Recurrent Neural Networks: from prediction to representation, a dynamical systems perspective

Guillaume Pourcel[1]

[1]*Bernoulli Institute & CogniGron, University of Groningen, Netherlands*

Summary. A major advance in Cognitive Science was moving from the study of behavior ("behaviorism") to the study of concepts in-between perception and action ("cognitivism") in terms of information processing concepts. Machine Learning also followed a similar trajectory by moving from the study of input/output prediction to the learning of representation [1]. However, these studies are centrally focused on Feed Forward Neural Networks, which is inadequate for more general Recurrent Neural Networks and other Nonlinear Dynamical Systems where it's not clear how abstract representations should be encoded. I present here a framework inspired by conceptors [2] to conceptualize the task of learning representation in Dynamical Systems by carefully organizing neural trajectories instead of organizing atemporal neural activations (as done classically in Machine Learning). I discuss how this framework applies to Dynamical Systems in general, where conceptors are just one entry point to a more general task.

Classically, the challenge of Learning Representation (LR) consists in extracting abstract/concept-like encodings of a perceptual input without having access to supervision. The adequacy of these encodings can be judged by their capacity to be decoded for regenerating the input. Additionally, these encodings need to generalize. It should be possible to interpolate to generate novel meaningful examples.

The major difficulty for LR in Nonlinear Dynamical Systems (NDS) is that they don't seem to offer an identifiable, controllable neural code. The solution proposed by conceptors is to use the linear subspaces of different input-driven neural trajectories as coarse-grained codes. Crucially, this code can be used bi-directionally: it can be extracted from input and decoded to regenerate the input.

To formalize the task, we start with a parametric family of time series $u_{\lambda^{inp}}$:

$$u_{\lambda^{inp}} = \mathrm{g}\left(\lambda^{inp}, v, w\right) \tag{1}$$

where λ^{inp} is the input parameter, an abstract, unobserved concept that has to be inferred, and g is a function that mixes the two signals v and w with λ^{inp}. Cognitively, λ^{inp} could represent an abstract speed variable allowing an agent to perceive and produce complex speech signals along a continuum of nonlinear distortions. For the cognitive system, I will use a discrete-time Recurrent Neural Network (RNN):

$$x(n+1) = tanh\left(Wx(n) + W^{in}.u_{\lambda^{inp}}(n)\right) \tag{2}$$

where $x(n)$ represents the N-dimensional neural activity at time $n \in \mathbb{N}$. W is the $N \times N$ connectivity matrix and W^{in} is the $N \times p$ input matrix expanding the input into the reservoir. For pedagogical purposes, we will use a simple linear morphing between two random signals of periodic 3, $u_{\lambda^{inp}}(n) = \left(1 - \lambda^{inp}\right)v(n) + \lambda^{inp}w(n)$ sent to a 3-dimensional RNN (Fig.1).

Conceptor methods allow the input-driven system (2) to be transformed into an autonomous system that regenerates the input. First, each input signal from a set $D = \left\{u_{\lambda_i^{inp}}\right\}_{i \in I}$ of signals can be translated into a code C^i. For a signal i, this code captures the linear subspace spanned by its associated neural trajectory (2). Mathematically, this code is a projection matrix on this subspace and can be easily computed [2]. For instance, Fig1.D displays conceptors in a specific configuration where they correspond to 2D planes[1] containing the neural trajectory. In higher dimensions, they can be visualized as ellipsoids (Fig1.A) [2].

The second step in the encoding phase (Fig.1A, left part) is to compute an additive weight matrix D (Fig.1A, green arrows) explicitly trained to reproduce the different trajectories created by the input [2]. While the D matrix entrenched[2] a set of input-driven dynamics into an autonomous system, each conceptor matrix C^i can be used to address and stabilize a specific one by projecting the dynamics with:

$$x(n+1) = \boldsymbol{C^i}\, tanh\left((W + \boldsymbol{D})x(n)\right) \tag{3}$$

[1] For geometrical interpretability, the RNN has been optimized to produce visually intuitive geometries, where the conceptors are projecting on various z-rotations of the the xy plane. In high dimension, this optimization is not necessary and interpolation with conceptors works with untrained RNNs (reservoir computers) [2].

[2] D creates a represention of the input within the autonomous RNN. To verify that the representation is appropriate, one can check if the input can be approximated by a time-invariant linear transformation on the neural trajectory (2).

