The Kanerva Machine: a Generative Distributed Memory[*]

Wu et al.

2018

1 What

A memory augmented generative model, which applies VAEs[Kingma and Welling, 2013] and a couple of other important ideas to improve Kanerva's distributed memory model.

End-to-end differentiable memory which quickly adapts to new data and can generate similar samples. This memory is analytically tractable, which leads to optimal on-line compression via a Bayesian update rule. The model is easier to train than DNC, and it has a higher capacity (empirically shown on Omniglot and CIFAR).

2 Why

We have seen a lot of progress in augmenting neural nets with memory recently. However, we haven't yet solved the problem of efficiently using the memory. For instance, in DNCs [Graves et al., 2016] often collapse to reading/writing to single slots (*Have never heard of this before, where can I read about it?*) We would like to be able to learn more distributed strategies for using the memory block.

Hopfield Net was a pioneer in storing data patterns in low-energy states of a dynamic system. However, these models are constrained by the dimensionality of the input patterns. Boltzmann Machine lifts these constraints using latent variables but is very costly for reading/writing. Kanerva's model solves this efficiency issue by introducing an addressing mechanism which relates to a memory M. The size of this memory M does not depend on the input data dimensionality. However, there are lots of assumptions on data (the authors describe them in Appendix B), which limit the applicability on these models in real life. The authors build upon Kanerva's model applying VAE's, analytically deriving Bayesian update rule for writing the memory, and learning the addressing mechanism and test in on Omniglot/CIFAR datasets.

[*]Notes by Vitaly Kurin `https://yobibyte.github.io/`

3 How

In contrast to VAEs, we have two latent variables here: Z and Y. The former is the latent representation of data X. The latter is the addressing variables which help us to control the memory operations: $p_\theta(z_t|y_t, M)$.

The model objective:

$$\mathcal{J} = \int p(X, M) \ln p_\theta\left(X\,|M\right)\,\mathrm{d}M\mathrm{d}X = \int p(X)p(M|X)\sum_{t=1}^{T}\ln p_\theta\left(x_t\,|M\right)\,\mathrm{d}M\mathrm{d}X \tag{1}$$

The three main moments we are to understand is how the generative model works and how the inference model reads and updates the memory.

First, the joint distribution of the generative model:

$$p_\theta\left(X, Y, Z|M\right) = \prod_{t=1}^{T} p_\theta\left(x_t, y_t, z_t|M\right) = \prod_{t=1}^{T} p_\theta\left(x_t|z_t\right) p_\theta\left(z_t|y_t, M\right) p_\theta\left(y_t\right) \tag{2}$$

, where M is the memory random $K \times C$ matrix $p(M) = \mathcal{MN}(R, U, V)$, where R is a $K \times C$ matrix as the mean of M, U is a $K \times K$ matrix that provides the covariance between rows of M, and V is a $C \times C$ matrix providing covariances between columns of M.

Then we compute the weights w_t to control the memory access:

$$w_t = {b_t}^\intercal \cdot A = f(y_t)^\intercal \cdot A \tag{3}$$

where $f(\cdot)$ transforms y_t and w_t to (may be) non-Gaussian distributions.

z_t prior depends on the memory:

$$p_\theta(z_t|y_t, M) = \mathcal{N}\left(z_t\middle|\, {w_t}^\intercal \cdot M, \sigma^2\, I_C\right) \tag{4}$$

Second, what's the probability of the adressing and latent data representation given the data and the memory state:

$$q_\phi\left(Y, Z|X, M\right) = \prod_{t=1}^{T} q_\phi\left(y_t, z_t|x_t, M\right) = \prod_{t=1}^{T} q_\phi\left(z_t|x_t, y_t, M\right)\, q_\phi\left(y_t|x_t\right) \tag{5}$$

Lastly, how to update the memory?

$$\begin{aligned} q_\phi\left(M|X\right) &= \int p_\theta\left(M, Y, Z|X\right)\,\mathrm{d}Z\mathrm{d}Y \\ &= \int p_\theta(M|\{y_1, \ldots, y_T\}, \{z_1, \ldots, z_T\}) \prod_{t=1}^{T} q_\phi(z_t|x_t) q_\phi(y_t|x_t)\,\mathrm{d}z_t\mathrm{d}y_t \\ &\approx p_\theta\left(M|\{y_1, \ldots, y_T\}, \{z_1, \ldots, z_T\}\right)\Big|_{y_t \sim q_\phi(y_t|x_t), z_t \sim q_\phi(z_t|x_t)} \end{aligned} \tag{6}$$

