Supplementary Material

1 Definitions of the system, training

1.1 Notations

All vectors are denoted using the bra-ket notation. For vectors $\boldsymbol{v}, \boldsymbol{u} \in \mathbb{R}^n$, we write from this point forward,

$$\boldsymbol{v} \rightarrow |v\rangle\,, \tag{S.1}$$

$$\boldsymbol{u}^T \rightarrow \langle u|\,. \tag{S.2}$$

Dot products in bra-ket notation are written as $\langle v|u\rangle$, for two vectors in the same vector space. In the context of this work, this means the network's visible and hidden detectors become memories ($|M_\mu\rangle$) and labels ($|L_\mu\rangle$). The memories ($|M_\mu\rangle$) live in the input space, and the labels (($|L_\mu\rangle$) live in the output space. To manipulate them, we define the following memory and label orthonormal bases,

$$\{|\hat{m}_i\rangle\}, \quad i \in [0, 784) \tag{S.3}$$

$$\left\{ \left|\hat{l}_d\right\rangle \right\}, \quad d \in [0, 9]. \tag{S.4}$$

Intuitively, the memory basis fetches individual pixels of a 28x28 (MNIST) image. The label basis, can be understood as isolating a probability of being (or not being) of category/class d.

1.2 Generalized Hopfield Network classifier

In the context of this work, the input is a flattened 28×28 image of a handwritten (MNIST) digit. The pixels of the image range between -1 and 1. Generally, when illustrated a blue-white-red ('bwr') colormap is used, where blue is -1, white is 0 and red is $+1$.

The goal of the network is to classify the input image into its digit class (0, 1, 2, ..., 9). The output is a vector, where each element can be understood as the probability of being (or **not** being) of digit class d. The elements of the output range continuously from -1 (is **not** digit d) to $+1$ (is digit class d). It is computed as follows,

$$|o(\sigma)\rangle = \tanh\left(\sum_\mu |L_\mu\rangle\, f_n\Big(\frac{\langle M_\mu|\sigma\rangle}{T}\Big) \right), \tag{S.5}$$

Notice each memory $|M_\mu\rangle$ is associated to a label vector $|L_\mu\rangle$, and tanh is applied element-wise.

$$f_n(x) = \Big(ReLU(x)\Big)^n = \begin{cases} x^n, & x \geq 0 \\ 0, & x < 0 \end{cases} . \tag{S.6}$$

Due to the nature of the inputs (ranging from -1 to +1), $\langle M_\mu|\sigma\rangle$ is almost[1] always positive. Hence,

$$\boxed{|o(\sigma)\rangle = \tanh\left(\sum_\mu |L_\mu\rangle \left(\frac{\langle M_\mu|\sigma\rangle}{T}\right)^n\right).} \tag{S.7}$$

The hyperparameters of the system are then n and T, and control the non-linearity of the activation function.

1.3 Training

The memories and labels are developed through training. To that end, we define a training set $\mathcal{T}$ composed of training (input) samples $|\sigma\rangle$ with a known expected output $\left|t_{|\sigma\rangle}\right\rangle$. To give an example of the structure of $\left|t_{|\sigma\rangle}\right\rangle$, if $|\sigma\rangle$ is a 5, then the expected output will be,

$$\left\langle \hat{l}_5 \middle| t_{|\sigma\rangle} \right\rangle = +1, \tag{S.8}$$

$$\left\langle \hat{l}_{d\neq 5} \middle| t_{|\sigma\rangle} \right\rangle = -1. \tag{S.9}$$

In another form, $\left|t_{|\sigma\rangle}\right\rangle = (-1,-1,-1,-1,-1,+1,-1,-1,-1,-1)$. Then, each training step is centered around gradient descent, with some normalizations.

1.3.1 Cost function

Gradient descent implies an energy/cost function that the system tries to optimize. In this classification task, the system will try to minimize the difference between the expected output and the network's computed output. That is,

$$C = \sum_{|\sigma\rangle\in\mathcal{T}} \sum_d \left(\left\langle \hat{l}_d \middle| t_{|\sigma\rangle} \right\rangle - \left\langle \hat{l}_d \middle| o(\sigma) \right\rangle \right)^{2n} \tag{S.10}$$

is minimized. [2]

[1] The negative cases are somewhat contrived, and trivial to recognize; red background.

[2] The exponent is generally written as $2m$, but in this text $m = n$ is always used.

1.3.2 Training dynamics

The dynamics are:

$$|M_\mu(t+\delta t)\rangle = N_\mu(t)\Bigg(|M_\mu(t)\rangle - \delta t\Big(\max_j \Big|\frac{\partial C}{\partial \langle \hat{m}_j|M_\mu(t)\rangle}\Big|\Big)^{-1} \frac{\partial C}{\partial \langle M_\mu(t)|}\Bigg), \tag{S.11}$$

$$|L_\mu(t+\delta t)\rangle = \mathcal{C}\Bigg(|L_\mu(t)\rangle - \delta t\Bigg(\max_d \Bigg|\frac{\partial C}{\partial \Big\langle \hat{l}_d\Big|L_\mu(t)\Big\rangle}\Bigg|\Bigg)^{-1} \frac{\partial C}{\partial \langle L_\mu(t)|}\Bigg). \tag{S.12}$$

For a given memory, the gradient is calculated - a vector - and is normalized so that its largest element has amplitude 1. The normalized gradient is subtracted from the memory with a factor δt.
The resulting memory is normalized by $N_\mu(t)$. If all pixels of a memory are less than 1, $N_\mu(t) = 1$. Otherwise, $N_\mu(t)$ is equal to the largest absolute pixel. A similar process is followed for the labels, but the last normalization step is replaced by clipping, $\mathcal{C}(\cdot)$.

$$N_\mu(t) = \max\Big(1, \max_i \Big|\langle \hat{m}_i|M_\mu(t)\rangle - \delta t\Big(\max_j \Big|\frac{\partial C}{\partial \langle \hat{m}_j|M_\mu(t)\rangle}\Big|\Big)^{-1} \frac{\partial C}{\partial \langle \hat{m}_i|M_\mu(t)\rangle}\Big|\Big), \tag{S.13}$$

$$\mathcal{C}(x) = \begin{cases} +1, & x > +1 \\ -1, & x < -1 \\ x, & else \end{cases} . \tag{S.14}$$

The gradient descent quantities, can be expanded into

$$\begin{aligned} \frac{\partial C}{\partial \langle M_\mu(t)|} = & -\sum_{|\sigma\rangle\in\mathcal{T}} \sum_d \Bigg(\Big\langle \hat{l}_d\Big|t_{|\sigma\rangle}\Big\rangle - \Big\langle \hat{l}_d\Big|o(\sigma)\Big\rangle \Bigg)^{2n-1} \\ & \times \Bigg(1 - \Big\langle \hat{l}_d\Big|o(\sigma)\Big\rangle^2\Bigg) \Big\langle \hat{l}_d\Big|L_\mu\Big\rangle \Big(\frac{\langle M_\mu|\sigma\rangle}{T}\Big)^{n-1} |\sigma\rangle \end{aligned} \tag{S.15}$$

$$\begin{aligned} \frac{\partial C}{\partial \langle L_\mu(t)|} = & -\sum_{|\sigma\rangle\in\mathcal{T}} \sum_d \Bigg(\Big\langle \hat{l}_d\Big|t_{|\sigma\rangle}\Big\rangle - \Big\langle \hat{l}_d\Big|o(\sigma)\Big\rangle \Bigg)^{2n-1} \\ & \times \Bigg(1 - \Big\langle \hat{l}_d\Big|o(\sigma)\Big\rangle^2\Bigg) \Big(\frac{\langle M_\mu|\sigma\rangle}{T}\Big)^{n} \Big|\hat{l}_d\Big\rangle \end{aligned} \tag{S.16}$$

Due to normalization, constant overall factors do not matter in the gradient descent quantities.

2 Memory Coefficients

Gradient descent quantities are effectively a weighed sum of the training samples. The normalizations are simply additional factors, hence any memory (with small initial conditions)

can be decomposed into

$$|M_\mu\rangle = \sum \alpha_{\mu,|\sigma\rangle} |\sigma\rangle \,. \tag{S.17}$$

Likewise, the training dynamics can be written in terms of the α-coefficients.

2.1 The Moore-Penrose inverse

One way to obtain the α's, is to use the Moore-Penrose inverse. Consider the following matrix,

$$\boldsymbol{T} = \begin{pmatrix} \langle\sigma_1| \\ \langle\sigma_2| \\ \langle\sigma_3| \\ \vdots \\ \langle\sigma_{N_T}| \end{pmatrix}, \tag{S.18}$$

containing all N_T training samples. Then, there **always** exists a Singular Value Decomposition (SVD) such that,

$$\boldsymbol{T} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T, \tag{S.19}$$

where $\boldsymbol{S}$ is a diagonal matrix (generally not square) with entries (all positive) sorted in descending order, and $\boldsymbol{U}, \boldsymbol{V}$ are unitary; $\boldsymbol{U}^T\boldsymbol{U} = I$. We define then, for this diagonal matrix $\boldsymbol{S}$, a pseudo inverse

$$(\boldsymbol{S}^\dagger)_{ij} = \begin{cases} \frac{1}{\boldsymbol{S}_{ij}}, & \boldsymbol{S}_{ij} > \epsilon \\ 0, & else \end{cases} \quad . \tag{S.20}$$

That is we take the reciprocal of all diagonal entries above a tolerance ϵ. Then, we can define the pseudo-matrix for all matrices, and in particular our training set matrix:

$$\boldsymbol{T}^\dagger = \boldsymbol{V}\boldsymbol{S}^\dagger\boldsymbol{U}^{\boldsymbol{T}}. \tag{S.21}$$

An important property of the pseudo-inverse is[3],

$$\boldsymbol{T}\boldsymbol{T}^\dagger\boldsymbol{T} = \boldsymbol{T}. \tag{S.22}$$

Consider a memory $|M\rangle$ with an unknown composition, and we obtain some a set of α's by using the pseudo-inverse,

$$\boldsymbol{\alpha}^T_{inv} = \langle M|\, \boldsymbol{T}^\dagger = \boldsymbol{\alpha}^T_{unknown}\boldsymbol{T}\boldsymbol{T}^\dagger, \tag{S.23}$$

which may or may not be the same, we are guaranteed by the property highlighted above that the reconstruction is the same,

$$\boldsymbol{\alpha}^T_{inv}\boldsymbol{T} = \boldsymbol{\alpha}^T_{unknown}\boldsymbol{T}. \tag{S.24}$$

[3] $\boldsymbol{T}\boldsymbol{T}^\dagger\boldsymbol{T} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T\boldsymbol{V}\boldsymbol{S}^\dagger\boldsymbol{U}^{\boldsymbol{T}}\boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T = \boldsymbol{U}\boldsymbol{S}\boldsymbol{S}^\dagger\boldsymbol{S}\boldsymbol{V}^T = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T = \boldsymbol{T}$. Which is exact if $\epsilon = 0$, but numerically we often choose $\epsilon \sim 10^{-15}$, and singular values less than that are approximated to be zero/negligible.

Uniqueness of our α representation, depends on the rank of $\boldsymbol{T}$, but existence and equivalence are always guaranteed. For examples see Fig. S.1.
Practically, we use the Numpy library to compute the Moore-Penrose inverse, with a tolerance value equal to 10^{-15} times the largest singular value. Generally, the largest deviation we observe between a reconstruction pixel and the original pixel is in the order of 10^{-6}.

2.1.1 Coarse-graining α

The α-representation is useful, but limited when the training set contains multiple examples per digit class. In that case, to properly visualize the dynamics we define,

$$\bar{\alpha}_d = \sum_{|\sigma\rangle \in \mathcal{T}_d} \alpha_{|\sigma\rangle} \tag{S.25}$$

where $\mathcal{T}_d$ is a subset of $\mathcal{T}$ which contains only digit of class d. The intuition here is as follows, consider a memory $|M\rangle$, one can show it can be decomposed into

$$|M\rangle = \sum_d \bar{\alpha}_d \, |\bar{\sigma}_d\rangle + \sum_d \sum_{|\sigma\rangle \in \mathcal{T}_d} \alpha_{|\sigma\rangle} \, |\delta\sigma\rangle \, . \tag{S.26}$$

Marginalization then recovers the proportions of class-averaged digits $|\bar{\sigma}_d\rangle$. Ignoring digit specific variations $|\delta\sigma\rangle$. E.g. $\bar{\alpha}_1$ represents the proportion of the average 1 in a memory.

3 Study of the first saddle / The 1-memory system

Due to the small initial conditions, we can approximate the memories of a system to be identical initially. For a system of N_k memories, this means

$$|o(\sigma)\rangle = \tanh\left(\sum_{\mu}^{N_k} |L_\mu\rangle \left(\frac{\langle M_\mu | \sigma \rangle}{T} \right)^n \right) \approx \tanh\left(N_k \, |L\rangle \left(\frac{\langle M | \sigma \rangle}{T} \right)^n \right), \tag{S.27}$$

$$|o(\sigma)\rangle \approx \tanh\left(|L\rangle \left(\frac{\langle M | \sigma \rangle}{\tilde{T}} \right)^n \right), \tag{S.28}$$

Hence, any system initially behaves as a 1-memory system (see Fig. S.21 and Fig. S.22) with an effective temperature,

$$\tilde{T} = \frac{T}{\sqrt[n]{N_k}}. \tag{S.29}$$

It is then important to understand the behavior of the fixed points of the 1-memory system, as these become the saddles of higher-dimensional systems.

3.1 The analytical case : 2 digits

Consider the case of two training samples, $|A\rangle$, $|B\rangle$, of (distinct) class A and B respectively. We define the following shorthand for our basis, $\left|\hat{l}_A\right\rangle$ for class A, $\left|\hat{l}_B\right\rangle$ for class B and $\left|\hat{l}_\gamma\right\rangle$ any class other than A, B. This means,

$$\left\langle \hat{l}_A \middle| t_{|A\rangle} \right\rangle = 1 = \left\langle \hat{l}_B \middle| t_{|B\rangle} \right\rangle, \tag{S.30}$$

$$\left\langle \hat{l}_B \middle| t_{|A\rangle} \right\rangle = \left\langle \hat{l}_A \middle| t_{|B\rangle} \right\rangle = -1 = \left\langle \hat{l}_\gamma \middle| t_{|A\rangle} \right\rangle = \left\langle \hat{l}_\gamma \middle| t_{|B\rangle} \right\rangle. \tag{S.31}$$

Next for $|A\rangle \neq |B\rangle$, we can always construct[4] $|A'\rangle, |B'\rangle$ such that,

$$\langle A'|A\rangle = \langle B'|B\rangle = 1, \tag{S.32}$$

$$\langle A'|B\rangle = \langle B'|A\rangle = 0. \tag{S.33}$$

For the memory we write,

$$|M\rangle = \alpha_{|A\rangle}\,|A\rangle + \alpha_{|B\rangle}\,|B\rangle\,. \tag{S.34}$$

3.1.1 The ℓ invariant manifold (approximation I)

For the label, plugging $\left\langle \hat{l}_A \middle| L \right\rangle = -\left\langle \hat{l}_B \middle| L \right\rangle$ into (S.12, S.16) leads directly to $\partial_t \left\langle \hat{l}_A \middle| L \right\rangle = -\partial_t \left\langle \hat{l}_B \middle| L \right\rangle$. This means we have an invariant manifold. As for stability, one can define

$$\left\langle \hat{l}_A \middle| L \right\rangle = \ell - \delta\ell, \tag{S.35}$$

$$\left\langle \hat{l}_B \middle| L \right\rangle = -\Big(\ell + \delta\ell\Big), \tag{S.36}$$

and obtain a first order approximation of $\partial_t(\delta\ell)$, this is long so we omit it, but it leads to wide regions of stability. Also, by examining the dynamics one can see that $\left\langle \hat{l}_\gamma \middle| L \right\rangle \to -1$ monotonically.[5] The 1-memory system enters this regime (and does so rapidly for high-n, see Fig. S.21 and Fig. S.22), so we assume for the entire dynamics,

$$\left\langle \hat{l}_A \middle| L \right\rangle = \ell = -\left\langle \hat{l}_B \middle| L \right\rangle, \tag{S.37}$$

$$\left\langle \hat{l}_\gamma \middle| L \right\rangle = -1. \tag{S.38}$$

Notice, this simplifies the system drastically, it is now 3 dimensional; $\alpha_{|A\rangle}, \alpha_{|B\rangle}$ and ℓ. The output is then,

$$\left\langle \hat{l}_A \middle| o(\sigma) \right\rangle = \tanh\left(\ell\Big(\frac{\langle M|\sigma\rangle}{\tilde{T}}\Big)^n\right) = -\left\langle \hat{l}_B \middle| o(\sigma) \right\rangle, \tag{S.39}$$

$$\left\langle \hat{l}_\gamma \middle| o(\sigma) \right\rangle = -\tanh\left(\Big(\frac{\langle M|\sigma\rangle}{\tilde{T}}\Big)^n\right), \tag{S.40}$$

[4] These are the columns of our pseudo-inverse.

[5] Up to numerical artefacts for very small temperatures.

where $\langle M|\sigma\rangle$ can be written directly in terms of the α's,

$$\langle M|\sigma\rangle = \alpha_{|A\rangle}\langle A|\sigma\rangle + \alpha_{|B\rangle}\langle B|\sigma\rangle\,. \tag{S.41}$$

This leads to the following cost function,

$$C = \sum_{|\sigma\rangle\in\{|A\rangle,|B\rangle\}} \left(\left\langle \hat{l}_A \middle| t_{|\sigma\rangle}\right\rangle - \left\langle \hat{l}_A \middle| o(\sigma)\right\rangle\right)^{2n} + 4\sum_{|\sigma\rangle\in\{|A\rangle,|B\rangle\}} \left(1 + \left\langle \hat{l}_\gamma \middle| o(\sigma)\right\rangle\right)^{2n}, \tag{S.42}$$

ignoring overall factors.

3.1.2 Memory normalization (approximation II)

Due to the nature of the MNIST dataset, any (non-identical) $|A\rangle$ and $|B\rangle$ will have two overlapping pixels with value -1, as well as two overlapping pixels of opposite signs with amplitude near 1. This means, for any (real) k_A, k_B,

$$\max_j \left| k_A \langle \hat{m}_j|A\rangle + k_B \langle \hat{m}_j|B\rangle \right| = |k_A| + |k_B|. \tag{S.43}$$

Simply put, for any linear combination of two (non-identical) training samples, the largest absolute pixel is approximately equal to the absolute sum of the linear coefficients of $|A\rangle$ and $|B\rangle$.