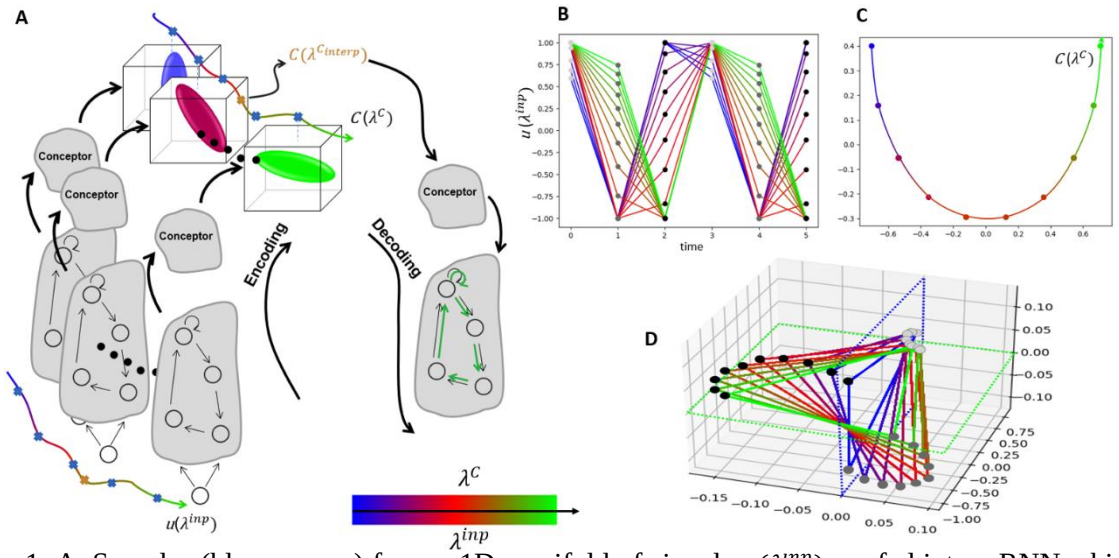

Figure 1. A. Samples (blue crosses) from a 1D manifold of signals $u(\lambda^{inp})$ are fed into a RNN which led to the computation of an additive weight matrix (green arrows) and a sets of conceptors (ellipsoids) on a manifold $C(\lambda^c)$. This manifold can be used to do interpolation (orange cross). B. Samples from the parametric family $u(\lambda^{inp})$. C. $C(\lambda^c)$ displayed in conceptor space (after PCA). D. Trajectories induced by the inputs in the space of neural activity (the initial transients are not displayed). The 2D planes correspond to conceptors associated with different trajectories.

We've seen that conceptor can be used to encode/decode signals, and we move now to the task of LR by looking at their capacity to generalize. More precisely, after learning (the computation of *D* and the $C^i s$), we want the following neural trajectory of an unseen data sample close to the example *i*:

$$x(n+1) = tanh\left(Wx(n) + W^{in}.u_{\lambda_i^{inp}+\Delta\lambda^{inp}}(n)\right) \quad (4)$$

to be approximated by:

$$x(n+1) = C_{interp}\ tanh\big((W+D)x(n)\big) \quad (5)$$

where C_{interp} is an appropriately interpolated code. Following ideas in Machine Learning [1], this code can be extracted from compression principles. Indeed, the variability of real-world data as well as our toy dataset *D* can be assumed to be encodable with limited variability between trajectories. Work in progress [3] shows that dimensionality reduction algorithms (for instance, self-organizing maps) on conceptor spaces are enough to uncover a parametrization of a manifold $C(\lambda^C)$ that correspond to input manifold $u(\lambda^{inp})$ (Fig1.A). In our case the conceptor manifold is one-dimensional (Fig1.C).

This proposal extends the use cases of conceptors from the manipulation of memories to the learning of representations. Additionally, this work points to general challenges of LR with NDS when we analyze two distinct roles for conceptors. First, they offer a space for compression to happen online without perturbing the unfolding of the dynamic. This space can be used to replace the input-induced perturbation by $\Delta\lambda^{inp}$ (4) by a conceptor perturbation (5). These phenomena can be analyzed with perturbation analysis [3]. The second role is to stabilize the interpolated trajectory. Indeed (5) pushes the NDS into a region where it was not trained to behave autonomously, and the induced trajectory might be unstable. This requirement is novel and should be further studied. Finally, this work opens new questions around this new task of LR with NDS. As opposed to FeedForward Neural Networks, which fix the direction of the flow of information (from encoding to decoding), RNNs and NDS don't have this restriction: encoding and decoding can happen simultaneously, opening the doors to new questions about cognitive systems coupled with their environment.

References

[1] Y. Bengio, et al., "Representation Learning: A Review and New Perspectives." arXiv, Apr. 23, 2014.
[2] H. Jaeger, "Controlling Recurrent Neural Networks by Conceptors," *arXiv:1403.3369 [cs]*,Apr.2017,
[3] G. Pourcel, "Learning Dynamical Representation with Conceptors" (manuscript in preparation)