic B-spline basis functions, shown in Figure 2. B-splines can approximate any single-variable function, similar to using the Fourier basis. With uniform B-splines, each basis function is scaled so that the unit interval is **uniformly** partitioned, as in Figure 2.

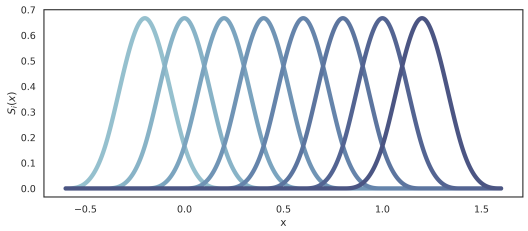
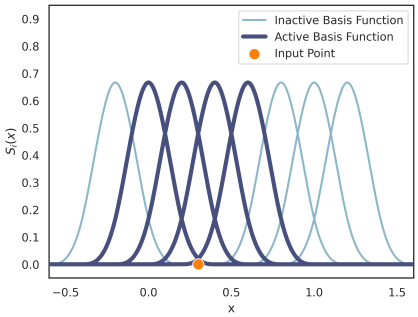


Figure 2: If uniformly spaced B-splines are used, then each basis function has the same shape. This makes it possible to use the same activation function by scaling and translating the inputs.

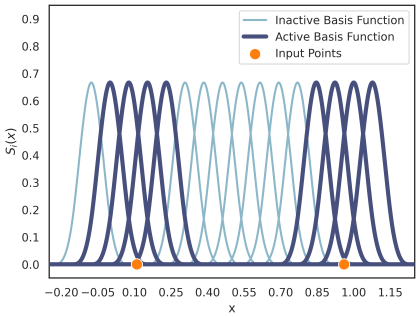
The activation function to implement B-splines is given by:

$$S(x) = \begin{cases} \frac{1}{6}x^3 & 0 \leq x < 1 \\ \frac{1}{6}\left[-3(x-1)^3 + 3(x-1)^2 + 3(x-1) + 1\right] & 1 \leq x < 2 \\ \frac{1}{6}\left[3(x-2)^3 - 6(x-2)^2 + 4\right] & 2 \leq x < 3 \\ \frac{1}{6}(4-x)^3 & 3 \leq x < 4 \\ 0 & otherwise \end{cases}$$

The choice was made to use uniform cubic B-splines due to their excellent performance and robustness to catastrophic forgetting, illustrated in Figure 3. Using uniform B-splines instead of arbitrary sub-interval partitions (also called knots in literature) makes optimisation easier. Optimising partitions is non-linear, but optimising only coefficient (also called control points) is linear and thus convex. By the contra-positive: If the basis functions do not have the same shape, then non-uniform B-splines are used.



(a) The activation is sparse and bounded.



(b) Distal orthogonality.

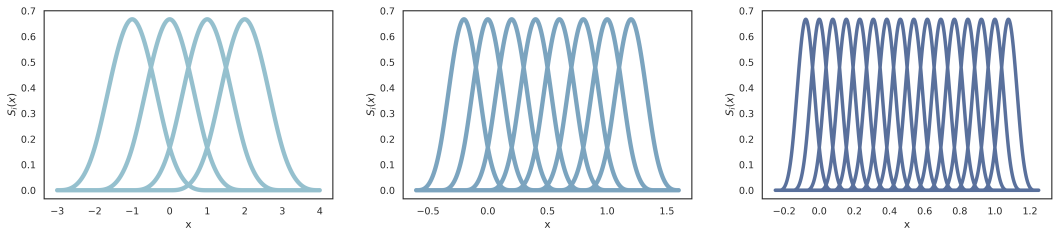
Figure 3: Single-variable function

Each basis function is multiplied by a parameter and summed together. The total number of basis functions is typically fixed. Cubic B-splines are 3rd order polynomials, and thus require a minimum of $3 + 1 = 4$ control points or basis functions. Instead of considering arbitrary densities of uniform cubic B-splines, we look at powers of two times the minimum number of basis functions, called ρ-density B-spline functions.