3.1.3 Gradient descent quantities

We define now the gradient descent quantities,

$$\nabla_{|A\rangle} = -\langle A'|\frac{\partial C}{\partial\langle M|}, \tag{S.44}$$

$$\nabla_{|B\rangle} = -\langle B'|\frac{\partial C}{\partial\langle M|}, \tag{S.45}$$

$$\nabla_{\ell} = -\left\langle \hat{l}_A\right|\frac{\partial C}{\partial\langle L|}, \tag{S.46}$$

$$\nabla_{\gamma} = -\left\langle \hat{l}_\gamma\right|\frac{\partial C}{\partial\langle L|}, \tag{S.47}$$

these are **scalar** quantities, see (S.15, S.16).[6]

[6]The minus sign makes things more intuitive...

3.1.4 α and ℓ dynamics

Using the above, with assumptions I and II, we get

$$\alpha_{|A\rangle}(t+\delta t) = N(t)\Big(\alpha_{|A\rangle}(t) + \delta t \frac{\nabla_{|A\rangle}}{|\nabla_{|A\rangle}| + |\nabla_{|B\rangle}|}\Big), \tag{S.48}$$

$$\alpha_{|B\rangle}(t+\delta t) = N(t)\Big(\alpha_{|B\rangle}(t) + \delta t \frac{\nabla_{|B\rangle}}{|\nabla_{|A\rangle}| + |\nabla_{|B\rangle}|}\Big), \tag{S.49}$$

$$\ell(t+\delta t) = \mathcal{C}\Big(\ell(t) + \delta t \frac{\nabla_{\ell}}{\max(|\nabla_{\ell}|, |\nabla_{\gamma}|)}\Big), \tag{S.50}$$

where $N(t)$ can also be simplified with assumption II,

$$N^{-1}(t) = \max(1, \Big|\alpha_{|A\rangle}(t) + \delta t \frac{\nabla_{|A\rangle}}{|\nabla_{|A\rangle}| + |\nabla_{|B\rangle}|}\Big| + \Big|\alpha_{|B\rangle}(t) + \delta t \frac{\nabla_{|B\rangle}}{|\nabla_{|A\rangle}| + |\nabla_{|B\rangle}|}\Big|). \tag{S.51}$$

3.1.5 Gradient descent quantities (expanded)

Using (S.15, S.16) and assumption I we can write,

$$\begin{aligned}\nabla_{|A\rangle} = &\left(1 - \left\langle \hat{l}_A \middle| o(A) \right\rangle\right)^{2n-1} \left(1 - \left\langle \hat{l}_A \middle| o(A) \right\rangle^2\right) \Big(\frac{\langle M|A\rangle}{\tilde{T}}\Big)^{n-1} \ell \\ &+ 4\left(1 + \left\langle \hat{l}_\gamma \middle| o(A) \right\rangle\right)^{2n-1} \left(1 - \left\langle \hat{l}_\gamma \middle| o(A) \right\rangle^2\right) \Big(\frac{\langle M|A\rangle}{\tilde{T}}\Big)^{n-1},\end{aligned} \tag{S.52}$$

$$\begin{aligned}\nabla_{|B\rangle} = &-\left(1 + \left\langle \hat{l}_A \middle| o(B) \right\rangle\right)^{2n-1} \left(1 - \left\langle \hat{l}_A \middle| o(B) \right\rangle^2\right) \Big(\frac{\langle M|B\rangle}{\tilde{T}}\Big)^{n-1} \ell \\ &+ 4\left(1 + \left\langle \hat{l}_\gamma \middle| o(B) \right\rangle\right)^{2n-1} \left(1 - \left\langle \hat{l}_\gamma \middle| o(B) \right\rangle^2\right) \Big(\frac{\langle M|B\rangle}{\tilde{T}}\Big)^{n-1},\end{aligned} \tag{S.53}$$

$$\begin{aligned}\nabla_{\ell} = &\left(1 - \left\langle \hat{l}_A \middle| o(A) \right\rangle\right)^{2n-1} \left(1 - \left\langle \hat{l}_A \middle| o(A) \right\rangle^2\right) \Big(\frac{\langle M|A\rangle}{\tilde{T}}\Big)^{n} \\ &- \left(1 + \left\langle \hat{l}_A \middle| o(B) \right\rangle\right)^{2n-1} \left(1 - \left\langle \hat{l}_A \middle| o(B) \right\rangle^2\right) \Big(\frac{\langle M|B\rangle}{\tilde{T}}\Big)^{n},\end{aligned} \tag{S.54}$$

$$\begin{aligned}\nabla_{\gamma} = &-\left(1 + \left\langle \hat{l}_\gamma \middle| o(A) \right\rangle\right)^{2n-1} \left(1 - \left\langle \hat{l}_\gamma \middle| o(A) \right\rangle^2\right) \Big(\frac{\langle M|A\rangle}{\tilde{T}}\Big)^{n} \\ &- \left(1 + \left\langle \hat{l}_\gamma \middle| o(B) \right\rangle\right)^{2n-1} \left(1 - \left\langle \hat{l}_\gamma \middle| o(B) \right\rangle^2\right) \Big(\frac{\langle M|B\rangle}{\tilde{T}}\Big)^{n}\end{aligned} \tag{S.55}$$

where time-dependence for α's, ℓ and $|M\rangle$ is implicit.

3.1.6 General behavior

Notice first, that $\nabla_{|A\rangle}, \nabla_{|B\rangle}$ cannot both be zero; one must be positive. Normalization is hence necessary to get a fixed point. Also from the gradient descent quantities, ℓ will tend to be positive when $\alpha_{|A\rangle} > \alpha_{|B\rangle}$ (conversely true), hence pushing the system towards $|\alpha_{|A\rangle}| + |\alpha_{|B\rangle}| = 1$. (see Fig. S.21 and Fig. S.22).

3.1.7 Normalization condition

To reiterate, normalization of the memory only happens when a pixel/element of the memory has amplitude greater than 1, otherwise we say there is no normalization or trivial normalization (which means $N(t) = 1$). If we look closer at $N(t)$, the condition for (non-trivial) normalization to occur is,

$$\left|\alpha_{|A\rangle}(t) + \delta t \frac{\nabla_{|A\rangle}}{|\nabla_{|A\rangle}| + |\nabla_{|B\rangle}|}\right| + \left|\alpha_{|B\rangle}(t) + \delta t \frac{\nabla_{|B\rangle}}{|\nabla_{|A\rangle}| + |\nabla_{|B\rangle}|}\right| > 1. \tag{S.56}$$

For very small δt we can write,

$$\left|\alpha_{|A\rangle}(t)\right| + \left|\alpha_{|B\rangle}(t)\right| + \frac{\left|\alpha_{|A\rangle}(t)\right|}{\alpha_{|A\rangle}(t)} \delta t \frac{\nabla_{|A\rangle}}{|\nabla_{|A\rangle}| + |\nabla_{|B\rangle}|} + \frac{\left|\alpha_{|B\rangle}(t)\right|}{\alpha_{|B\rangle}(t)} \delta t \frac{\nabla_{|B\rangle}}{|\nabla_{|A\rangle}| + |\nabla_{|B\rangle}|} > 1, \tag{S.57}$$

this is essentially a first order expansion of the absolute value, and works for small enough δt and non-zero α's. The special case where one of the α's is exactly zero is simple enough, omitted here for conciseness.
In the limit of $\delta t \to 0$, (S.57) is true when:

$$\left|\alpha_{|A\rangle}(t)\right| + \left|\alpha_{|B\rangle}(t)\right| = 1, \tag{S.58}$$

$$\frac{\left|\alpha_{|A\rangle}(t)\right|}{\alpha_{|A\rangle}(t)} \nabla_{|A\rangle} + \frac{\left|\alpha_{|B\rangle}(t)\right|}{\alpha_{|B\rangle}(t)} \nabla_{|B\rangle} > 0. \tag{S.59}$$

To give some intuition, recall that because of normalization $\left|\alpha_{|A\rangle}(t)\right| + \left|\alpha_{|B\rangle}(t)\right| \leq 1$. In the case where $\left|\alpha_{|A\rangle}(t)\right| + \left|\alpha_{|B\rangle}(t)\right| < 1$ by the definition of the limit there always exists a δt small enough such that the LHS of (S.57) is less than 1. This leads to our requirement for the equality (S.58). Then if this equality is held we can substitute it into (S.57) and obtain (S.59) with some manipulations.

3.1.8 Continuous memory dynamics (no normalization)

If either of (S.58, S.59) are not held, then taking the $\lim_{\delta t \to 0}$ leads to

$$\frac{\partial \alpha_{|A\rangle}}{\partial t} = \frac{\nabla_{|A\rangle}}{|\nabla_{|A\rangle}| + |\nabla_{|B\rangle}|}, \tag{S.60}$$

$$\frac{\partial \alpha_{|B\rangle}}{\partial t} = \frac{\nabla_{|B\rangle}}{|\nabla_{|A\rangle}| + |\nabla_{|B\rangle}|} \tag{S.61}$$

As outlined before, this offers no memory nullclines; which would require both $\nabla_{|A\rangle}$ and $\nabla_{|B\rangle}$ to be zero at the same time.

3.1.9 Continuous memory dynamics (with normalization)

If both (S.58, S.59) are held, we can write

$$N(t) \approx 1 - \delta t \frac{|\alpha_{|A\rangle}(t)|}{\alpha_{|A\rangle}(t)} \frac{\nabla_{|A\rangle}}{|\nabla_{|A\rangle}| + |\nabla_{|B\rangle}|} - \delta t \frac{|\alpha_{|B\rangle}(t)|}{\alpha_{|A\rangle}(t)} \frac{\nabla_{|B\rangle}}{|\nabla_{|A\rangle}| + |\nabla_{|B\rangle}|} \tag{S.62}$$

Applying this to the dynamics, keeping only first order δt and taking the limit we get,

$$\frac{\partial \alpha_{|A\rangle}}{\partial t} = \frac{\nabla_{|A\rangle} - \alpha_{|A\rangle}(t)\Big(\frac{|\alpha_{|A\rangle}(t)|}{\alpha_{|A\rangle}(t)}\nabla_{|A\rangle} + \frac{|\alpha_{|B\rangle}(t)|}{\alpha_{|B\rangle}(t)}\nabla_{|B\rangle}\Big)}{|\nabla_{|A\rangle}| + |\nabla_{|B\rangle}|}, \tag{S.63}$$

$$\frac{\partial \alpha_{|B\rangle}}{\partial t} = \frac{\nabla_{|B\rangle} - \alpha_{|B\rangle}(t)\Big(\frac{|\alpha_{|A\rangle}(t)|}{\alpha_{|A\rangle}(t)}\nabla_{|A\rangle} + \frac{|\alpha_{|B\rangle}(t)|}{\alpha_{|B\rangle}(t)}\nabla_{|B\rangle}\Big)}{|\nabla_{|A\rangle}| + |\nabla_{|B\rangle}|}. \tag{S.64}$$

These can, and do for the right ℓ and α's, lead to points/curve where $\frac{\partial \alpha_{|A\rangle}}{\partial t} = \frac{\partial \alpha_{|B\rangle}}{\partial t} = 0$. Importantly, we can show from (S.63, S.64) that $\frac{\partial |\alpha_{|A\rangle}|}{\partial t} = -\frac{\partial |\alpha_{|B\rangle}|}{\partial t}$, although this comes indirectly from normalization, it means that if $\frac{\partial \alpha_{|A\rangle}}{\partial t} = 0$, then $\frac{\partial \alpha_{|B\rangle}}{\partial t} = 0$.

Also while the normalization conditions are held, our memories are effectively one-dimensional and can be described by

$$\alpha_{|A\rangle} = \alpha, \tag{S.65}$$

$$\alpha_{|B\rangle} = \pm\Big(1 - |\alpha|\Big), \tag{S.66}$$

where $-1 \leq \alpha \leq 1$. Essentially, we have two one-dimensional branches for the memory, one where $\alpha_{|B\rangle}$ is positive and the other for negative $\alpha_{|B\rangle}$. Generally we need to examine these two branches independently for fixed points, but for intermediate and high n all fixed points are held in the positive branch. The above should also give intuition as to why $\frac{\partial \alpha_{|A\rangle}}{\partial t} = 0 \iff \frac{\partial \alpha_{|B\rangle}}{\partial t} = 0$.

3.1.10 Continuous label dynamics

The same can be done for labels, for $\mathcal{C}(\cdot)$ to be non-trivial we require

$$|\ell| = 1, \tag{S.67}$$

$$\text{sign}(\nabla_\ell) = \text{sign}(\ell). \tag{S.68}$$

which leads to,

$$\frac{\partial \ell}{\partial t} = 0, \tag{S.69}$$

a label nullcline. When either of (S.67, S.68) is not satisfied then,

$$\frac{\partial \ell}{\partial t} = \frac{\nabla_\ell}{\max(|\nabla_\ell|, |\nabla_\gamma|)}, \tag{S.70}$$

which may (does) lead to nullclines as well.

3.1.11 Memory nullclines

When (S.58, S.59) are held, our nullclines can be written as

$$0 = \frac{\nabla_{|A\rangle} - \alpha_{|A\rangle}(t)\left(\frac{|\alpha_{|A\rangle}(t)|}{\alpha_{|A\rangle}(t)}\nabla_{|A\rangle} + \frac{|\alpha_{|B\rangle}(t)|}{\alpha_{|B\rangle}(t)}\nabla_{|B\rangle}\right)}{|\nabla_{|A\rangle}| + |\nabla_{|B\rangle}|}, \tag{S.71}$$

$$0 = \frac{\nabla_{|B\rangle} - \alpha_{|B\rangle}(t)\left(\frac{|\alpha_{|A\rangle}(t)|}{\alpha_{|A\rangle}(t)}\nabla_{|A\rangle} + \frac{|\alpha_{|B\rangle}(t)|}{\alpha_{|B\rangle}(t)}\nabla_{|B\rangle}\right)}{|\nabla_{|A\rangle}| + |\nabla_{|B\rangle}|}. \tag{S.72}$$

which are redundant - if one of the above is zero, the other must be zero. Either equation can be rearranged into,

$$0 = \frac{\nabla_{|A\rangle}}{\alpha_{|A\rangle}} - \frac{\nabla_{|B\rangle}}{\alpha_{|B\rangle}}, \tag{S.73}$$

or,

$$\frac{\nabla_{|A\rangle}}{\alpha_{|A\rangle}} = \frac{\nabla_{|B\rangle}}{\alpha_{|B\rangle}}. \tag{S.74}$$

3.1.12 Label nullclines

Label nullclines come either from (S.67, S.68) being both satisfied, or from

$$\nabla_\ell = 0. \tag{S.75}$$

3.1.13 The intersection

The boundary condition (S.58) makes the space essentially 2D (one α dimension and ℓ). The memory and label nullclines each define on dimensional curves and hence intersect at at one or many points; these are fixed points of the dynamics.

3.1.14 A simple case

It should be clear that each nullcline equation is numerically solvable to a curve on the $|\alpha_{|A\rangle}| + |\alpha_{|B\rangle}| = 1$ boundary. Generally (and in particular for very low-n), this means dealing with two branches of the boundary, where at most one of $\alpha_{|A\rangle}$ and $\alpha_{|B\rangle}$ is negative otherwise

the activation function makes things trivial. Beyond the low-n case, both α's will be positive at the fixed points, hence we only need to consider

$$\alpha_{|A\rangle} = \alpha,$$
$$\alpha_{|B\rangle} = 1 - \alpha,$$

for $0 \leq \alpha \leq 1$. One can show the dynamics lead to $\frac{\partial \alpha_{|A\rangle}}{\partial t} = -\frac{\partial \alpha_{|B\rangle}}{\partial t}$. It also drastically simplifies (S.59),

$$\nabla_{|A\rangle} + \nabla_{|B\rangle} > 0$$

which is always held on the memory nullcline, which are now defined by,

$$\frac{\alpha}{\nabla_{|A\rangle}} = \frac{1-\alpha}{\nabla_{|B\rangle}} \tag{S.76}$$

where it should be clear that denominators depend solely on (α, ℓ). For examples of memory and label nullclines, see Fig. S.20.

3.2 The purely numerical case : N_k digits

A similar theoretical framework is more difficult to achieve for larger systems, nonetheless the first saddle(s) can still be obtained in a computationally efficient way, by simulating a 1-memory system, with the same training set and with an effective temperature

$$\tilde{T} = \frac{T}{\sqrt[n]{N_k}}. \tag{S.77}$$

This can be done for Gaussian initial conditions, where the primary saddle will be found, or with various initial condition (using α coefficients) where other saddles will be found.

