1 The Problem

We consider a feedforward neural network with a single hidden layer and activation function σ. It receives an input $x \in \mathbb{R}^n$ and produces a scalar output $\hat{y} \in \mathbb{R}$. The hidden layer has K units. The weights for the first and second layer are $W_1 \in \mathbb{R}^{K \times n}$ and $W_2 \in \mathbb{R}^{1 \times K}$, respectively, and the corresponding biases are $b_1 \in \mathbb{R}^K$ and $b_2 \in \mathbb{R}$.

$$\hat{y} = W_2 \sigma(W_1 x + b_1) + b_2. \tag{1}$$

Our data x are sampled from a mixture of two translation-invariant distributions in some family $\{p_\xi\}_\xi$ parameterized by a correlation length-scale ξ. That is, we sample $x \sim p_{\xi_1}$ with probability $\frac{1}{2}$ and $x \sim p_{\xi_2}$ otherwise. If x is sampled from p_{ξ_1}, then $y(x) = 1$; otherwise, $y(x) = 0$. We can train using either mean-squared error or cross-entropy loss, though we primarily consider the former.

Alessandro's paper primarily considers the case where $W_2 = \frac{1}{K}\mathbf{1}^\top$ (take the mean of the hidden activations) is fixed and $\sigma(h) = \mathrm{erf}(\frac{h}{\sqrt{2}})$. I have also tried $\sigma = \text{sigmoid}, \text{ReLU}$. For the former, the results are qualitatively identical, while for the latter we get localization if $\xi_1 > \xi_2$ and short-range oscillations otherwise. For $\sigma = \text{ReLU}$, one can further remove the bias terms b_1 and b_2 (though not for sigmoid).

We consider two types of datasets: the nonlinear Gaussian process (NLGP) and the single pulse (SP). We explain them in more detail later. They differ primarily in that the former has continuous support on $\mathbb{R}^n$, while the latter has discrete support on a subset of $\{0, 1\}^n$. The former also has a gain parameter that controls the degree of localization, while the latter does not.

Motivated by localization with the ReLU model, we tried to see how well a gated deep linear network (GDLN) could do. Next, we'll provide some experimental support for why this might work, and then do some analysis. We'll conclude with some questions, concerns, and ideas.

2 Have You Tried Making It Linear?

Gating lets us decompose the ReLU post-activation in terms of the pre-activation's sign and magnitude.

$$\text{ReLU}(\langle w_1, x\rangle) = g(x)\langle w_1, x\rangle \qquad \text{where} \qquad g(x) = \mathbb{1}(\langle w_1, x\rangle \geq 0). \tag{2}$$

We generally assume that g does not vary during learning. Later on, we'll try to analyze what happens when this does not hold.

To assess the validity of this assumption, we need to see how much $g(x)$, as defined above, changes during learning. Additionally, post-hoc, we can usually pick a somewhat sensible gating structure that mimics a specific run's behavior. But we'd like to be able to determine this gating upfront. We explore all this in the following subsections.

2.1 Sign Flipping

Note that g is invariant to the scale of w_1. We've observed in previous experiments that w_1 appears to grow uniformly in size during much of its training. (There is, importantly, a phase where it goes from Gaussian to non-Gaussian, but the localization seems to be more likely to occur around its mode.) This suggests that

$g(x)$ may be relatively constant during learning. If this is so, then it would be reasonable to try using a standard GDLN to model the ReLU network.

2.2 Predicting Loca(liza)tion

2.3 Evolving Gates?

3 Let's Consider a Single Layer with Linear Activation...

3.1 Model

Our GDLN model is defined as follows:

$$\hat{y}(x) = \frac{1}{K}\left(\sum_{k\in[K]} g_k(x) w_k^\top\right) x, \tag{3}$$

where g_k are (node) gates, and $w_k \in \mathbb{R}^n$ are the rows of the first-layer weight matrix $W_1 \in \mathbb{R}^{K\times n}$. That is,

$$W_1 = \begin{pmatrix} w_1^\top \\ \vdots \\ w_K^\top \end{pmatrix} \tag{4}$$

3.2 Gradient Flow

Recalling the GDLN paper, the gradient flow for w_1 is given by

$$\tau \frac{d}{dt} w_1^\top = \frac{1}{K}\left[\Sigma^{yx}(p_1) - \sum_{k\in[K]} w_k^\top \Sigma^{xx}(p_1, p_k)\right], \tag{5}$$

where

$$\Sigma^{yx}(p_i) = \left\langle g_i y x^\top \right\rangle_{g,x,y} \tag{6}$$

$$\Sigma^{xx}(p_i, p_j) = \left\langle g_i g_j x x^\top \right\rangle_{g,x,y}. \tag{7}$$

Because we typically take g to be a deterministic function of x, I will often suppress the dependence on g in the expectation.