Definition 1 (ρ-density B-spline function)**.** A ρ-density B-spline function is a uniform cubic B-spline function with $2^{\rho+2}$ basis functions:

$$f(x) = \sum_{i=1}^{2^{\rho+2}} \theta_i S_i(x) = \sum_{i=1}^{2^{\rho+2}} \theta_i S(w_i x + b_i) = \sum_{i=1}^{2^{\rho+2}} \theta_i S((2^{\rho+2} - 3)x + 4 - i)$$

Consider the problem of expanding a single-variable function approximator with more basis functions to increase its expressiveness. Using the Fourier basis makes it trivially easy by adding higher frequency sines and cosines with coefficients initialised to zero. It is trickier to achieve something similar with cubic B-splines. There are algorithms for creating new splines from existing splines with knot insertion, but the intermediate steps result in non-uniform knots and splines. A simple and practical compromise that we propose is to use mixtures of different ρ-density B-spline functions, as illustrated in Figure 4.



(a) $\rho = 0$ (b) $\rho = 1$ (c) $\rho = 2$

Figure 4: Doubling densities of basis functions.

Definition 2 (mixed-density B-spline function)**.** A mixed-density B-spline function is a single-variable function approximator that is obtained by summing together different ρ-density B-spline functions. Only the maximum ρ-density B-spline function has trainable parameters, the others are constant. Mixed-density B-spline functions are of the form:

$$f(x) = \sum_{\rho=0}^{r} \sum_{i=1}^{2^{\rho+2}} \theta_{\rho,i} S_{\rho,i}(x)$$

Only the **maximum** $r = \rho$-density B-spline has trainable coefficients. All lower density $r > \rho$-density B-spline have frozen and constant coefficients. The maximum $r = \rho$-density B-spline has trainable coefficients with gradient updates that are orthogonal if the distance between two inputs is large enough.

Similar to increasing the expressiveness of a Fourier basis function approximator by adding higher frequency terms, one can add larger density cubic B-spline functions. Analytically, we can initialise all the new parameters $\theta_{r+1,i} = 0,\ \forall i \in \mathbf{N}$ such that:

$$f(x) = \sum_{\rho=0}^{r} \sum_{i=1}^{2^{\rho+2}} \theta_{\rho,i} S_{\rho,i}(x) = \sum_{\rho=0}^{r+1} \sum_{i=1}^{2^{\rho+2}} \theta_{\rho,i} S_{\rho,i}(x)$$

It is therefore possible to create a minimal model with $r = 0$ initialised at zero, and train the model until convergence. Then one can create a new model with $r = 1$, by subsuming the previous model's parameters, and train this more expressive model until convergence. This process of training and expansion can be continued indefinitely, and is shown in Figure 5.

6 ATLAS

ATLAS is named for carrying the burden of all it must remember, after the Titan god Atlas in Greek mythology who was tasked with holding the weight of the world. ATLAS is also an acronym for AddiTive exponentiaL Additive Splines.

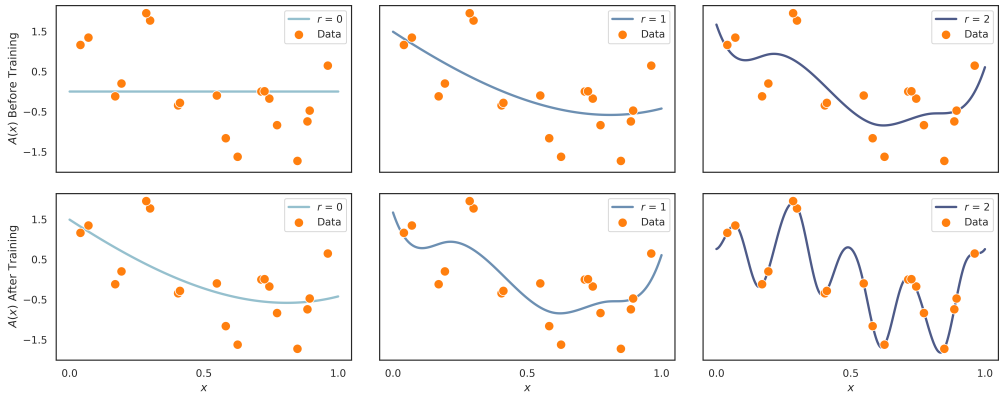


Figure 5: Doubling densities of basis functions before and after training.