4 Splitting / The 2-memory system

Splitting is exhibited in a 2-memory system after reaching the saddle described in the previous section. We write for the labels $|L_+\rangle, |L_-\rangle$,

$$\langle l_A|L_+\rangle = \ell + \delta\ell = -\langle l_B|L_+\rangle, \tag{S.78}$$
$$\langle l_A|L_-\rangle = \ell - \delta\ell = -\langle l_B|L_-\rangle, \tag{S.79}$$
$$\langle l_\gamma|L_\pm\rangle = -1 \tag{S.80}$$

the only assumption here is that we are near the ℓ-manifold described in the 1-memory system. We can do a similar expansion for the memories, which generally are defined as

$$|M_\pm\rangle = \alpha_{|A\rangle,\pm}|A\rangle + \alpha_{|B\rangle,\pm}|B\rangle.$$

For simplicity assume $\alpha_{|A\rangle,\pm}, \alpha_{|B\rangle,\pm} > 0$, and $\alpha_{|A\rangle,\pm} + \alpha_{|B\rangle,\pm} = 1$ - this holds if we are near a (positive) 1-memory fixed point. Then, we can write

$$|M_\pm\rangle = (\alpha \pm \delta\alpha)|A\rangle + (1 - \alpha \mp \delta\alpha)|B\rangle. \tag{S.81}$$

Up to first order in $\delta\alpha$ and $\delta\ell$, the output is equal to the 1-memory case,

$$|o(\sigma)\rangle \approx \tanh\left(|L\rangle\left(\frac{\langle M|\sigma\rangle}{\tilde{T}}\right)^n\right), \tag{S.82}$$

with

$$\tilde{T} = \frac{T}{\sqrt[n]{2}}, \tag{S.83}$$

and

$$|M\rangle = |M_\pm\rangle\Big|_{\delta\alpha=0}, \tag{S.84}$$

$$|L\rangle = |L_\pm\rangle\Big|_{\delta\ell=0}. \tag{S.85}$$

For the δ-quantities we first write,

$$\langle M_\pm|\sigma\rangle^n \approx \langle M|\sigma\rangle^n \pm n\langle M|\sigma\rangle^{n-1}\frac{\partial\langle M|\sigma\rangle}{\partial\alpha}\delta\alpha, \tag{S.86}$$

where

$$\frac{\partial\langle M|A\rangle}{\partial\alpha} = \langle A|A\rangle - \langle A|B\rangle, \tag{S.87}$$

$$\frac{\partial\langle M|B\rangle}{\partial\alpha} = \langle A|B\rangle - \langle B|B\rangle. \tag{S.88}$$

This leads to the following gradient descent quantities [7],

$$\begin{aligned}
\nabla_{|A\rangle,\pm} \approx & \left(1-\left\langle\hat{l}_A\middle|o(A)\right\rangle\right)^{2n-1}\left(1-\left\langle\hat{l}_A\middle|o(A)\right\rangle^2\right)\langle M|A\rangle^{n-1}\,\ell \\
& + 4\left(1+\left\langle\hat{l}_\gamma\middle|o(A)\right\rangle\right)^{2n-1}\left(1-\left\langle\hat{l}_\gamma\middle|o(A)\right\rangle^2\right)\langle M|A\rangle^{n-1} \\
& \pm (n-1)\left(1-\left\langle\hat{l}_A\middle|o(A)\right\rangle\right)^{2n-1}\left(1-\left\langle\hat{l}_A\middle|o(A)\right\rangle^2\right)\langle M|A\rangle^{n-2}\frac{\partial\langle M|A\rangle}{\partial\alpha}\ell\delta\alpha \\
& \pm 4(n-1)\left(1+\left\langle\hat{l}_\gamma\middle|o(A)\right\rangle\right)^{2n-1}\left(1-\left\langle\hat{l}_\gamma\middle|o(A)\right\rangle^2\right)\langle M|A\rangle^{n-2}\frac{\partial\langle M|A\rangle}{\partial\alpha}\delta\alpha \\
& \pm \left(1-\left\langle\hat{l}_A\middle|o(A)\right\rangle\right)^{2n-1}\left(1-\left\langle\hat{l}_A\middle|o(A)\right\rangle^2\right)\langle M|A\rangle^{n-1}\,\delta\ell
\end{aligned} \tag{S.89}$$

[7] This is a shorthand where $\sigma \in \{A, B\}$

$$
\begin{aligned}
\nabla_{|B\rangle,\pm} \approx & -\left(1+\left\langle \hat{l}_A\middle|o(B)\right\rangle\right)^{2n-1}\left(1-\left\langle \hat{l}_A\middle|o(B)\right\rangle^2\right)\langle M|B\rangle^{n-1}\,\ell \\
& +4\left(1+\left\langle \hat{l}_\gamma\middle|o(B)\right\rangle\right)^{2n-1}\left(1-\left\langle \hat{l}_\gamma\middle|o(B)\right\rangle^2\right)\langle M|B\rangle^{n-1} \\
& \mp(n-1)\left(1+\left\langle \hat{l}_A\middle|o(B)\right\rangle\right)^{2n-1}\left(1-\left\langle \hat{l}_A\middle|o(B)\right\rangle^2\right)\langle M|B\rangle^{n-2}\,\frac{\partial\langle M|B\rangle}{\partial\alpha}\ell\delta\alpha \\
& \pm 4(n-1)\left(1+\left\langle \hat{l}_\gamma\middle|o(B)\right\rangle\right)^{2n-1}\left(1-\left\langle \hat{l}_\gamma\middle|o(B)\right\rangle^2\right)\langle M|B\rangle^{n-2}\,\frac{\partial\langle M|B\rangle}{\partial\alpha}\delta\alpha \\
& \mp\left(1+\left\langle \hat{l}_A\middle|o(B)\right\rangle\right)^{2n-1}\left(1-\left\langle \hat{l}_A\middle|o(B)\right\rangle^2\right)\langle M|B\rangle^{n-1}\,\delta\ell
\end{aligned}
\tag{S.90}
$$

$$
\begin{aligned}
\nabla_{\ell,\pm} \approx & \left(1-\left\langle \hat{l}_A\middle|o(A)\right\rangle\right)^{2n-1}\left(1-\left\langle \hat{l}_A\middle|o(A)\right\rangle^2\right)\langle M|A\rangle^{n} \\
& -\left(1+\left\langle \hat{l}_A\middle|o(B)\right\rangle\right)^{2n-1}\left(1-\left\langle \hat{l}_A\middle|o(B)\right\rangle^2\right)\langle M|B\rangle^{n} \\
& \pm n\left(1-\left\langle \hat{l}_A\middle|o(A)\right\rangle\right)^{2n-1}\left(1-\left\langle \hat{l}_A\middle|o(A)\right\rangle^2\right)\langle M|A\rangle^{n-1}\,\frac{\partial\langle M|A\rangle}{\partial\alpha}\delta\alpha \\
& \mp n\left(1+\left\langle \hat{l}_A\middle|o(B)\right\rangle\right)^{2n-1}\left(1-\left\langle \hat{l}_A\middle|o(B)\right\rangle^2\right)\langle M|B\rangle^{n-1}\,\frac{\partial\langle M|B\rangle}{\partial\alpha}\delta\alpha
\end{aligned}
\tag{S.91}
$$

keeping only first order in δ. Our normalization condition becomes

$$
\nabla_{|A\rangle,\pm}+\nabla_{|B\rangle,\pm}>0, \tag{S.92}
$$

for our initial assumptions to hold, this inequality must be held. Then,

$$
\frac{\partial(\alpha\pm\delta\alpha)}{\partial t}\approx\frac{\nabla_{|A\rangle,\pm}-\Big(\alpha\pm\delta\alpha\Big)\Big(\nabla_{|A\rangle,\pm}+\nabla_{|B\rangle,\pm}\Big)}{|\nabla_{|A\rangle,\pm}|+|\nabla_{|B\rangle,\pm}|}, \tag{S.93}
$$

again keeping only first order terms,

$$
\frac{\partial(\alpha)}{\partial t}\pm\frac{\partial(\delta\alpha)}{\partial t}\approx\frac{\nabla_{|A\rangle,\pm}-\alpha\Big(\nabla_{|A\rangle,\pm}+\nabla_{|B\rangle,\pm}\Big)\mp\delta\alpha\Big(\nabla_{|A\rangle}+\nabla_{|B\rangle}\Big)}{|\nabla_{|A\rangle,\pm}|+|\nabla_{|B\rangle,\pm}|}. \tag{S.94}
$$

Where $\nabla_{|A\rangle},\nabla_{|B\rangle}$ represent the 1-memory quantities; equivalent to setting $\delta\alpha=\delta\ell=0$ in $\nabla_{|\sigma\rangle,\pm}$. From the above one can recover $\partial_t\alpha$ and $\partial_t\delta\alpha$. As for ℓ, we assume $\ell<1$ and obtain,

$$
\frac{\partial(\ell)}{\partial t}\pm\frac{\partial(\delta\ell)}{\partial t}\approx\frac{\nabla_{\ell,\pm}}{\max(|\nabla_{\ell,\pm}|,|\nabla_\gamma|)}. \tag{S.95}
$$

Further approximations are available (notably for the denominators). This first-order approximation works particularly well near a 1-memory fixed point, as shown in Fig. S.24, and Fig. S.25.

5 Final states / The 2-memory system

The 2-memory system can also be used to understand the final states. Consider a system which has already split,

$$\left\langle \hat{l}_A \middle| L_A \right\rangle = 1 = \left\langle \hat{l}_B \middle| L_B \right\rangle, \tag{S.96}$$

$$\left\langle \hat{l}_B \middle| L_A \right\rangle = \left\langle \hat{l}_A \middle| L_B \right\rangle = -1 = \langle l_\gamma | L_A \rangle = \langle l_\gamma | L_B \rangle . \tag{S.97}$$

As for the memory,

$$|M_A\rangle = \Big(1 - |\delta_A|\Big)\,|A\rangle + \delta_A\,|B\rangle\,, \tag{S.98}$$

$$|M_B\rangle = \delta_B\,|A\rangle + \Big(1 - |\delta_B|\Big)\,|B\rangle\,. \tag{S.99}$$

The output reduces to,

$$\left\langle \hat{l}_A \middle| o(\sigma) \right\rangle = \tanh\left(\Big(\frac{\langle M_A|\sigma\rangle}{T}\Big)^n - \Big(\frac{\langle M_B|\sigma\rangle}{T}\Big)^n \right) = -\left\langle \hat{l}_B \middle| o(\sigma) \right\rangle, \tag{S.100}$$

$$\left\langle \hat{l}_\gamma \middle| o(\sigma) \right\rangle = \tanh\left(-\Big(\frac{\langle M_A|\sigma\rangle}{T}\Big)^n - \Big(\frac{\langle M_B|\sigma\rangle}{T}\Big)^n \right) \tag{S.101}$$

and the cost function becomes,

$$\begin{aligned} C = &\left(1 - \left\langle \hat{l}_A \middle| o(A) \right\rangle\right)^{2n} + 4\left(1 + \left\langle \hat{l}_\gamma \middle| o(A) \right\rangle\right)^{2n} \\ &+ \left(1 + \left\langle \hat{l}_A \middle| o(B) \right\rangle\right)^{2n} + 4\left(1 + \left\langle \hat{l}_\gamma \middle| o(B) \right\rangle\right)^{2n}. \end{aligned} \tag{S.102}$$

As before, we define

$$\nabla_{|\sigma\rangle,\mu} = -\langle \sigma' | \frac{\partial C}{\partial \langle M_\mu |}, \tag{S.103}$$

$$\langle \sigma' | M_\mu \rangle = \alpha_{\sigma,\mu} \tag{S.104}$$

and we can write,

$$\partial_t \alpha_{\sigma,\mu} = \frac{\nabla_{|\sigma\rangle,\mu} - \alpha_{\sigma,\mu}\Big(\frac{|\alpha_{A,\mu}|}{\alpha_{A,\mu}}\nabla_{|A\rangle,\mu} + \frac{|\alpha_{B,\mu}|}{\alpha_{B,\mu}}\nabla_{|B\rangle,\mu}\Big)}{|\nabla_{|A\rangle,\mu}| + |\nabla_{|B\rangle,\mu}|}, \tag{S.105}$$

which leads to the following fixed point,

$$\frac{\alpha_{A,\mu}}{\nabla_{|A\rangle,\mu}} = \frac{\alpha_{B,\mu}}{\nabla_{|B\rangle,\mu}}. \tag{S.106}$$

If we combine this with our initial assumption, we get the following system of equation

$$\begin{cases} \frac{1-|\delta_A|}{\nabla_{|A\rangle,A}} = \frac{\delta_A}{\nabla_{|B\rangle,A}} \\ \frac{\delta_B}{\nabla_{|A\rangle,B}} = \frac{1-|\delta_B|}{\nabla_{|B\rangle,B}} \end{cases} . \tag{S.107}$$

Two variables, two equations; numerically solvable.

6 Other models

6.1 The intraclass feature-to-prototype transition

Consider a 1-memory system with two training samples $|A_1\rangle, |A_2\rangle$ of the same class A. The label can always be written as

$$\left\langle \hat{l}_A \middle| L \right\rangle = 1 = -\left\langle \hat{l}_\gamma \middle| L \right\rangle, \tag{S.108}$$

for any $\gamma \neq A$. For the memory, we define two strategies the system may implement, the generalist and the specialist.

$$|M_g\rangle = \frac{|A_1\rangle + |A_2\rangle}{2}, \tag{S.109}$$

$$|M_s\rangle = |A_1\rangle \tag{S.110}$$

where the choice of either $|A_1\rangle$ or $|A_2\rangle$ in the specialist strategy depends strongly on the initial conditions. For simplicity, the generalist case is written for $\langle A_1|A_1\rangle = \langle A_2|A_2\rangle$.
The output of each strategy is,

$$\left\langle \hat{l}_A \middle| o(\sigma) \right\rangle_g = \tanh\left(\left(\frac{\langle A_1|\sigma\rangle + \langle A_2|\sigma\rangle}{2T} \right)^n \right) = -\left\langle \hat{l}_\gamma \middle| o(\sigma) \right\rangle_g, \tag{S.111}$$

$$\left\langle \hat{l}_A \middle| o(\sigma) \right\rangle_s = \tanh\left(\left(\frac{\langle A_1|\sigma\rangle}{T} \right)^n \right) = -\left\langle \hat{l}_\gamma \middle| o(\sigma) \right\rangle_s. \tag{S.112}$$

The cost function of a 1-memory system (with labels as defined previously) is,

$$C = 10\left(1 - \left\langle \hat{l}_A \middle| o(A_1) \right\rangle\right)^{2n} + 10\left(1 - \left\langle \hat{l}_A \middle| o(A_2) \right\rangle\right)^{2n}, \tag{S.113}$$

where the factors of 10, have been kept to hint at symmetry involved. For each strategy we write,

$$C_g = 2\left(1 - \tanh\left(\left(\frac{\langle A_1|A_1\rangle + \langle A_1|A_2\rangle}{2T} \right)^n \right)\right)^{2n}, \tag{S.114}$$

$$C_s = \left(1 - \tanh\left(\left(\frac{\langle A_1|A_1\rangle}{T} \right)^n \right)\right)^{2n} + \left(1 - \tanh\left(\left(\frac{\langle A_1|A_2\rangle}{T} \right)^n \right)\right)^{2n}, \tag{S.115}$$

where the assumption $\langle A_1|A_1\rangle = \langle A_2|A_2\rangle$ was used to simplify C_g and overall (10) factors are removed. Now we approximate that in the region where C_s is valid, $C_s \approx 1$. [8] We say the curve where both strategies lead to the same cost/energy is the transition curve,

$$C_g = C_s \approx 1. \tag{S.116}$$

This leads to,

$$T \approx \frac{\langle A_1|A_1\rangle + \langle A_1|A_2\rangle}{2\sqrt[n]{\operatorname{arctanh}\left(1 - \frac{1}{\sqrt[n]{2}}\right)}}. \tag{S.117}$$

6.2 Populations and final states

Consider a system of N_k memories, with two training samples $|A\rangle, |B\rangle$ of classes A, B. Such systems tend to have only two unique memories, $|M_A\rangle$ and $|M_B\rangle$, with other memories being clones of the latter.[9] In which case, the output can be written as,

$$\left\langle \hat{l}_A \middle| o(\sigma) \right\rangle = \tanh\left(N_A \left(\frac{\langle M_A|\sigma\rangle}{T}\right)^n - N_B \left(\frac{\langle M_B|\sigma\rangle}{T}\right)^n \right) = -\left\langle \hat{l}_B \middle| o(\sigma) \right\rangle, \tag{S.118}$$

$$\left\langle \hat{l}_\gamma \middle| o(\sigma) \right\rangle = -\tanh\left(N_A \left(\frac{\langle M_A|\sigma\rangle}{T}\right)^n + N_B \left(\frac{\langle M_B|\sigma\rangle}{T}\right)^n \right). \tag{S.119}$$

with of course,

$$N_A + N_B = N_k. \tag{S.120}$$

The cost function of the system is,

$$C = \left(1 - \left\langle \hat{l}_A \middle| o(A) \right\rangle\right)^{2n} + \left(1 + \left\langle \hat{l}_A \middle| o(B) \right\rangle\right)^{2n} \tag{S.121}$$

where $\left\langle \hat{l}_\gamma \middle| o(\sigma) \right\rangle \approx -1$ for $\sigma \in \{A, B\}$, which eliminated a few terms. There is a competition between the two classes, and we make a symmetry argument that the system will converge towards,

$$\left(1 - \left\langle \hat{l}_A \middle| o(A) \right\rangle\right)^{2n} = \left(1 + \left\langle \hat{l}_A \middle| o(B) \right\rangle\right)^{2n}. \tag{S.122}$$

[8] Explicitly, $\left(1 - \tanh\left(\left(\frac{\langle A_1|A_1\rangle}{T}\right)^n\right)\right)^{2n} \approx 0$ and, $\left(1 - \tanh\left(\left(\frac{\langle A_1|A_2\rangle}{T}\right)^n\right)\right)^{2n} \approx 1$; only one training sample is understood.

[9] This can be dealt with in a more nuanced way with average $\left|\bar{M}_A\right\rangle$ and neglecting deviations from the average.

This is equivalent to writing,

$$N_A \langle M_A|A\rangle^n - N_B \langle M_B|A\rangle^n = N_B \langle M_B|B\rangle^n - N_A \langle M_A|B\rangle^n , \tag{S.123}$$

which directly links the final $|M_A\rangle, |M_B\rangle$ states to the population proportions;

$$N_A = (N_k)\frac{\langle M_B|B\rangle^n + \langle M_B|A\rangle^n}{\langle M_A|A\rangle^n + \langle M_A|B\rangle^n + \langle M_B|B\rangle^n + \langle M_B|A\rangle^n}. \tag{S.124}$$

Knowing the final states of $|M_A\rangle$ and $|M_B\rangle$ tells you the population proportions.

7 UMAP representation

The UMAP embedding is generated using all 60000 training images of the MNIST training dataset. The correlation metric is used, as well as the following hyperparameters: random state (4), number of neighbours (55), minimum distance (0.05). The resulting UMAP object is saved and reused for all plots. Note, we see similar dynamics with the default UMAP metric, and other hyperparameter values.

When projecting the memory dynamics onto the UMAP landscape, the previously defined object is used and memories of the same epoch are projected together.

In movies, it can be difficult to track the trajectories of individual memories/points, to counter this we add linear interpolation frames, to make the trajectories easier to follow. Note however, that this has no effect on the general dynamics and is mainly a cosmetic modification.

