Modularity with noise-gating

Max Clarke

October 21, 2022

1 TL;DR

1. Regularise residual block (or other "module") outputs to be gaussian with a witness function / discriminator.
2. Mix noise with residual block outputs to limit the information rate, and prevent against low-value information leakage.
3. Don't screw up the residual stream with noise, by coupling the mixing parameter of adding the noise with the mixing parameter of adding to the residual stream.

2 Idea

I have an idea for a "noise gate" which can gate the activations of some neural network component, such that it can be learned when or when not to turn off, but also which guarantees that when it's turned off, it's really turned off (And we needn't worry the potential ability of the network to hide information in small fluctuations of the activations.)

3 Mix-Add

I've just tried out a "Mix-Add" operation $\text{mix}(a, b, m)$ that adds two vectors a and b which have some norm, let's say both have norm of 1, and scales them before adding them such that the resulting vector has norm of 1. The scaling factor m is a learnable parameter. In maths:

$$\begin{aligned}
\text{sigmoid}(m) = \sigma(m) &= \frac{1}{1+e^{-m}} \\
\text{mix}(a, b, m) &\stackrel{\text{def}}{=} a\sqrt{\sigma(m)} + b\sqrt{1-\sigma(m)} \\
\text{residual}(x, f) &= \text{mix}(x, f(x), m) \\
m &\in \mathbb{R}
\end{aligned}$$

where f is a residual block and m is a learned mixing parameter (which we can set high at the start of training to keep the gradient particularly stable). Also σ doesn't have to be the sigmoid function, it could be any function that maps $\mathbb{R}$ to $[0, 1]$.

4 Noise gate

I realized we might be able to use this to make a kind of principled modularity gate, an "information gate", that lets the network choose an information rate for a given residual block.

Let $n \sim (\mu = 0, \sigma = frac1\sqrt{D})$ be some noise vector sampled from a Gaussian independently at each training step, which has norm ≈ 1.

Let's also say we have some large batch of residual block outputs $f(\mathbf{x})$. We can regularize the distribution of $f(\mathbf{x})$ to be Gaussian with a certain variance using a witness function like MMD (or with a learned discriminator) as in [1]. Each $f(x_i)$ is constrained to be within some region of the space.

Then, if we add $x_{noisy} = \text{mix}(n, x, m)$, for some m, we restrict the maximum information rate that can pass out of the residual block (for gaussians the signal-to-noise ratio is simply the ratio of the variance of the signal to the variance of the noise).

Ok, but now the noise will mess up the residual stream? Well, we can add a second gate that gates the noise, so that when the noise is being added, the residual block is *not* being added.

So the whole thing is:

$$\begin{aligned} f &= \text{residual block} \\ n &= \text{noise} \\ m &= \text{mixing parameter} \\ \text{noise gate}(x, f) &= \text{mix}(x, \text{mix}(n, f(x), m), m) \end{aligned}$$

where when m is high, the noise is high but is not being added to the residual stream, and when m is low, the noise is low and is being added to the residual stream.

5 Regularizing the gate for modularity loss

If we add another loss term [1] then we can regularize a network to be more or less modular.

What this means is that we should incentivize the network to set the mixing parameter m to be high (high noise, low addition to residual stream) for as many residual blocks as possible.

[1] the one we have already is a discriminator/witness function which keeps the residual block outputs gaussian

If we fix m then we have an ordinary residual network.

If we learn m as a weight, then the network will eventually learn to use the fewest residual blocks possible for the task.

If the network can compute m as a function of the activations in previous layers, then the network will eventu learn when and when not to set m high. The network should learn blocks that are very useful for some inputs, but can be turned off for others.

References

[1] Daniel T. Braithwaite and W. Bastiaan Kleijn. "Bounded Information Rate Variational Autoencoders". In: *CoRR* abs/1807.07306 (2018). arXiv: 1807.07306. URL: http://arxiv.org/abs/1807.07306.