Definition 3 (ATLAS)**.** ATLAS is a function approximator of n variables, with mixed-density B-spline functions $f_j(x_j)$, $g_{i,j}(x_j)$, and $h_{i,j}(x_j)$ in the form:

$$A(\vec{\mathbf{x}}) := \sum_{j=1}^{n} f_j(x_j) + \sum_{k=1}^{M} \frac{1}{k^2} \exp\left(\Sigma_{j=1}^{n} g_{k,j}(x_j)\right) - \frac{1}{k^2} \exp\left(\Sigma_{j=1}^{n} h_{k,j}(x_j)\right)$$

ATLAS is equivalently given by the compact notation:

$$A(\vec{\mathbf{x}}) := F(\vec{\mathbf{x}}) + \sum_{k=1}^{M} \frac{1}{k^2} \exp(G_k(\vec{\mathbf{x}})) - \frac{1}{k^2} \exp(H_k(\vec{\mathbf{x}}))$$

The absolutely convergent series of scale factors k^{-2} was chosen for numerical stability and to ensure the model is absolutely convergent. Another feature is that the series of scale factors also breaks the symmetry that would otherwise exist if all mixed-density B-spline functions were initialised to zero. Initialising all the parameters to be zero is a departure from the conventional approach of random initialisation. The number of exponential terms can be increased without changing the output of the model. We can choose to initialise $G_{M+1}(\vec{\mathbf{x}}) = 0$ and $H_{M+1}(\vec{\mathbf{x}}) = 0$, such that the model capacity can be increased without changing the output of the model.

ATLAS is a universal function approximator with robust memory retention. It possesses three properties atypical of most universal function approximators:

1. The activity within ATLAS is sparse – most neural units are zero and inactive.
2. The gradient vector with respect to trainable parameters is bounded regardless of the size and capacity of the model, so training is numerically stable for many possible training hyper-parameters.
3. Inputs that are sufficiently far from each other have orthogonal representations.

Property 1 (Sparsity)**.** *For any $\vec{\mathbf{x}} \in D(A) \subset R^n$ and bounded trainable parameters θ_i with index set Θ, the gradient vector of trainable parameters (for ATLAS) is sparse:*

$$\left\| \vec{\nabla}_{\vec{\theta}} A(\vec{\mathbf{x}}) \right\|_0 = \sum_{i \in \Theta} d_{Hamming} \left(\frac{\partial A}{\partial \theta_i}(\vec{\mathbf{x}}), 0 \right) \leq 4n(2M+1)$$

Remark. For a fixed number of variables n, the model has a total of $n2^{r+2}(2M+1)$ trainable parameters. The gradient vector has a maximum of $4n(2M+1)$ non-zero entries, which is independent of r. Recall that only the maximum density ($\rho = r$) cubic B-spline function has trainable parameters. The fraction of trainable basis functions that are active is at most 2^{-r}. Sparsity entails efficient implementation, and suggests possible memory retention and robustness to catastrophic forgetting.

Property 2 (Gradient flow attenuation)**.** *For any $\vec{\mathbf{x}} \in D(A) \subset R^n$ and bounded trainable parameters θ_i with index set Θ: if all the mixed-density B-spline functions are bounded, then the gradient vector of trainable parameters for ATLAS is bounded:*

$$\left\| \vec{\boldsymbol{\nabla}}_{\vec{\theta}} A(\vec{\mathbf{x}}) \right\|_1 = \sum_{\theta_i \in \Theta} \left| \frac{\partial A}{\partial \theta_i}(\vec{\mathbf{x}}) \right| < U$$

Remark. For a fixed number of variables n, the model has a total of $n2^{r+2}(2M+1)$ trainable parameters. The factor of k^{-2} inside the expression for ATLAS is necessary to ensure the sum is convergent in the limit of infinitely many exponential terms $M \to \infty$. Only the maximum density ($\rho = r$) cubic B-spline function has trainable parameters, so that the gradient vector is bounded in the limit of arbitrarily large densities $r \to \infty$. It is worth recalling that at most four basis functions are active for uniform cubic B-spline functions, regardless of the density, but the smaller densities cannot be trainable, otherwise this property does not hold. The gradient vector has bounded L1 norm for any number of basis functions and exponential terms. The bounded gradient vector implies that ATLAS is numerically stable during training, regardless of its size or parameter count.

Property 3 (Distal orthogonality)**.** *For any $\vec{\mathbf{x}}, \vec{\mathbf{y}} \in D(A) \subset R^n$ and bounded trainable parameters θ_i, there exists a $\delta > 0$ such that:*

$$\min_{j=1,\ldots,n} \{|x_j - y_j|\} > \delta \implies \langle \vec{\boldsymbol{\nabla}}_{\vec{\theta}} A(\vec{\mathbf{x}}), \vec{\boldsymbol{\nabla}}_{\vec{\theta}} A(\vec{\mathbf{y}}) \rangle = 0$$

Remark. Two points that sufficiently differ in each input variable have orthogonal parameter gradients. It is worth mentioning that the condition resembles a cross-like region in two variables, and planes that intersect in higher dimensions. Distal orthogonality means ATLAS is reasonably robust to catastrophic forgetting.