8 Supplementary Figures

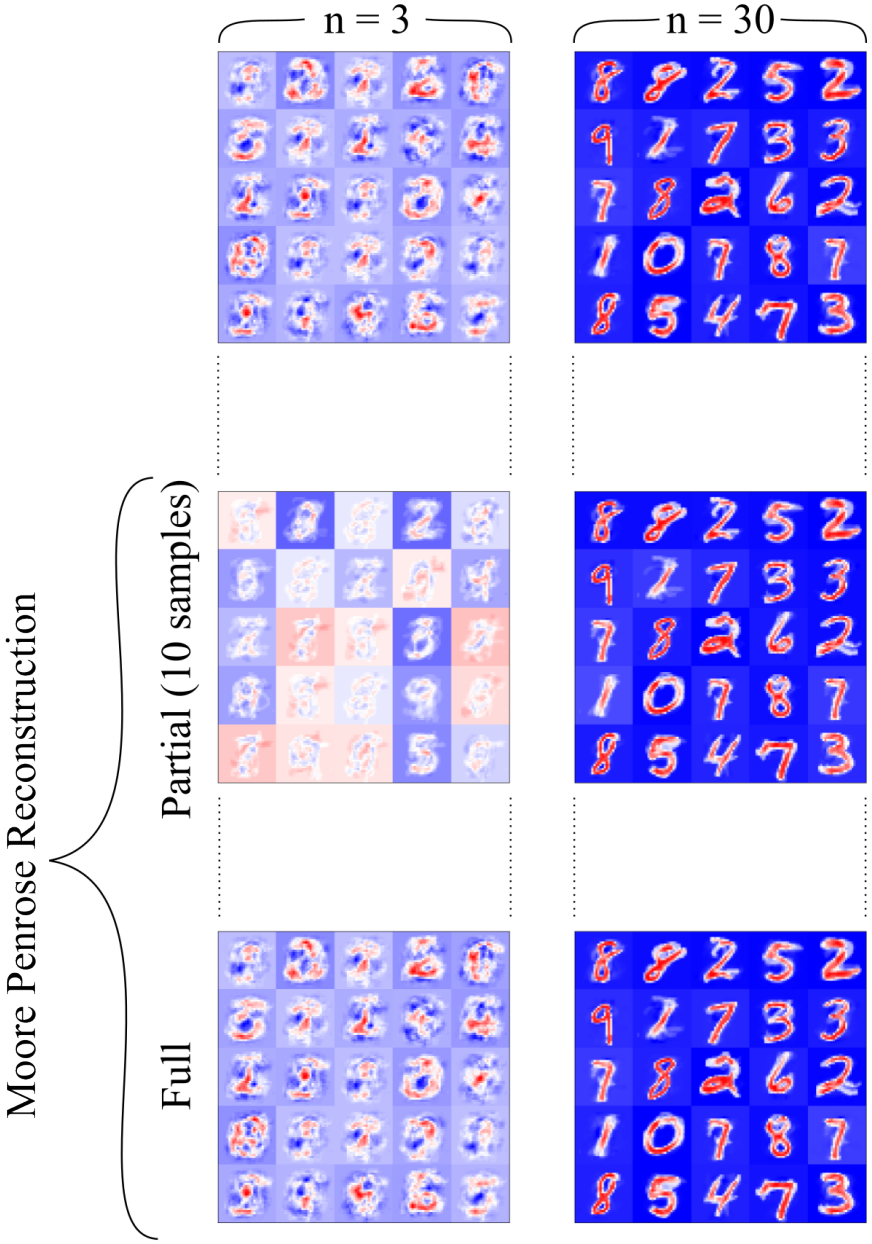

Figure S.1: The first row are sample memories from a 100-memory system training on 200 digits (20 for each class) as in Fig. 3. The rescaled temperature is 0.85, and $n = 3$ or $n = 30$. The second and third rows are Moore-Penrose based memory reconstructions. The second row includes only the 10 most dominating coefficients, the third row includes all coefficients.

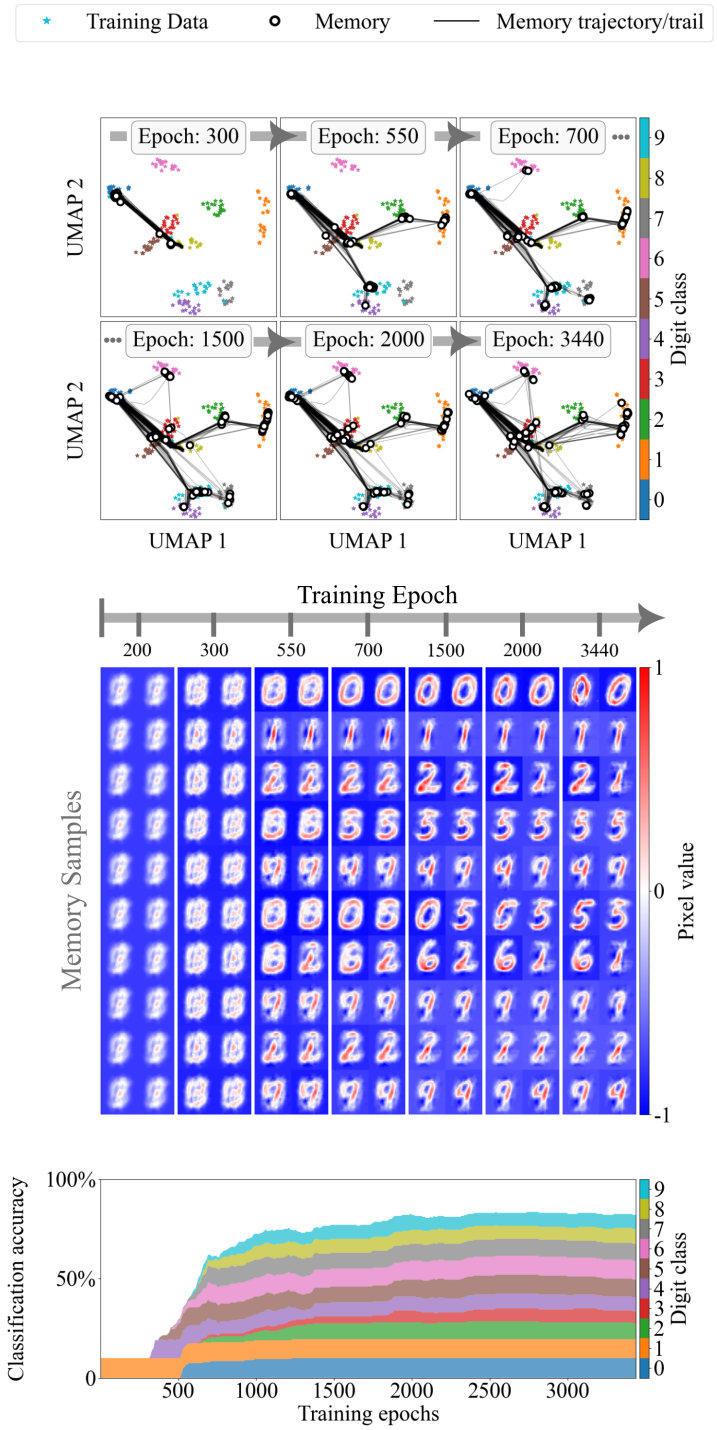

Figure S.2: A 100-memory system training on 200 digits (20 samples of each class) the same as in Fig. 3, the hyperparameters are $n = 15$, and $T_r = 0.85$.

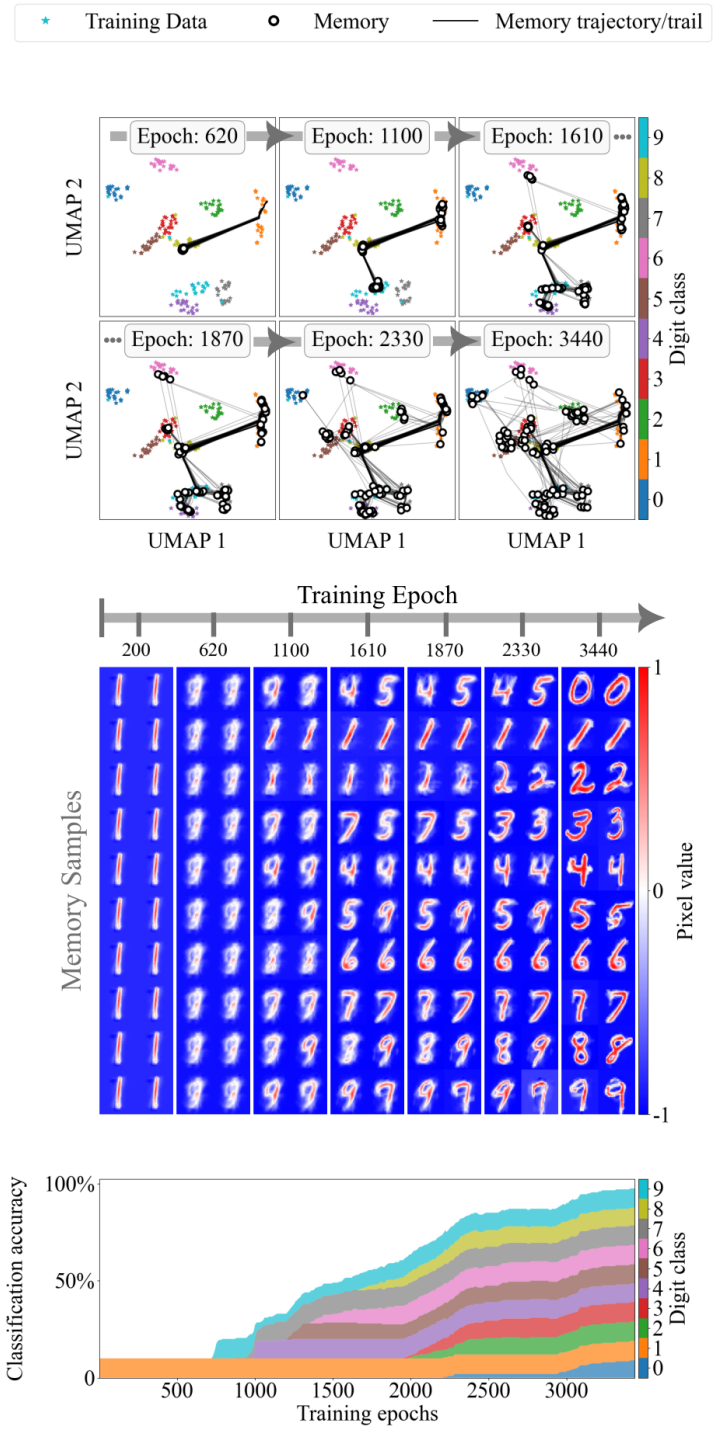

Figure S.3: A 100-memory system training on 200 digits (20 samples of each class) the same as in Fig. 3, the hyperparameters are $n = 40$, and $T_r = 0.85$.

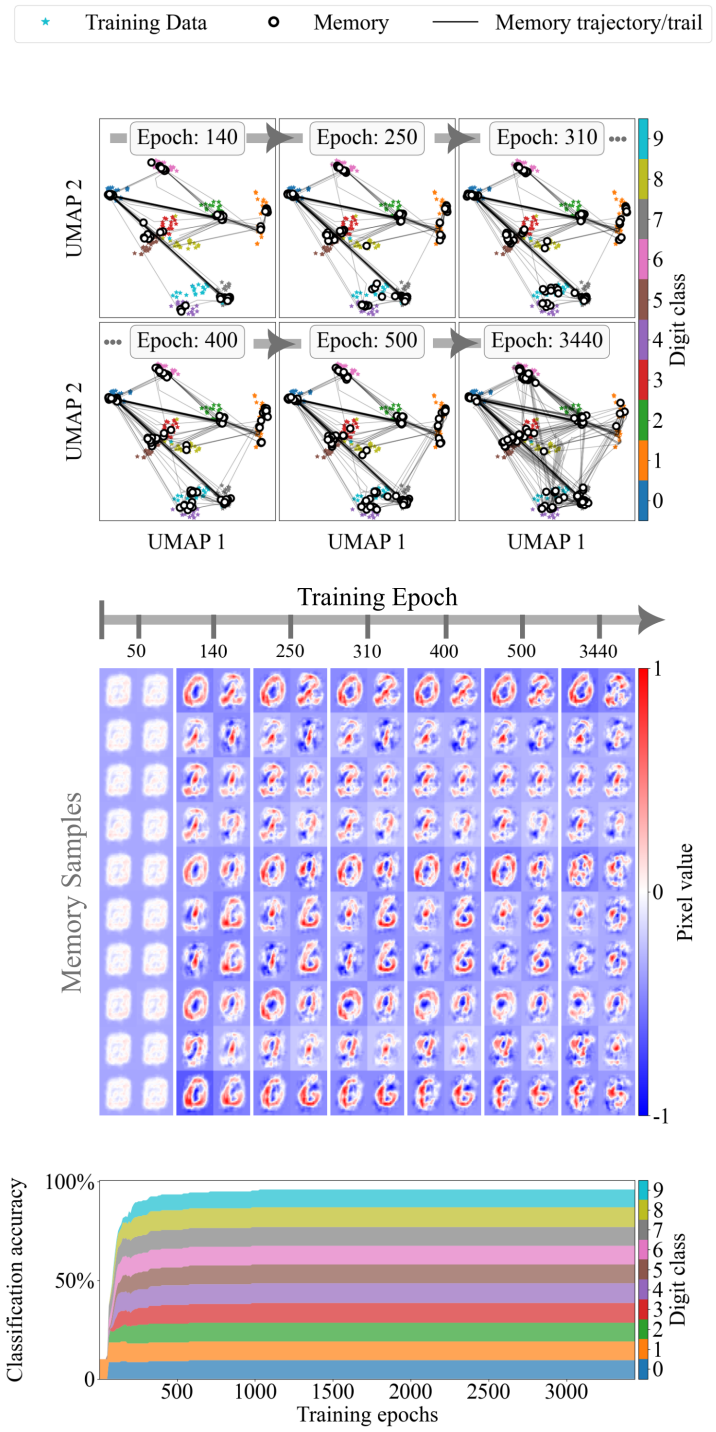

Figure S.4: A 100-memory system training on 200 digits (20 samples of each class) the same as in Fig. 3, the hyperparameters are $n = 3$, and $T_r = 0.85$, here however, the 200 digit training set is split into 4 miniBatchs each containing 50 digits (5 of each class).

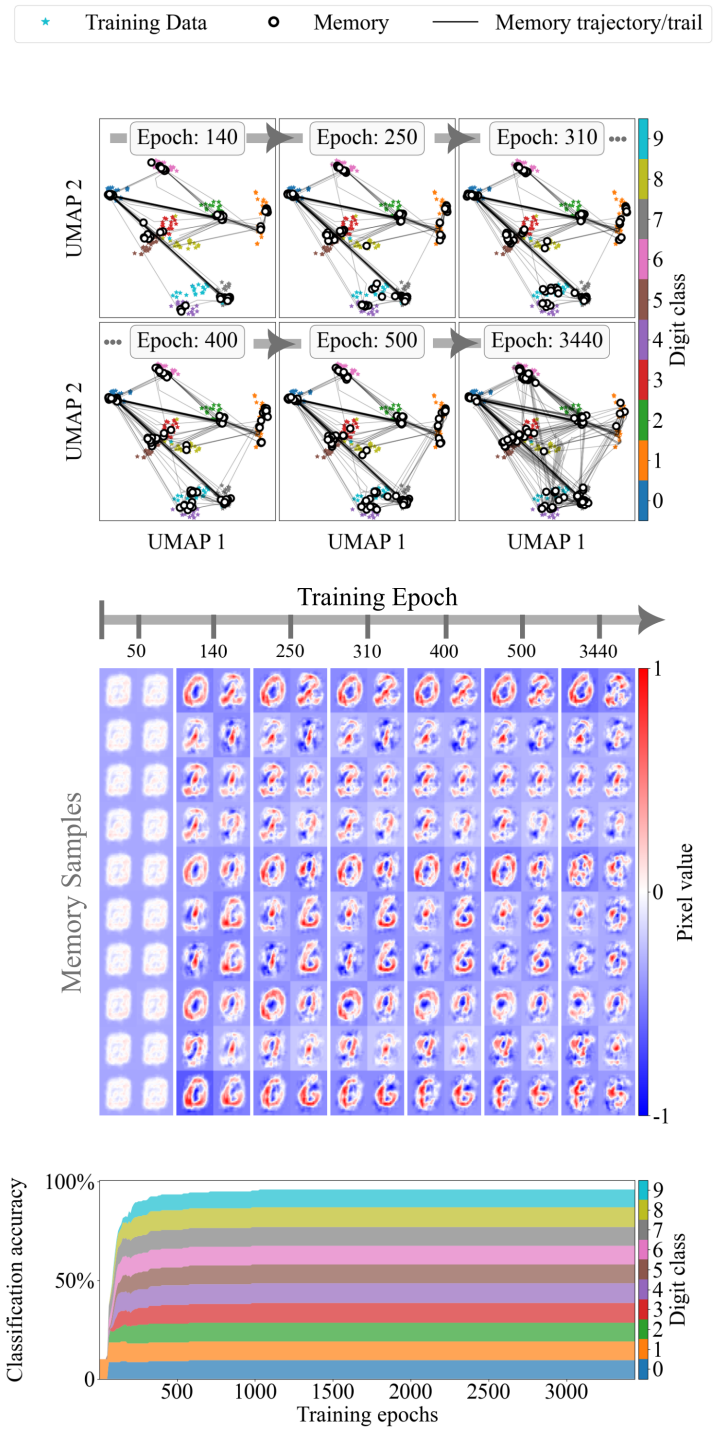

Figure S.5: A 100-memory system training on 200 digits (20 samples of each class) the same as in Fig. 3, the hyperparameters are $n = 3$, and $T_r = 0.85$, here however, the 200 digit training set is split into 4 miniBatchs each containing 50 digits (5 of each class).