Updating the memory means updating the matrices R and U:

$$\Delta \leftarrow Z - W\,R \tag{7}$$

$$\Sigma_c \leftarrow W\,U \qquad \Sigma_z \leftarrow W\,U\,W^\intercal + \Sigma_\xi \tag{8}$$

$$R \leftarrow R + {\Sigma_c}^\intercal\,\Sigma_z^{-1}\,\Delta \qquad U \leftarrow U - {\Sigma_c}^\intercal \Sigma_z^{-1} \Sigma_c \tag{9}$$

where Δ is the prediction error before updating the memory, Σ_c is a $T \times C$ matrix providing the cross-covariance between Z and M, Σ_ξ is a $T \times T$ diagonal matrix whose diagonal elements are the noise variance σ^2 and Σ_z is a $T \times T$ matrix that encodes the covariance for $z_1, \dots, z_T$.

All above makes sense because (i) we have the linear Gaussian model, (ii) Bayes' rule provides an optimal trade off between preserving the old information and writing the new one [MacKay, 2003].

4 Evaluation

The authors test their models comparing the performance with VAEs and DNCs on Omniglot and CIFAR (*which one?*) for one-shot generation and denosing and interpolation. They compare negative variational lower bound and demonstrate that introducing memory helps a generative model to reduce the variational lower bound. They also plot the histogram of weights controlling the memory slots and show that they are widely distributed (most likely to show that this is the opposite to DNCs' collapsing to using one slot).

Though we should always take eyeballing with a grain of salt, the pics generated with the Kanerva machine are much less blurry and makes the VAE curse less noticeable.

The main point of denoising/interpolation evaluation is that the proposed model is able to iteratively recover the original image (since we can iteratively sample the memory). As a nice bonus of having linear combinations of memory slots as data representation, the weights are supposed to be meaningful and more interpretable. They empirically show that interpolating between different weights enables to smoothly change the produced images.

5 Comments

- Where can I read about DNCs collapsing to using only one memory slot for reading/writing?

- I don't fully get the second transition in Formula (3). Is it because we build this graphical model saying that x_t depends only on z_t, not on memory/y_t directly. If this is true, then the phrase 'the joint distribution of the generative model can be factorised as' is misleading. It should be something like 'we can build the following graphical model where...'. I'm stuck here:

$\prod_{t=1}^{T} p_\theta(x_t, y_t, z_t|M) = \prod_{t=1}^{T} p_\theta(y_t) p_\theta(x_t, z_t|y_t, M) = \prod_{t=1}^{T} p_\theta(x_t|y_t, z_t, M) p(z_t|y_t, M) p_\theta(y_t)$, but in Formula (3), we see $p_\theta(x_t|z_t)$, not $p(x_t|y_t, z_t, M)$.

- If I get it right, the addressing variables y_t are also notated as A. It's confusing since the memory M is always uses as M, but y_t are always used instead A.
- The paper is extremely hard to grasp, but I suppose it's just me who doesn't have enough experience with VAEs and similar models/problems.
- Can we apply all of this magic to RL? I'm not sure I understand the concept of *exchangeable episode*. If it's about the conditional independence of x_t given the memory, this looks like a markovian assumption and we can proceed. If not (they say an exchangeable episode is a subset of an entire dataset whose order doesn't matter), this is bad for us. However, we can try to apply this to GAIL where we train the discriminator on state-action pairs, not on the trajectories.

References

[Graves et al., 2016] Graves, A., Wayne, G., Reynolds, M., Harley, T., Danihelka, I., Grabska-Barwińska, A., Colmenarejo, S. G., Grefenstette, E., Ramalho, T., Agapiou, J., et al. (2016). Hybrid computing using a neural network with dynamic external memory. *Nature*, 538(7626):471.

[Kingma and Welling, 2013] Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

[MacKay, 2003] MacKay, D. J. (2003). *Information theory, inference and learning algorithms*. Cambridge university press.