ATLAS can be implemented with 1D convolution, reshaping, embedding, multiplication and dense layers. The same kind of basis functions have to be computed for each input variable, hence the choice of 1D convolutions. By correctly scaling, shifting, and rounding inputs one can compute only the non-zero basis functions with the aid of convolution and embedding layers to lookup the parameters. ATLAS has time complexity $\mathcal{O}(\rho M n)$, and space complexity $\mathcal{O}(2^\rho M n)$, thanks to efficient usage of embedding layers. It is worth mentioning that the engineering process settled on using 1D convolution and embedding layers as an efficient method for approximating high-dimensional functions, which potentially suggests a deeper connection to other high-dimensional convolutional models (with small filters) and embedding layers.

7 Methodology

The main point of the experimentation was to demonstrate that ATLAS exhibits distal orthogonality, and slightly improved memory retention without any regularisation or augmentation techniques. Demonstrating this for every possible target function is not feasible, thus a random set of problems were considered instead.

Two variants of ATLAS are considered for comparison: one with guaranteed distal orthogonality, and another without distal orthogonality that lets lower density (or lower frequency) basis function coefficients change. This modification leads to a variant that does not possess any of the three properties of ATLAS as outlined in the theory sections. The ATLAS and variant models have $M = 10$ exponential terms that are strictly positive, and $M = 10$ exponential terms that are strictly positive. The maximum density $r = 4 \implies 4 \times 2^r = 64$ is the number of basis functions for each input variable, with the greatest expressive power. The lower density basis functions do not add expressive power, and can alter training dynamics, potentially including greater off-target effects.

To test memory retention, two tasks, presented to ATLAS one after the other, were constructed. The details of each task are given below.

Task 1 The target function is a randomly generated function with 1000 radial basis functions with positions chosen uniformly over the input domain, using the Euclidean metric. The scale parameters and weights are also randomly sampled, and all radial basis functions are added together to obtain a

target function. 10000 data points are sampled for training, validation and testing. All sampled data points have added Gaussian noise with a variance chosen randomly for each trial.

Task 2 The target function is the same as Task 1, except for a randomly positioned region of a fixed width, where the function values for Task 1 are replaced with another randomly sampled function of 1000 radial basis functions that is sampled independently of the target function for Task 1.

8 Results

A randomly chosen trial is presented for visual inspection. As shown in Figure 6 and Figure 7 the effect of distal orthogonality is clear and crisp boundaries, with the absolute change in the model output before and after training on Task 2. The absolute difference between the target functions and the 2nd output of each model is shown in Figure 8 and Figure 9.

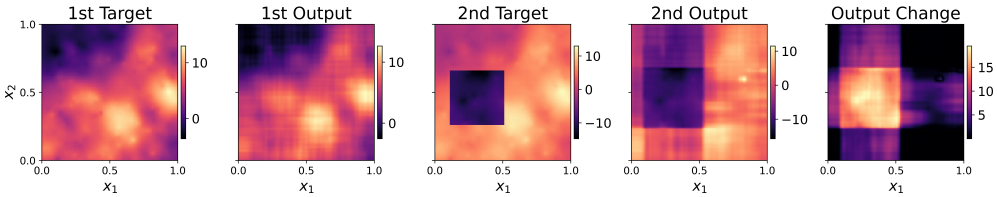


Figure 6: Guaranteed distal orthogonality.

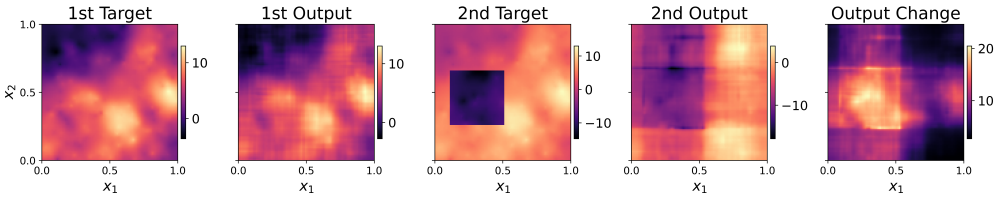


Figure 7: No guaranteed distal orthogonality.