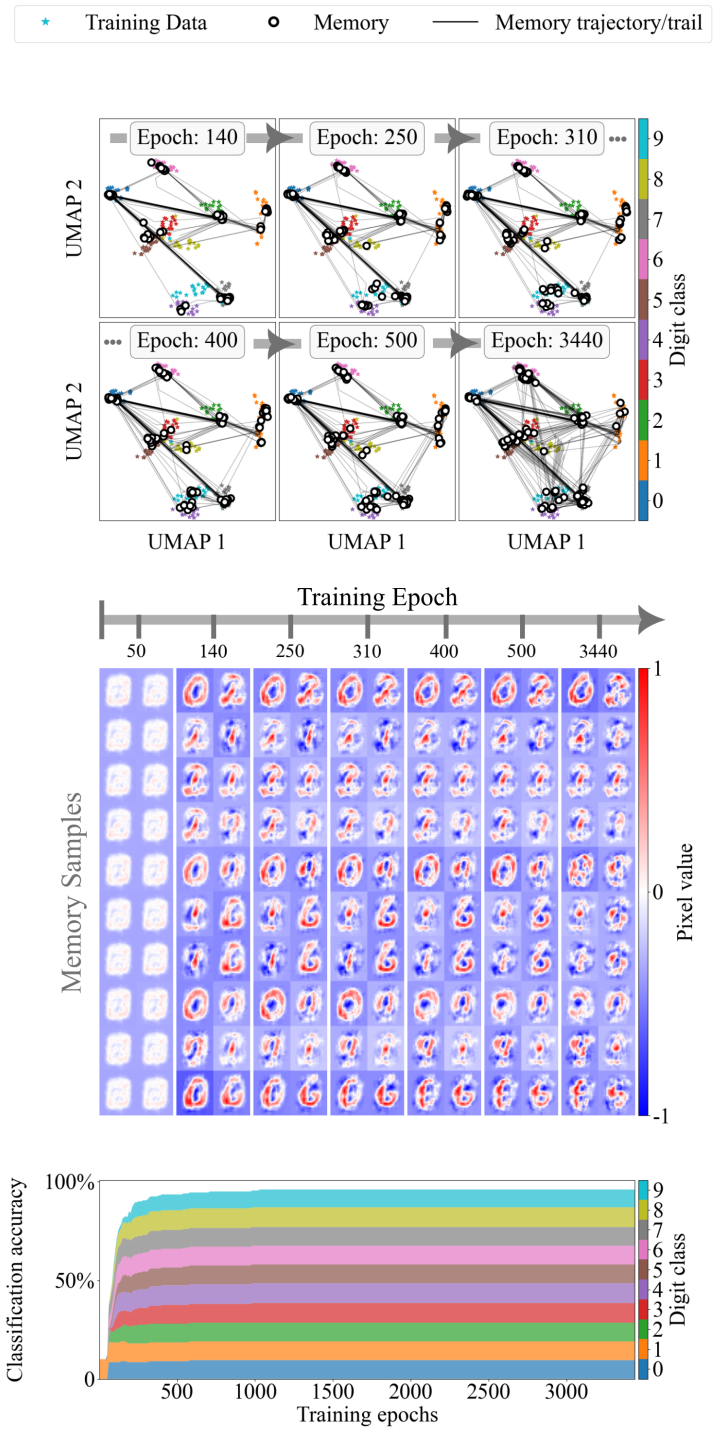

Figure S.6: A 100-memory system training on 200 digits (20 samples of each class) the same as in Fig. 3, the hyperparameters are $n = 3$, and $T_r = 0.85$, here however, the 200 digit training set is split into 4 miniBatchs each containing 50 digits (5 of each class).

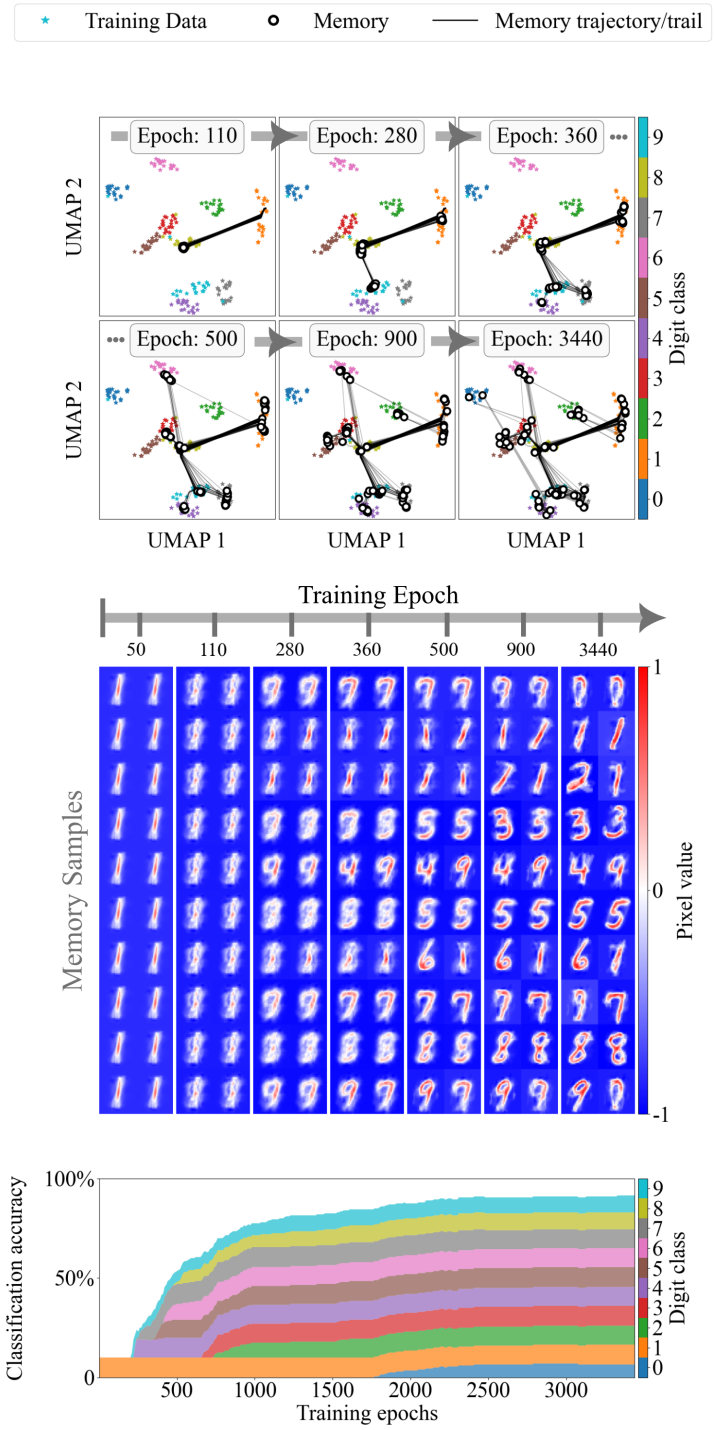

Figure S.7: A 100-memory system training on 200 digits (20 samples of each class) the same as in Fig. 3, the hyperparameters are $n = 30$, and $T_r = 0.85$, here however, the 200 digit training set is split into 4 miniBatchs each containing 50 digits (5 of each class).

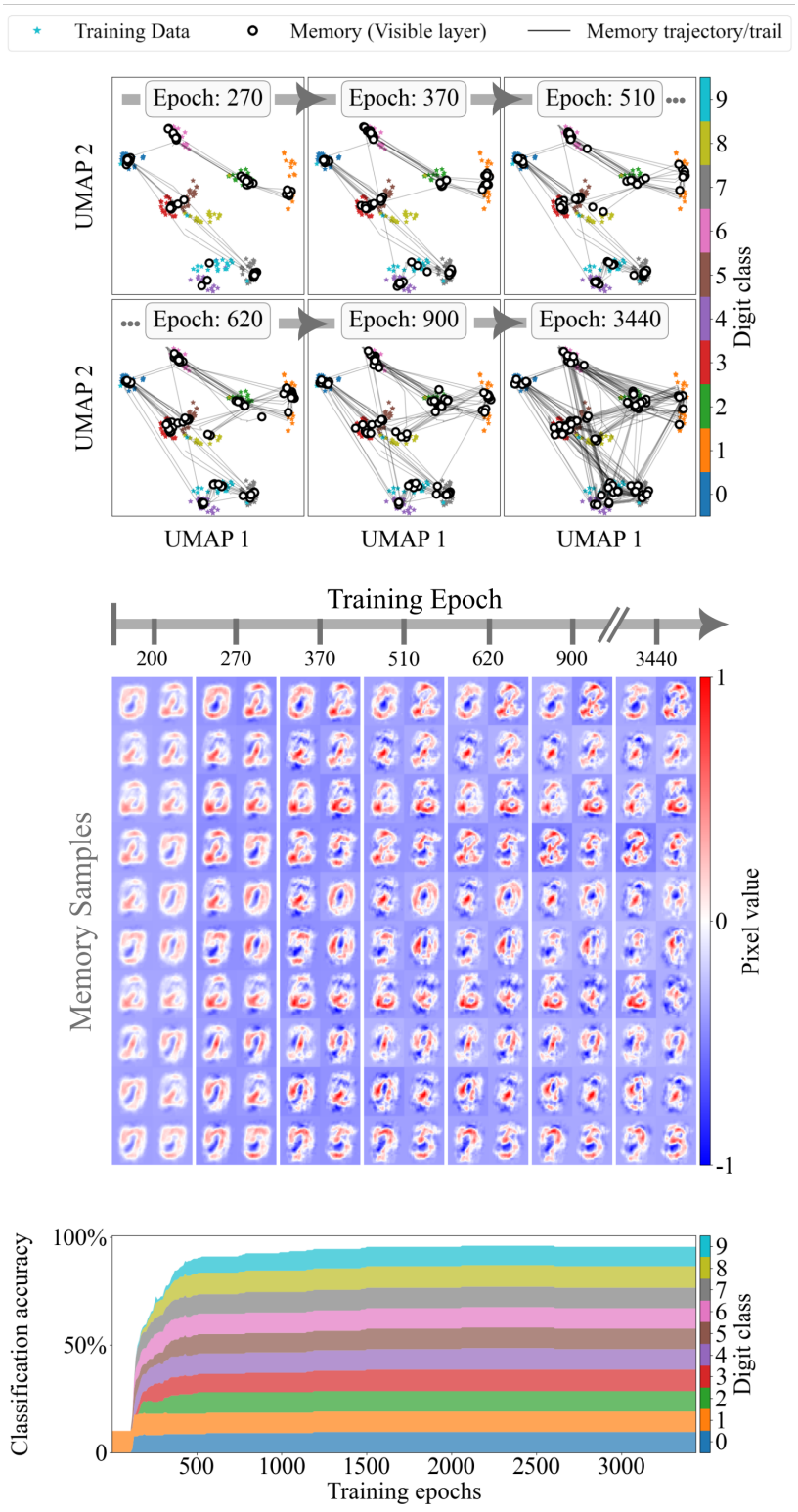

Figure S.8: A 100-memory system training on 200 digits (20 samples of each class) the same as in Fig. 3, the hyperparameters are $n = 3$, and $T_r = 0.85$, here however, with a momentum parameter set to 0.6.

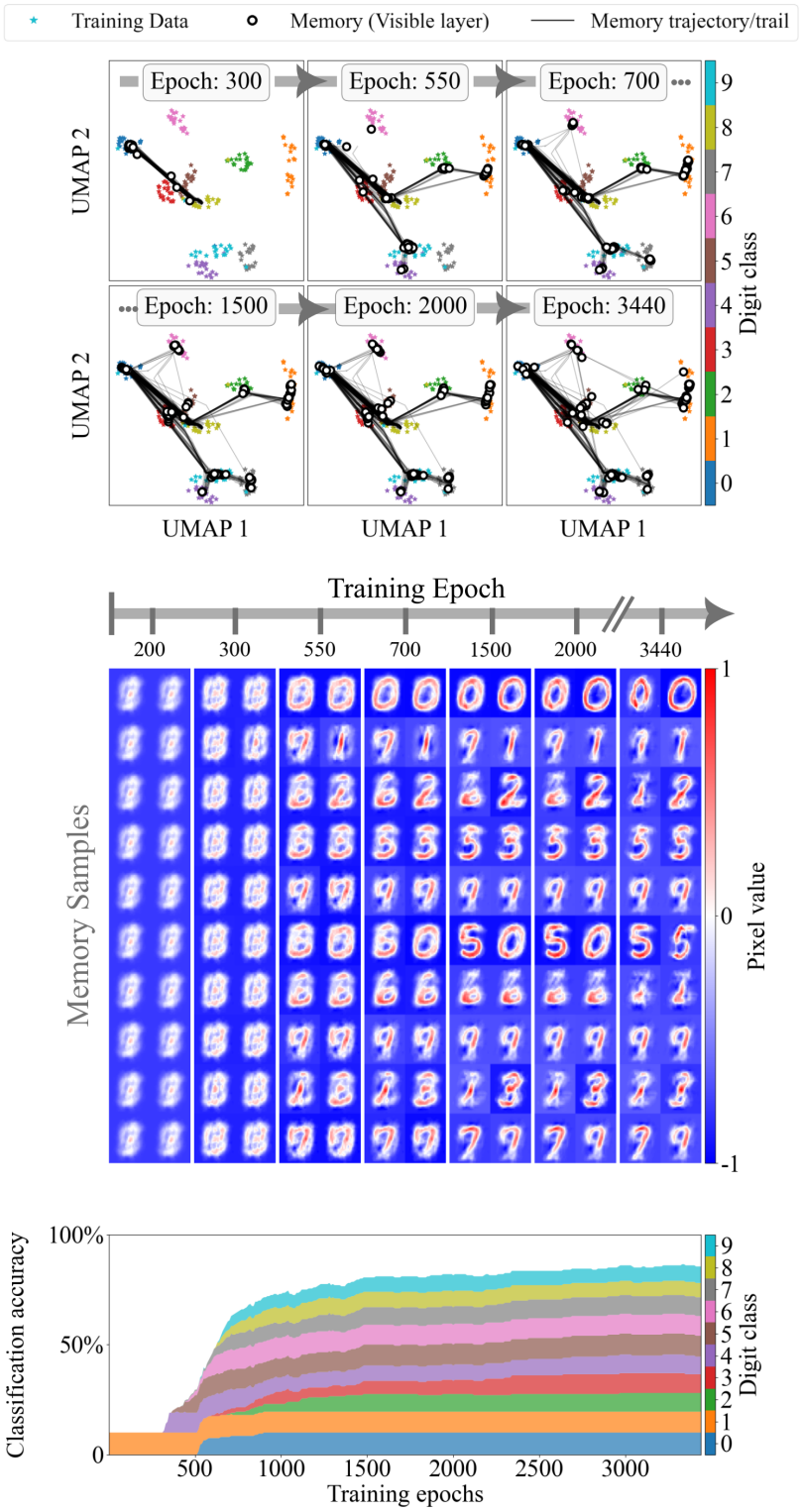

Figure S.9: A 100-memory system training on 200 digits (20 samples of each class) the same as in Fig. 3, the parameters are $n = 15$, and $T_r = 0.85$, here however, with a momentum parameter set to 0.6.

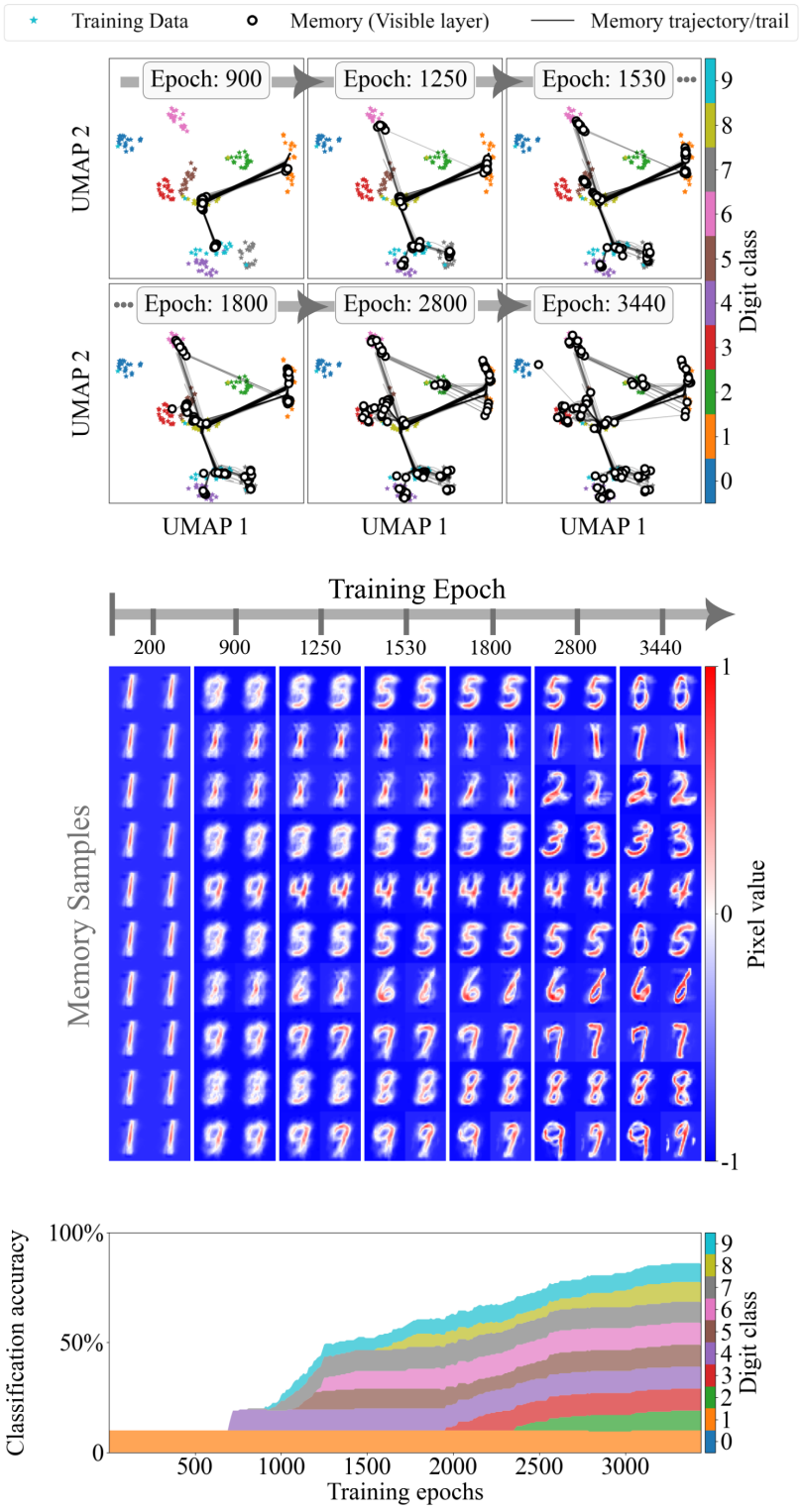

Figure S.10: A 100-memory system training on 200 digits (20 samples of each class) the same as in Fig. 3, the hyperparameters are $n = 30$, and $T_r = 0.85$, here however, with a momentum parameter set to 0.6.

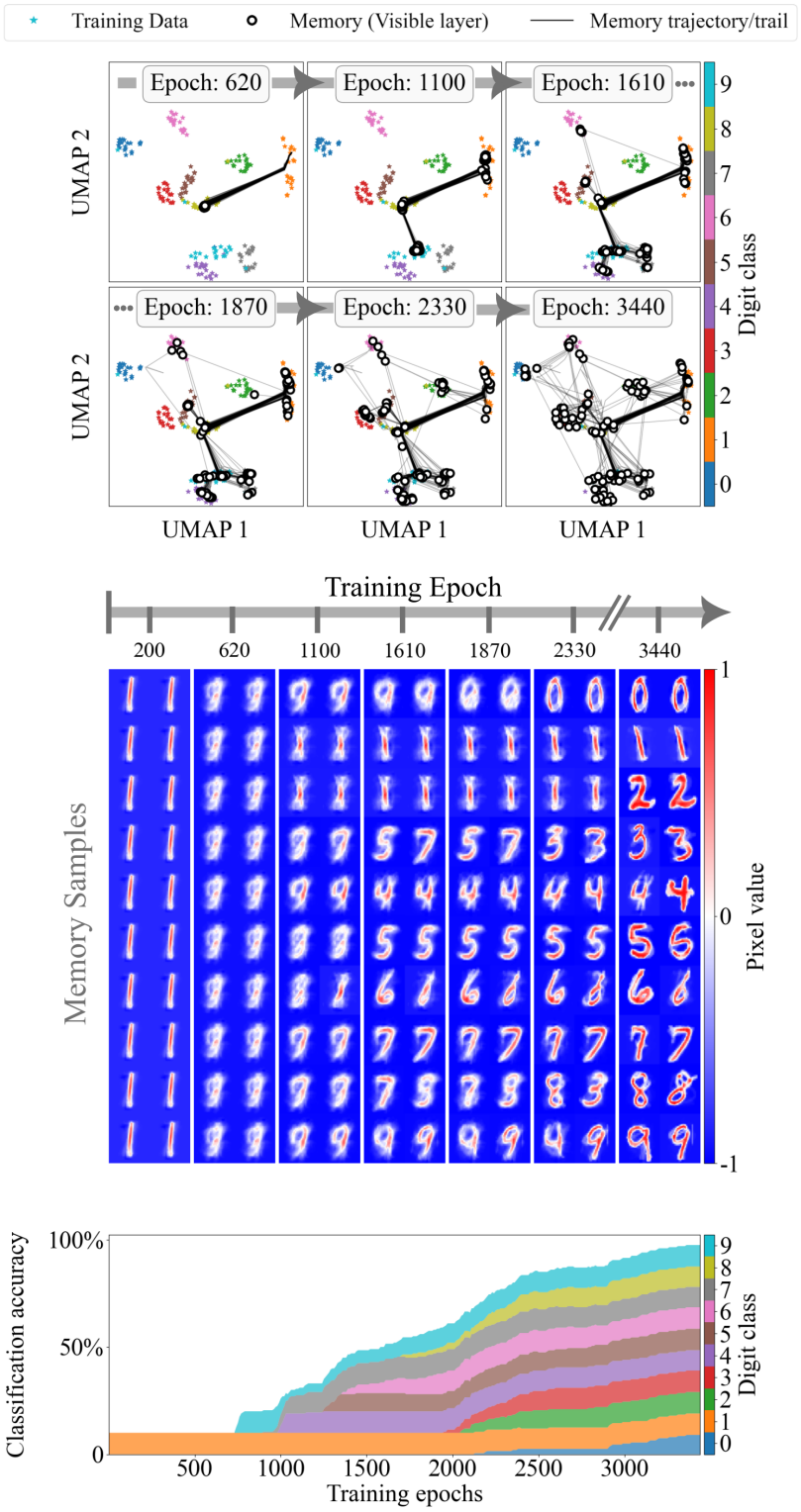

Figure S.11: A 100-memory system training on 200 digits (20 samples of each class) the same as in Fig. 3, the hyperparameters are $n = 40$, and $T_r = 0.85$, here however, with a momentum parameter set to 0.6.

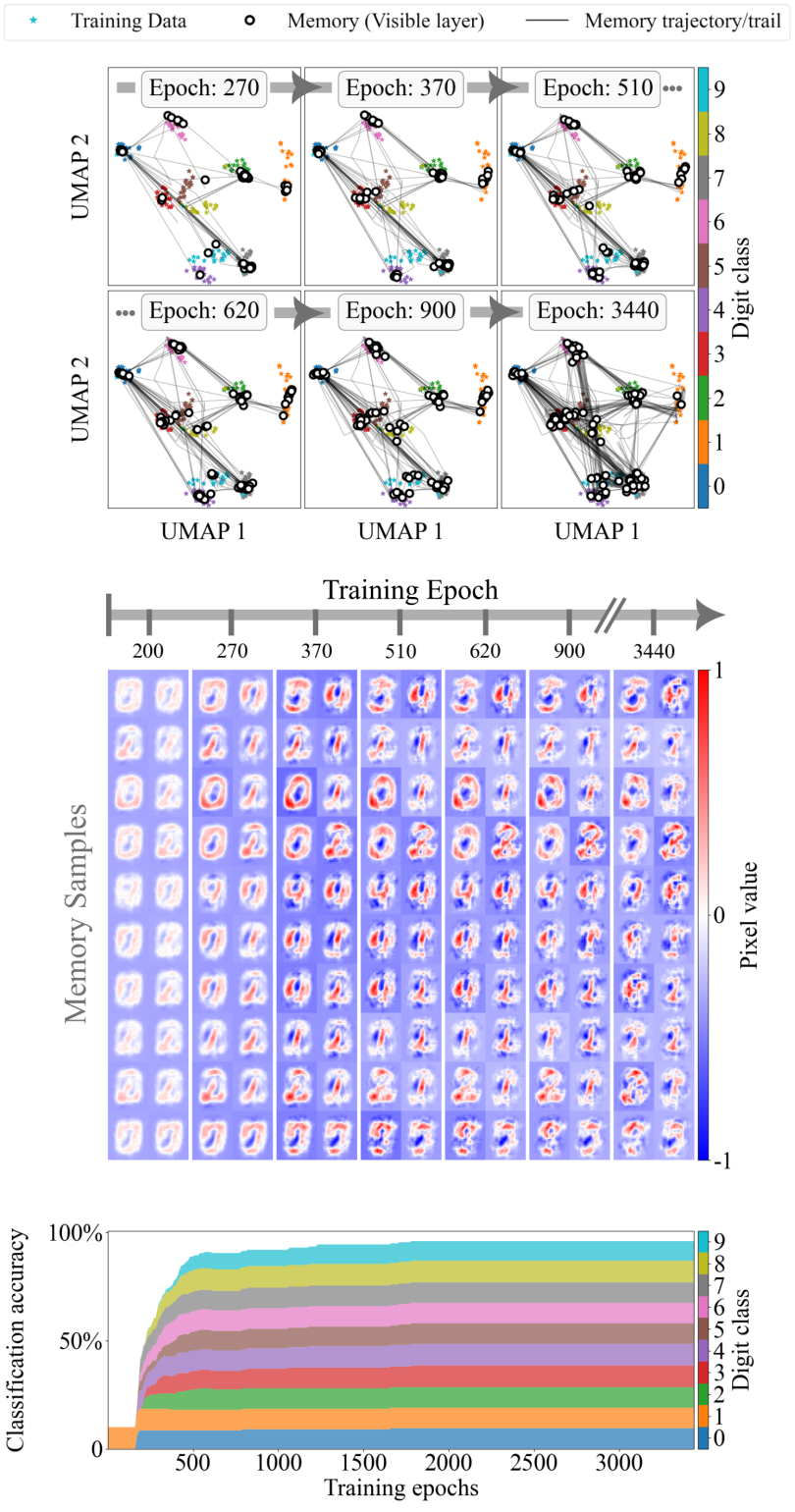

Figure S.12: A 100-memory system training on 200 digits (20 samples of each class) the same as in Fig. 3, the hyperparameters are $n = 3$, and $T_r = 0.85$, here however, with added noise at every epoch. The noise is Gaussian distributed with mean 0 and standard deviation 10^{-5}.

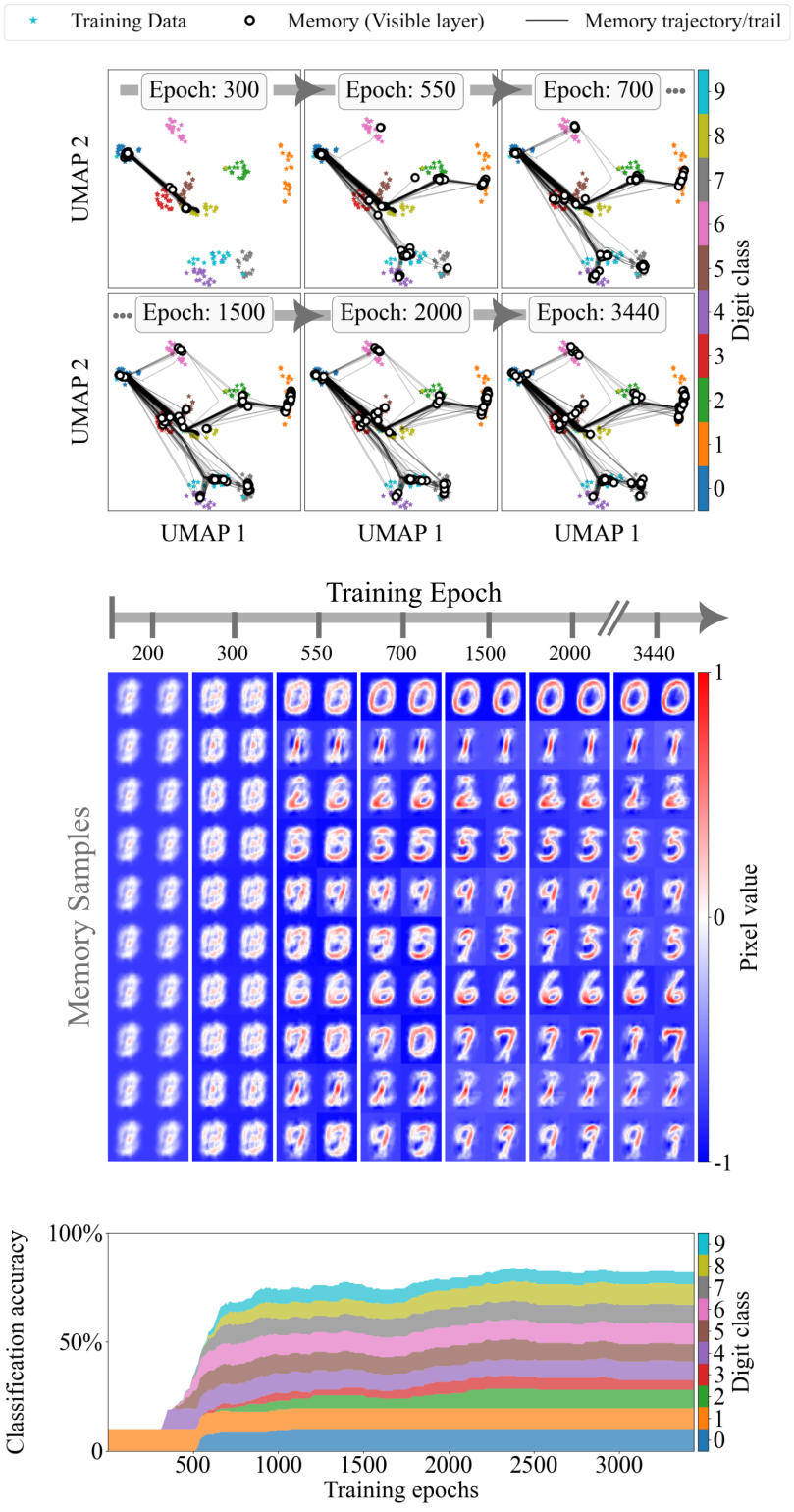

Figure S.13: A 100-memory system training on 200 digits (20 samples of each class) the same as in Fig. 3, the hyperparameters are $n = 15$, and $T_r = 0.85$, here however, with added noise at every epoch. The noise is Gaussian distributed with mean 0 and standard deviation 10^{-5}.

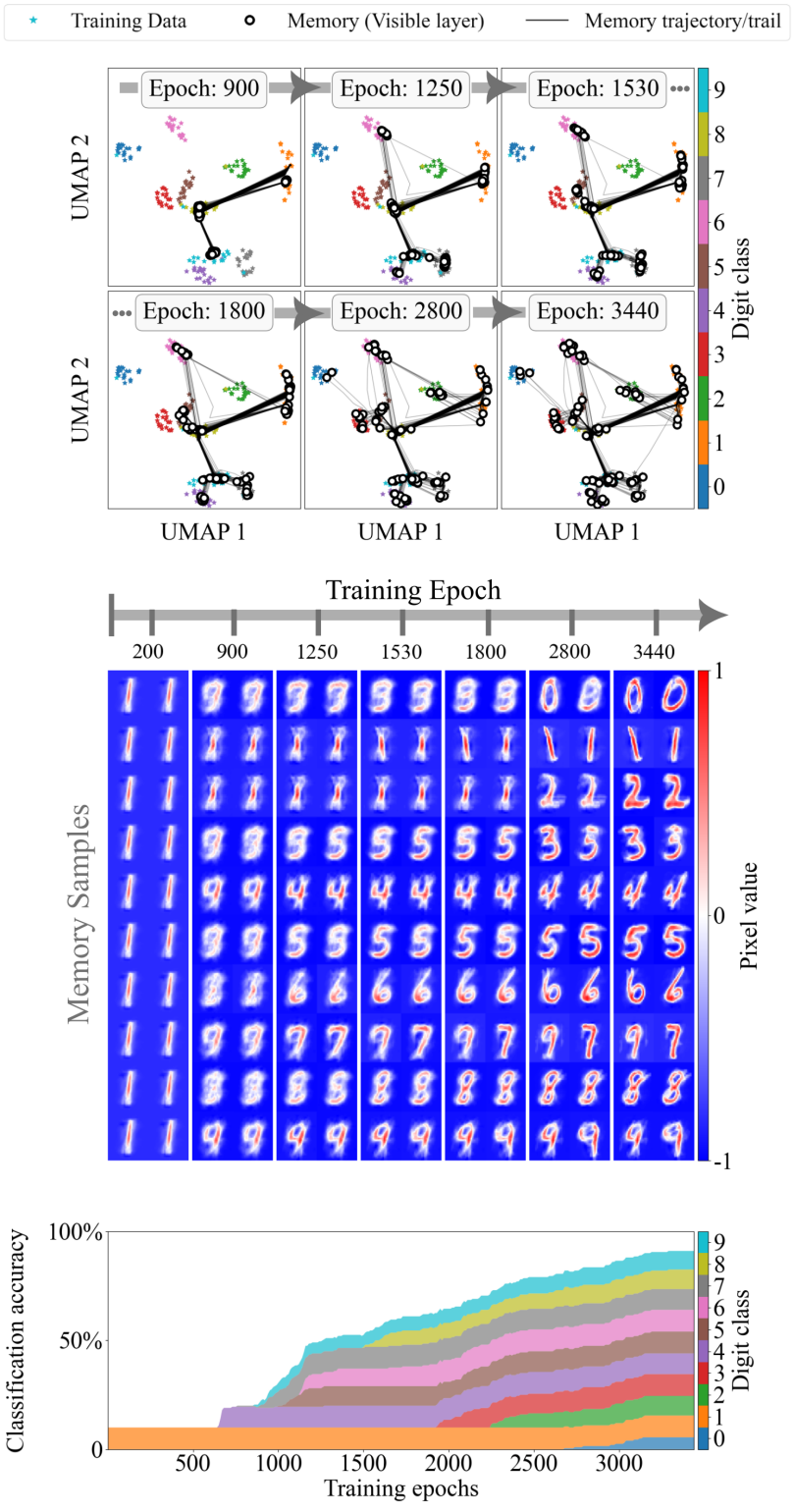

Figure S.14: A 100-memory system training on 200 digits (20 samples of each class) the same as in Fig. 3, the hyperparameters are $n = 3$, and $T_r = 0.85$, here however, with added noise at every epoch. The noise is Gaussian distributed with mean 0 and standard deviation 10^{-5}.

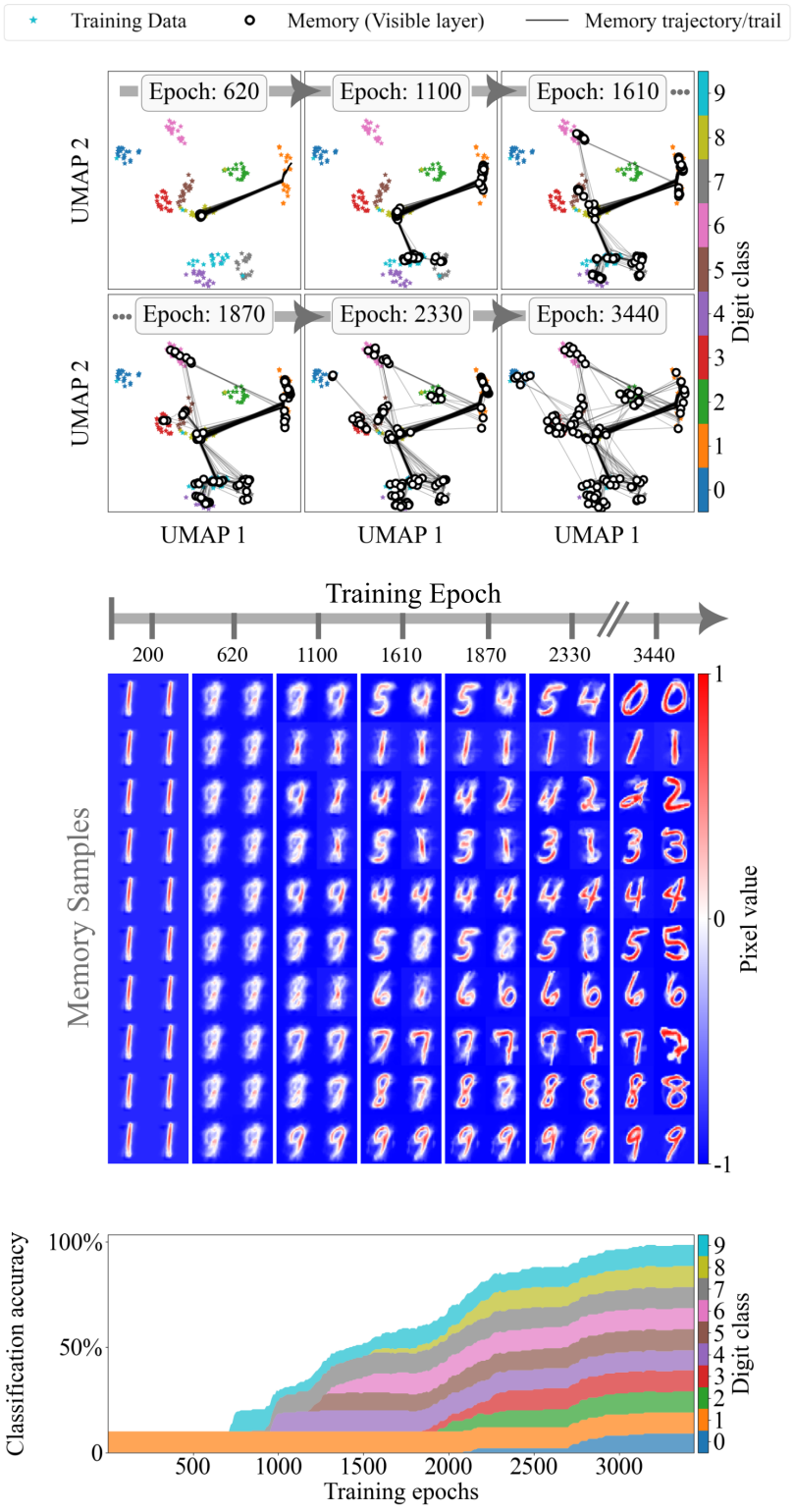

Figure S.15: A 100-memory system training on 200 digits (20 samples of each class) the same as in Fig. 3, the hyperparameters are $n = 40$, and $T_r = 0.85$, here however, with added noise at every epoch. The noise is Gaussian distributed with mean 0 and standard deviation 10^{-5}.