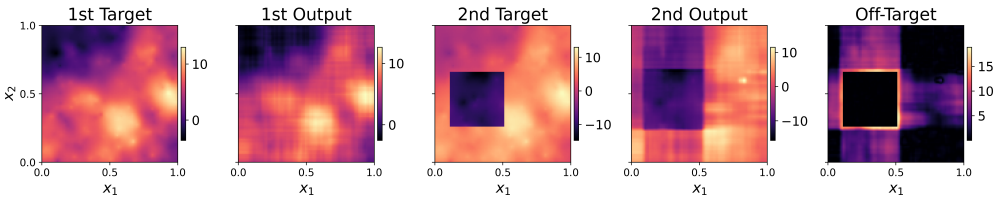


Figure 8: Guaranteed distal orthogonality.

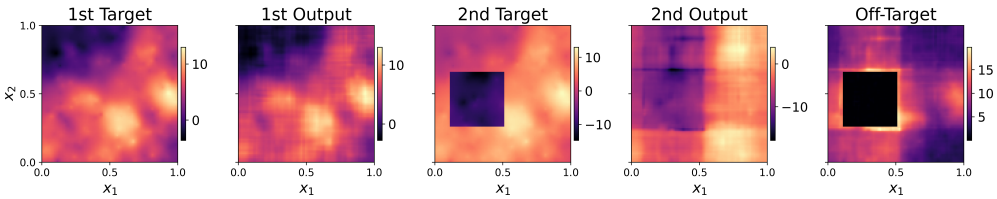


Figure 9: No guaranteed distal orthogonality.

The effect of distal orthogonality on the averaged MAE for various trials is presented in Figure 10 and Figure 11. The removal of distal orthogonality lead to increased errors and off-target effects on average.

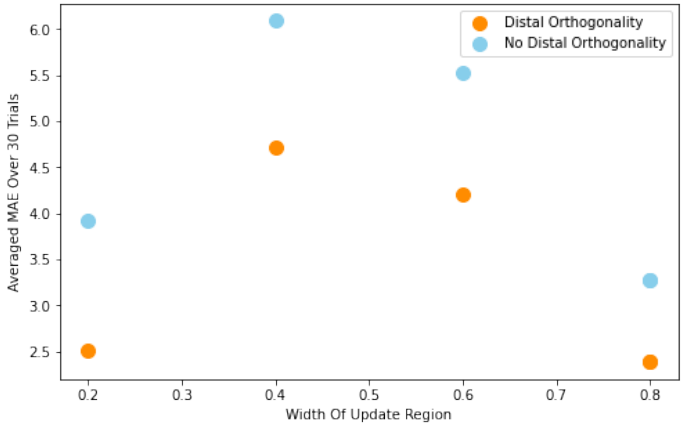


Figure 10: Scatter plot showing the effect of distal orthogonality on the final test error on task 2 for the two dimensional input.

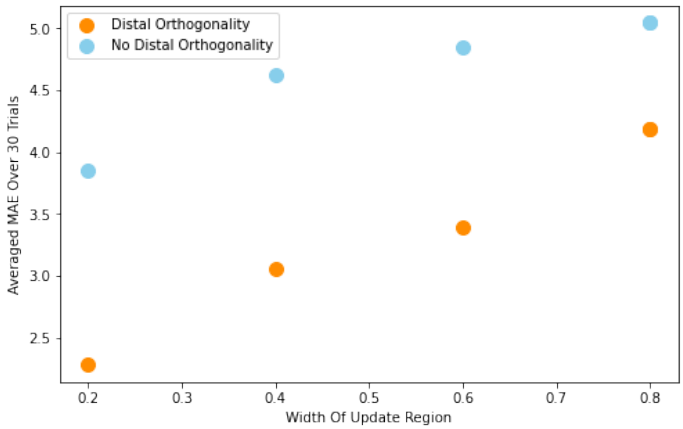


Figure 11: Scatter plot showing the effect of distal orthogonality on the final test error on task 2 for the 8-dimensional input.

9 Conclusion

The main contribution of the paper is theoretical and technical. A representation theorem is presented that outlines how to approximate multi-variable functions with single-variable functions, and exponential functions. ATLAS approximates all arbitrary single-variable functions with mixed-density B-spline functions. ATLAS is constructed in such a way that the gradient vector with respect to trainable parameters is bounded, regardless of how large an ATLAS model is. ATLAS was shown to exhibit intrinsic memory retention, except for a small cross-shaped region of interference and off-target effects.

The chosen experiments demonstrated the theoretically derived predictions and contrasted two models, incuding a variant of ATLAS without distal orthogonality guarantees.

As far as societal impacts are concerned: It is possible that ATLAS could allow for the creation of more powerful machine learning algorithms, that require less resources to train and deploy. Further testing is needed to make any concrete claim.

References

[1] M. A. Bennani, T. Doan, and M. Sugiyama. Generalisation guarantees for continual learning with orthogonal gradient descent, 2021. URL `https://openreview.net/forum?id=hecuSLbL_vC`.

[2] K. Branson. A practical review of uniform b-splines, 2004.

[3] M. Delange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh, and T. Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2021. doi: 10.1109/tpami.2021.3057446. URL https://doi.org/10.1109%2Ftpami.2021.3057446.

[4] T. Doan, M. Abbana Bennani, B. Mazoure, G. Rabusseau, and P. Alquier. A theoretical analysis of catastrophic forgetting through the ntk overlap matrix. In A. Banerjee and K. Fukumizu, editors, *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 1072–1080. PMLR, 13–15 Apr 2021. URL https://proceedings.mlr.press/v130/doan21a.html.

[5] A. S. Douzette. B-splines in machine learning. Master's thesis, Department of Mathematics, University of Oslo, 2017.

[6] M. Farajtabar, N. Azizan, A. Mott, and A. Li. Orthogonal gradient descent for continual learning. In S. Chiappa and R. Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 3762–3773. PMLR, 26–28 Aug 2020. URL https://proceedings.mlr.press/v108/farajtabar20a.html.

[7] R. M. French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

[8] S. Golkar, M. Kagan, and K. Cho. Continual learning via neural pruning, 2019. URL https://arxiv.org/abs/1903.04476.

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

[10] B. Hanin. Universal function approximation by deep neural nets with bounded width and relu activations. *Mathematics*, 7(10):992, 2019.

[11] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[12] P. Kaushik, A. Gain, A. Kortylewski, and A. Yuille. Understanding catastrophic forgetting and remembering in continual learning with optimal relevance mapping. *CoRR*, abs/2102.11343, 2021. URL https://arxiv.org/abs/2102.11343.

[13] R. Kemker, M. McClure, A. Abitino, T. Hayes, and C. Kanan. Measuring catastrophic forgetting in neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[14] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017. ISSN 0027-8424. doi: 10.1073/pnas.1611835114. URL https://www.pnas.org/content/114/13/3521.

[15] A. Kratsios. The universal approximation property: Characterization, construction, representation, and existence. *Annals of Mathematics and Artificial Intelligence*, 89(5):435–469, 2021. doi: 10.1007/s10472-020-09723-1.

[16] R. Krishnan and P. Balaprakash. Meta continual learning via dynamic programming, 2020. URL https://arxiv.org/abs/2008.02219.

[17] S. H. Lane, M. Flax, D. Handelman, and J. Gelfand. Multi-layer perceptrons with b-spline receptive field functions. In *Advances in Neural Information Processing Systems*, pages 684–692, 1991.

[18] A. C. Li and D. Pathak. Functional regularization for reinforcement learning via learned fourier features, 2021. URL https://arxiv.org/abs/2112.03257.

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

[20] K. McRae and P. A. Hetherington. Catastrophic interference is eliminated in pretrained networks. In *Proceedings of the 15h Annual Conference of the Cognitive Science Society*, pages 723–728, 1993.

[21] H. Mehta, A. Cutkosky, and B. Neyshabur. Extreme memorization via scale of initialization, 2020. URL `https://arxiv.org/abs/2008.13363`.

[22] A. Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7(2): 123–146, 1995.

[23] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell. Progressive neural networks, 2016. URL `https://arxiv.org/abs/1606.04671`.

[24] S. Scardapane, M. Scarpiniti, D. Comminiello, and A. Uncini. Learning activation functions from data using cubic spline interpolation. In *Italian Workshop on Neural Nets*, pages 73–83. Springer, 2017.

[25] Y. Shin and J. Ghosh. The pi-sigma network: an efficient higher-order neural network for pattern classification and function approximation. In *IJCNN-91-Seattle International Joint Conference on Neural Networks*, volume i, pages 13–18 vol.1, 1991. doi: 10.1109/IJCNN.1991.155142.

[26] H. van Deventer, P. J. van Rensburg, and A. Bosman. Kasam: Spline additive models for function approximation, 2022. URL `https://arxiv.org/abs/2205.06376`.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? **[TODO]**
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? **[TODO]**
   (b) Did you mention the license of the assets? **[TODO]**
   (c) Did you include any new assets either in the supplemental material or as a URL? **[TODO]**
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]