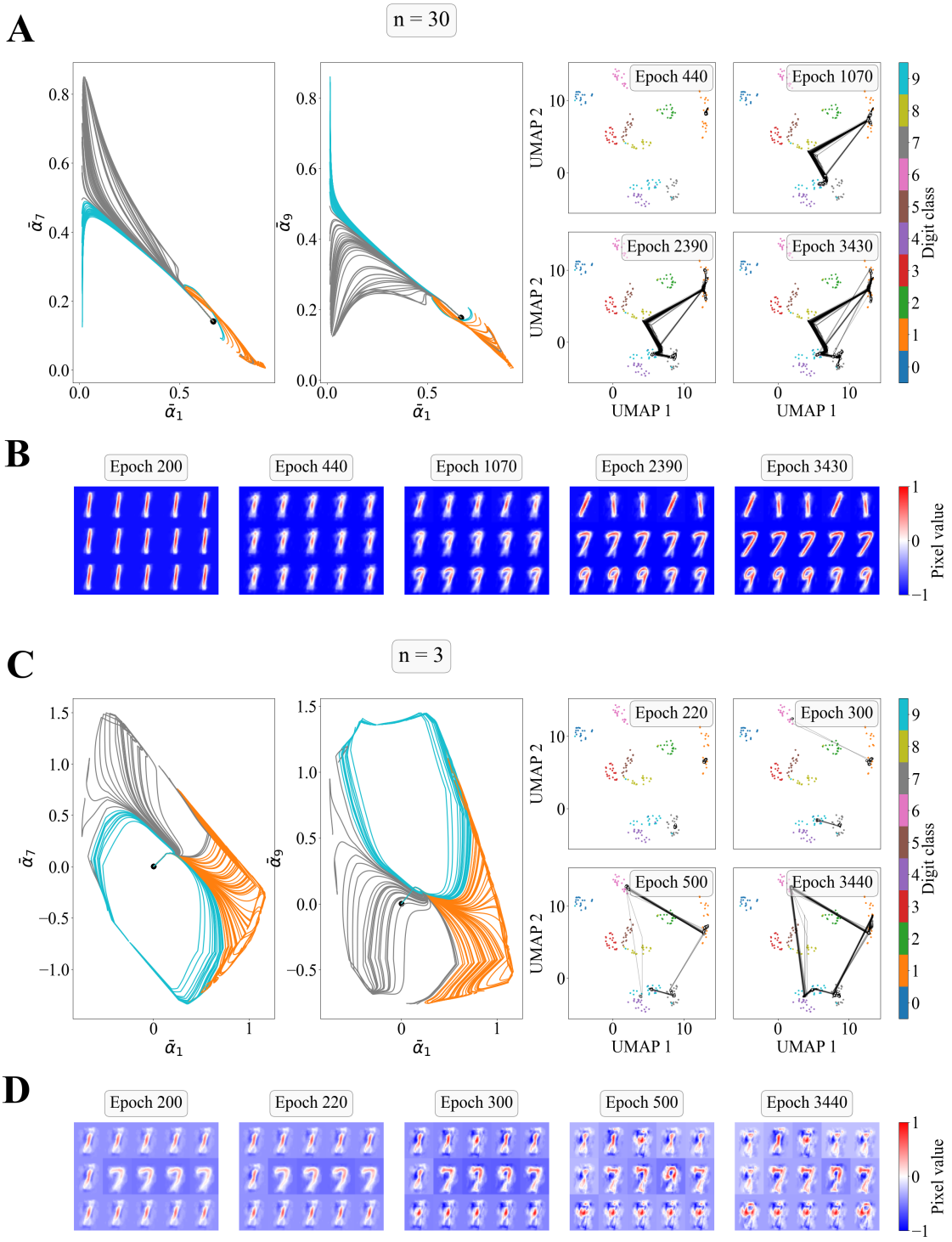

Figure S.16: A 100-memory system training on a set composed of 60 digits (20 of each class), the classes included in the training set are 1, 7 and 9. This system recapitulates the dynamics shown in Fig. 4.

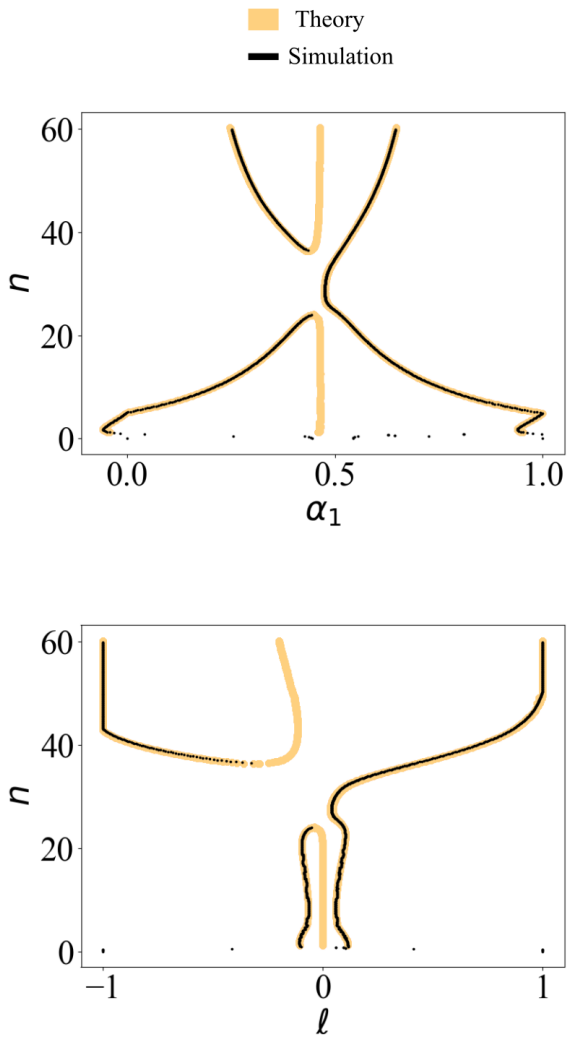

Figure S.17: Using the same training set as in Fig. 5, a 1-memory system with two training samples. We compare the analytical bifurcation diagram to the one obtain through simulation.

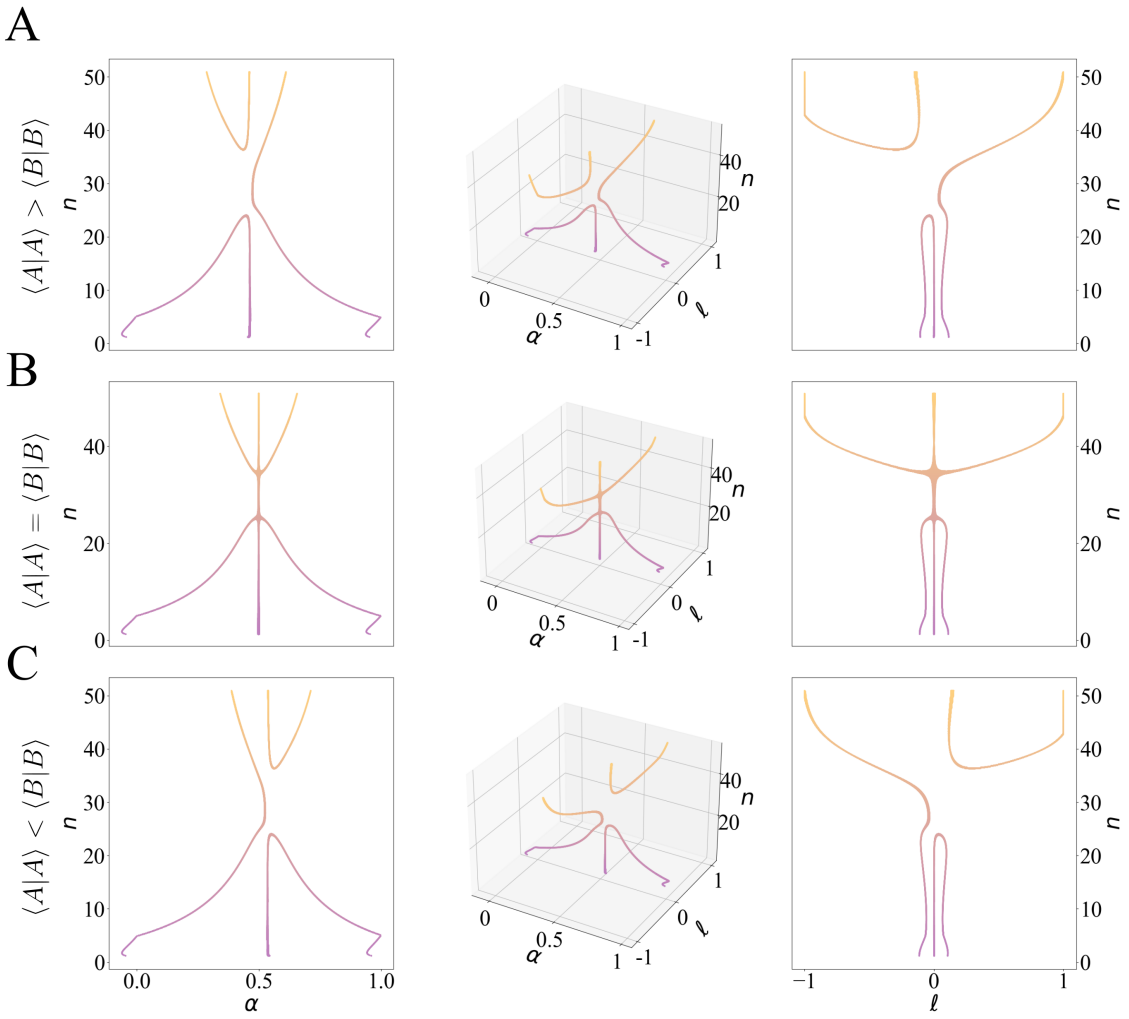

Figure S.18: The bifurcation diagrams of 3 1-memory systems, each with a different $\langle A|A\rangle$, $\langle B|B\rangle$ relationship, while $\langle A|B\rangle$ is fixed. This shows the symmetry required to obtain a pitchfork bifurcation is $\langle A|A\rangle = \langle B|B\rangle$. This is complementary to Fig. 5.

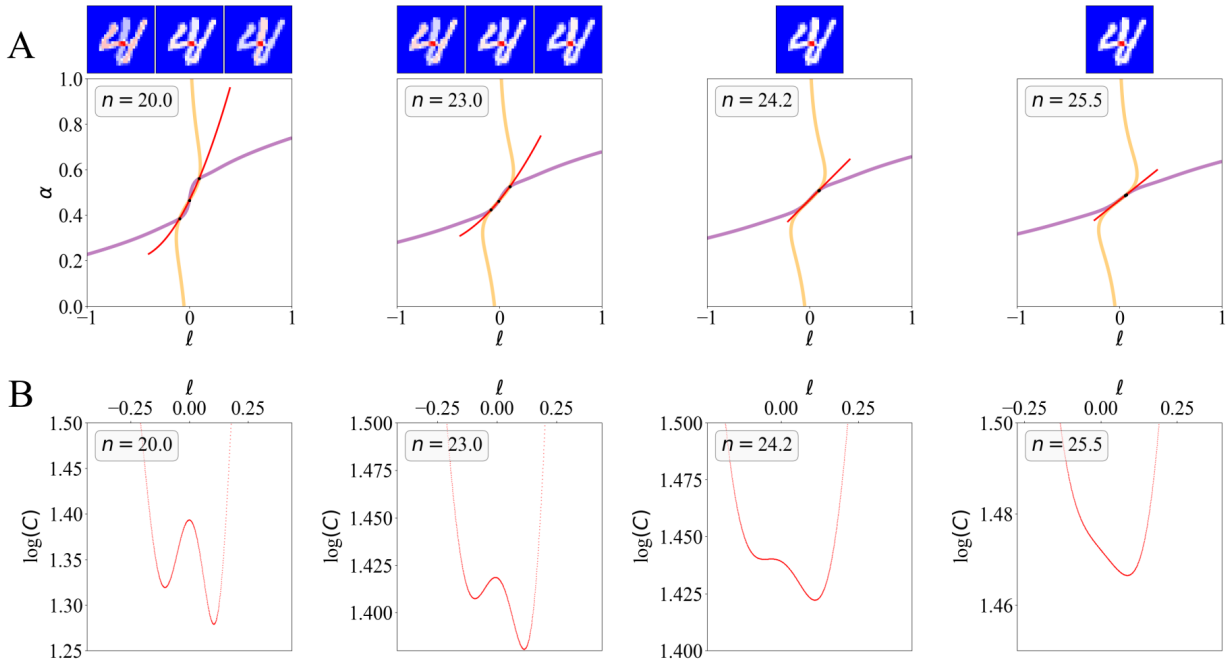

Figure S.19: Using the same training set as in Fig. 5, a 1-memory system with two training samples. In A, we show the nullclines for 4 different n values, near the saddle node bifurcation. We also define a red curve which goes through all fixed points, and varies as little as possible (qualitatively) from one n to the other. In B, we plot the cost function along that red curve to show the saddle-node bifurcation.

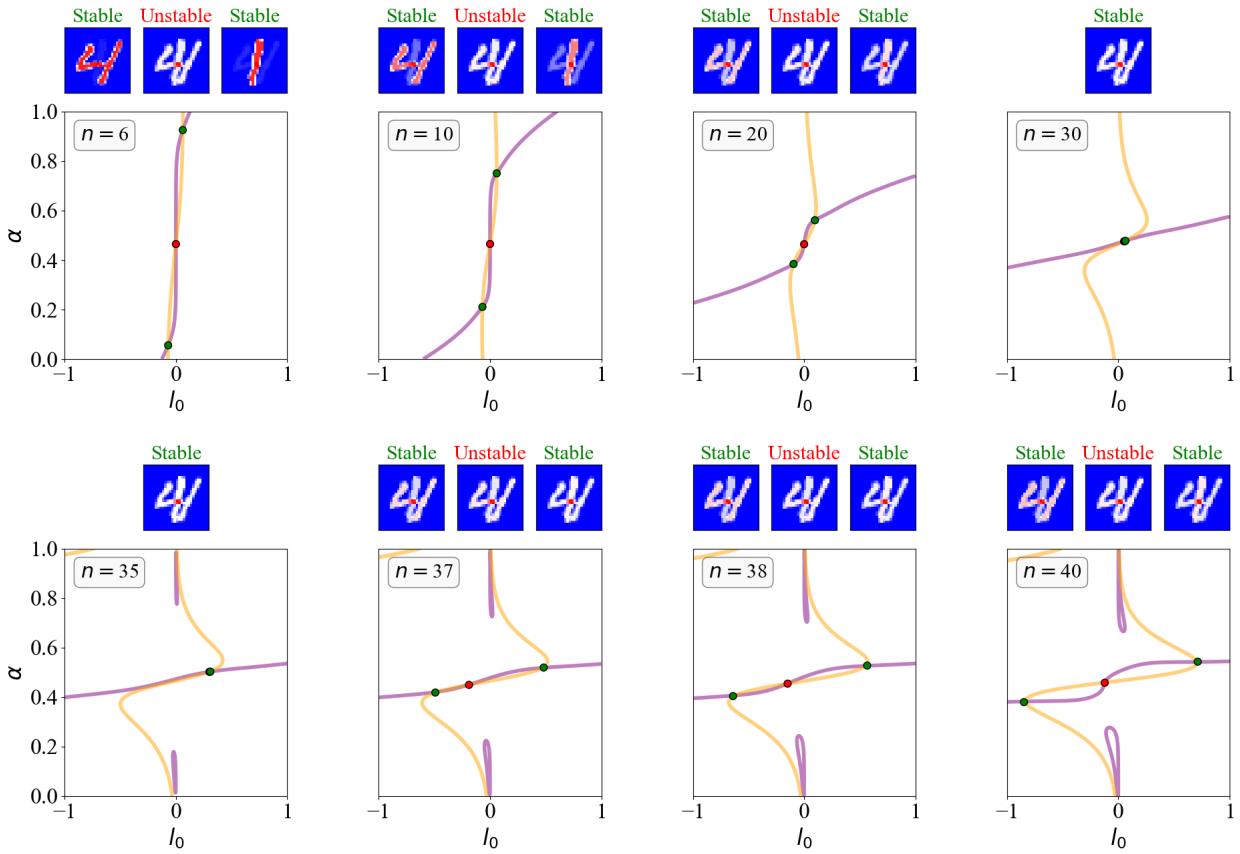

Figure S.20: The same system as in Fig. 5, a 1-memory system with two training samples. Here we show nullclines for more values of n.

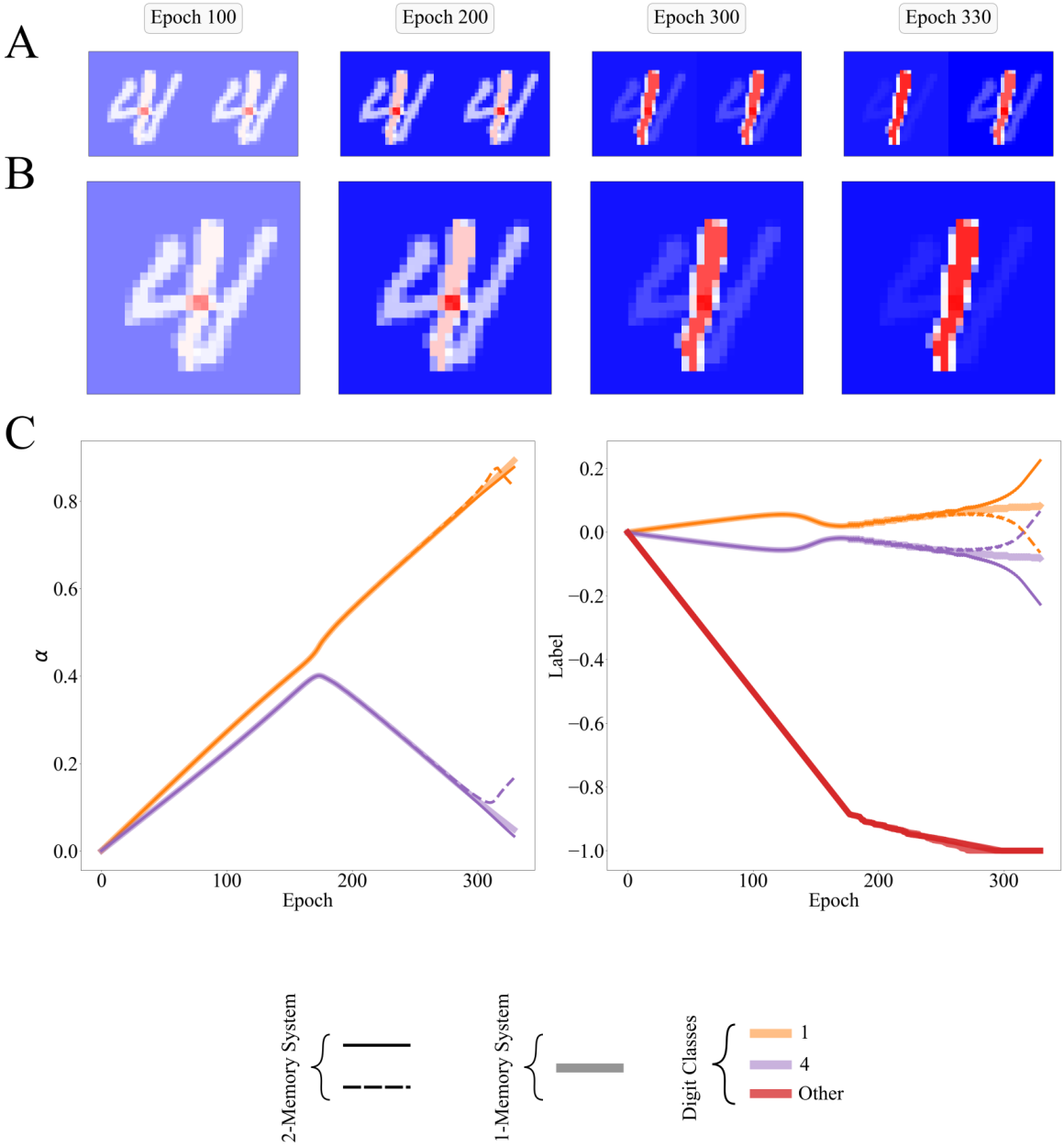

Figure S.21: The same training set as in Fig. 5, two training samples, a 1 and a 4. In A, a two memory system up to the split. In B, an equivalent 1-memory system. Both A, B indeed vary identically until the split. In C (left), the α's of both systems. In C (right), the labels of both systems. Here $n = 3$, $T_r = 0.89$.

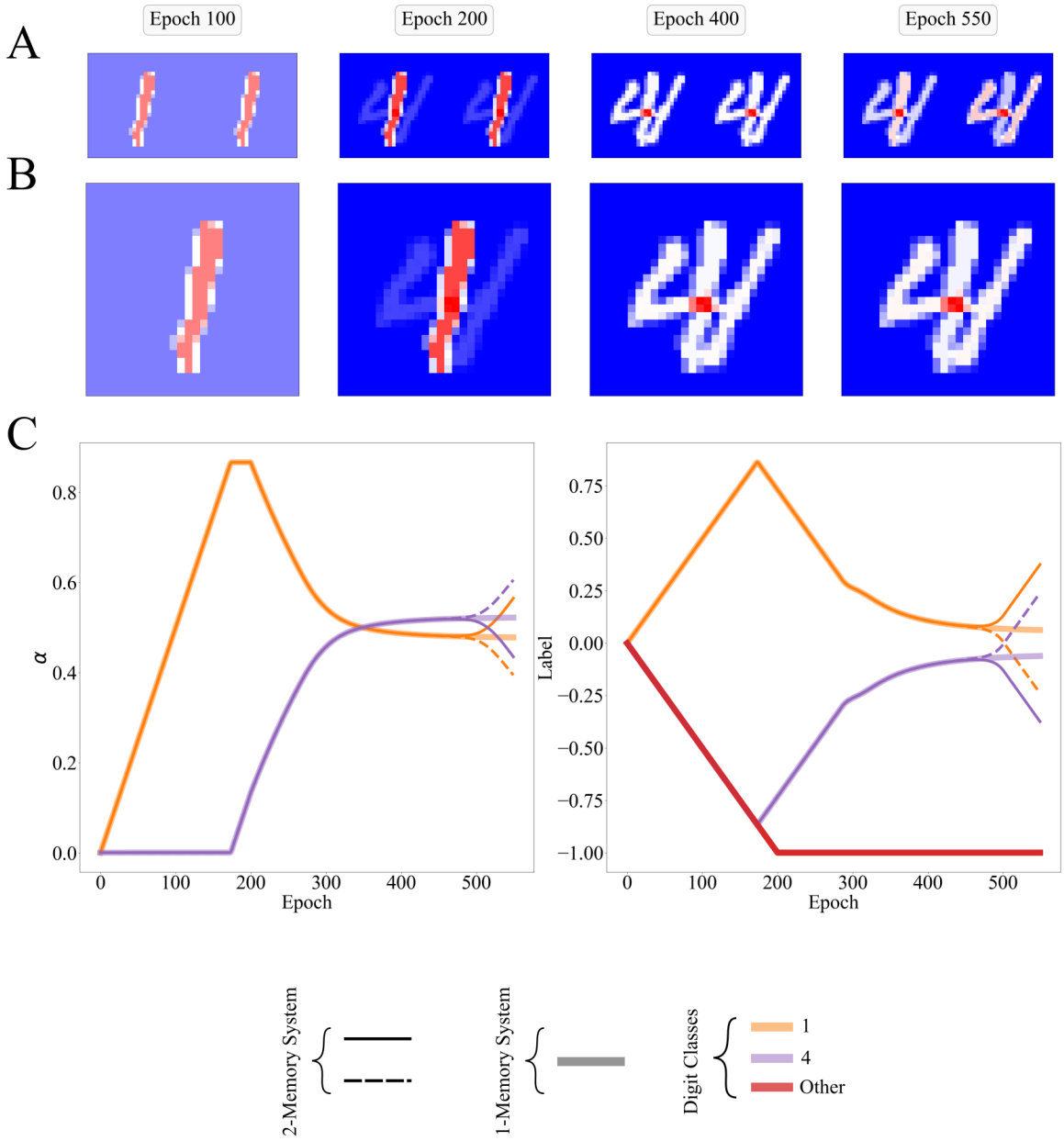

Figure S.22: The same training set as in Fig. 5, two training samples, a 1 and a 4. In A, a two memory system up to the split. In B, an equivalent 1-memory system. Both A, B indeed vary identically until the split. In C (left), the α's of both systems. In C (right), the labels of both systems. Here $n = 30$, $T_r = 0.89$.

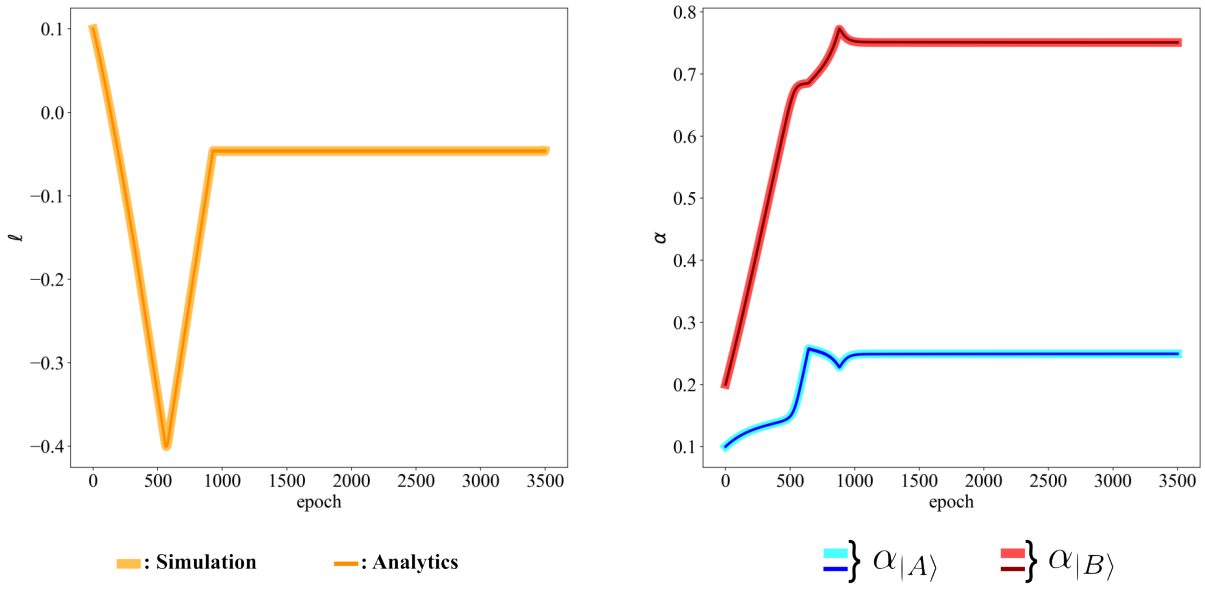

Figure S.23: The same training set as in Fig. 5, two training samples, a 1 and a 4. Comparing the simulations of the complete system to the numerical simulations of the equations derived in 3.

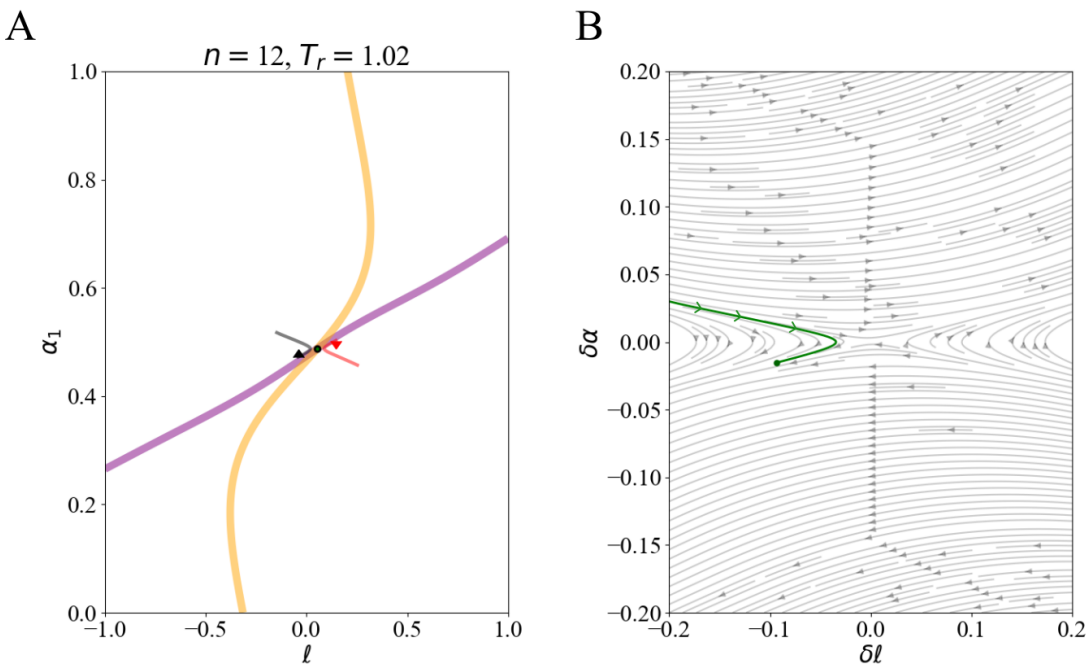

Figure S.24: A 2-memory system with the same training set (2 digits) as in Fig. 5. In A, the dynamics are plotted in the α, ℓ space for each memory (one in red, the other in black). In B, the difference between memories along α ($\delta\alpha$) and ℓ ($\delta\ell$). The flow lines are defined from the first degree expansion of the system as described in section 4. The hyperparameters are $n = 12$, and $T = 0.89$. The system is initialized centered at the fixed point shown in A, with $\delta\alpha = 0.31$ and $\delta\ell = -0.2$.

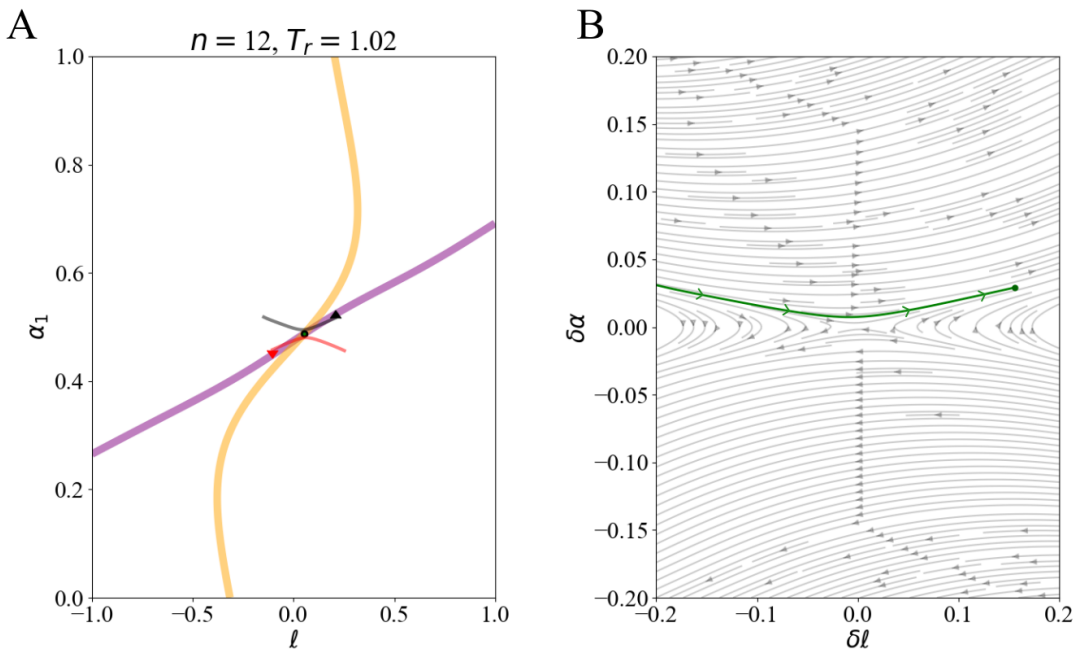

Figure S.25: A 2-memory system with the same training set (2 digits) as in Fig. 5. In A, the dynamics are plotted in the α, ℓ space for each memory (one in red, the other in black). In B, the difference between memories along α ($\delta\alpha$) and ℓ ($\delta\ell$). The flow lines are defined from the first degree expansion of the system as described in section 4. The hyperparameters are $n = 12$, and $T = 0.89$. The system is initialized centered at the fixed point shown in A, with $\delta\alpha = 0.30$ and $\delta\ell = -0.2$.

9 Supplementary Movies

Movie 1

Based on Figure 3, this movie illustrates the learning dynamics for a 100 memory system, trained on 20 digits of each class (200 in total). The rescaled temperature is 0.85, and $n = 30$. On the left we show the labels, each row is a separate class d. In the middle, the memories are plotted in a 10x10 grid. The standard colormap is used (blue: -1, white: 0, red: 1). On the right, the UMAP projection of the memories.

Movie 2

Based on Figure 3, this movie illustrates the learning dynamics for a 100 memory system, trained on 20 digits of each class (200 in total). The rescaled temperature is 0.85, and $n = 3$. On the left we show the labels, each row is a separate class d. In the middle, the memories are plotted in a 10x10 grid. The standard colormap is used (blue: -1, white: 0, red: 1). On the right, the UMAP projection of the memories.

Movie 3

Based on Figure 5, this movie illustrates the dynamics of a 2-memory system for many n. As in Figure 5 the rescaled temperature is 0.89 and the training set consists of a 1 and a 4. Each subplot illustrates the dynamics for different n-value is shown (6, 10, 20, 30, 35, 37, 38, 40). For each subplot, six 2-memory systems are shown; each black/red (upper/lower) triangle marker is an independent simulation. The streamlines show the dynamics in the region when the normalization condition is held. The yellow line represents the label nullcline. The purple line represents the memory nullcline. Green points are stable fixed points of an equivalent 1-memory system; saddles of the 2-memory system. Red points are unstable fixed points of an equivalent 1-memory system. The memories tend to converge towards the nearest nullcline, follow them towards a fixed point and split. Notice that post-split, the streamlines are no longer valid as additional dimension are necessary to describe the system.

Movie 4

Based on Figure 8, this movie illustrates the final states of a 100-memory system as a function of temperature. The hyperparameter n is fixed to 25, and the rescaled temperature is varied from 0.83 to 1.15. The system is trained on a training set containing all digit classes, with 20 training samples per class. We see that as temperature increases, the system gets stuck at saddles, starting at the most downstream saddle and going up until the system only splits once and contains effectively two types of memory.

Movie 5

Similar to Movie 4, here the hyperparameter n is fixed to 15, and the rescaled temperature is varied from 0.83 to 1.2.

Movie 6

Same temperature range and training set as Movie 5, only change is the hyperparameter n is fixed to 40.