3.3 General Case

Let us relabel $b_i = \Sigma^{yx}(p_i)^\top$ and $A_{ij} = \Sigma^{xx}(p_i, p_j)$. Note that A_{ij} is symmetric and $A_{ij} = A_{ji}$. Then, we can write the gradient flow for all weights as

$$K\tau \frac{d}{dt} \underbrace{\begin{bmatrix} w_1 \\ \vdots \\ w_K \end{bmatrix}}_{w\in\mathbb{R}^{Kn}} = \underbrace{\begin{bmatrix} b_1 \\ \vdots \\ b_k \end{bmatrix}}_{b\in\mathbb{R}^{Kn}} - \underbrace{\begin{bmatrix} A_{11} & \cdots & A_{1K} \\ \vdots & \ddots & \vdots \\ A_{K1} & \cdots & A_{KK} \end{bmatrix}}_{A\in\mathbb{R}^{Kn\times Kn}} \begin{bmatrix} w_1 \\ \vdots \\ w_K \end{bmatrix}. \tag{8}$$

Observe that w is the vectorized form of our $K \times n$ first-layer weight matrix. Note also that A is a symmetric real matrix, so we can diagonalize it as $A = P\Lambda P^\top$, where the columns of P are the eigenvectors of A and the diagonal entries of Λ are the corresponding (nonnegative) eigenvalues. (It is symmetric because A is block symmetric with blocks A_{ij}, and the blocks are also symmetric.) We can reparameterize in terms of $u = P^\top w$ and $c = P^\top b$.

$$K\tau \frac{d}{dt} u = -\Lambda u + c \implies u(t) = \Lambda^{-1} e^{-\frac{t}{K\tau}\Lambda + C}\mathbf{1} + \Lambda^{-1} c, \tag{9}$$

where C is a constant diagonal matrix that defines the initial condition. So,

$$w(t) = P\Lambda^{-1} e^{-\frac{t}{K\tau}\Lambda + C}\mathbf{1} + P\Lambda^{-1} c \tag{10}$$

$$= A^{-1} P e^{-\frac{t}{K\tau}\Lambda + C}\mathbf{1} + A^{-1} b. \tag{11}$$

If none of the eigenvalues are zero, then $w(\infty) = A^{-1}b$.

If we write

$$\tilde{x} = \begin{bmatrix} g_1(x)x \\ \vdots \\ g_K(x)x \end{bmatrix} \in \mathbb{R}^{Kn}, \tag{12}$$

then $A = \langle \tilde{x}\tilde{x}^\top \rangle_{x,y}$ and $b = \langle \tilde{x} y \rangle_{x,y}$. Then, it is clear that this is the population solution to the OLS problem of regressing y on $\tilde{x}$.

In this context, one might ask, which gating structure minimizes the MSE loss? The loss is

$$\mathcal{L}_{OLS} = \left\langle \left(\tilde{x}'^\top (\langle \tilde{x}\tilde{x}^\top \rangle_{x,y})^{-1} \langle \tilde{x} y \rangle_{x,y} - y' \right)^2 \right\rangle_{x',y'} \tag{13}$$

$$= \left\langle (\tilde{x}'^\top (\langle \tilde{x}\tilde{x}^\top \rangle_{x,y})^{-1} \langle \tilde{x} y \rangle_{x,y})^2 - 2(y' \tilde{x}'^\top (\langle \tilde{x}\tilde{x}^\top \rangle_{x,y})^{-1} \langle \tilde{x} y \rangle_{x,y}) + (y')^2 \right\rangle_{x',y'} \tag{14}$$

$$= \frac{1}{2} - \langle y \tilde{x}^\top \rangle_{x,y} (\langle \tilde{x}\tilde{x}^\top \rangle_{x,y})^{-1} \langle \tilde{x} y \rangle_{x,y} \tag{15}$$

$$= \frac{1}{2} - \frac{1}{2} \langle \tilde{x} \rangle_{x|y=1}^\top (\langle \tilde{x}\tilde{x}^\top \rangle_{x|y=1} + \langle \tilde{x}\tilde{x}^\top \rangle_{x|y=0})^{-1} \langle \tilde{x} \rangle_{x|y=1}. \tag{16}$$

In the final step, we assumed (WLOG) that the negative class is $y = 0$ and the positive class is $y = 1$. (Throughout, we also assume that the classes are balanced.) The question is: For fixed p_{ξ_1} and p_{ξ_2}, how do we choose the gates g_k to minimize equation (16)? *I will have to think more about this.*

After a bit more thinking... I think that the best precision matrix would be maximally diagonal (no clue if this actually true! but maybe it holds empirically?). For Gaussian data (at least), this mean that the blocks are independent conditioned on all the other blocks. Gates that tile the space without overlap would achieve this (I think?). But tbh I haven't got the slightest clue!!

After an even littler bit of thinking... It may be useful to write this in terms of Kronecker products. Let

$$\tilde{g}(x) = \begin{bmatrix} g_1(x) \\ \vdots \\ g_K(x) \end{bmatrix} \in \{0,1\}^K. \tag{17}$$

Then, minimizing equation (16) is equivalent to maximizing

$$\mathcal{L}^*(\tilde{g}) = \langle \tilde{g} \otimes x \rangle_{x|y=1}^\top (\langle (\tilde{g} \otimes x)(\tilde{g} \otimes x)^\top \rangle_{x|y=1} + \langle (\tilde{g} \otimes x)(\tilde{g} \otimes x)^\top \rangle_{x|y=0})^{-1} \langle \tilde{g} \otimes x \rangle_{x|y=1} \tag{18}$$

over $\tilde{g} : \mathrm{supp}(p) \to \{0,1\}^K$.

3.4 Exclusive Gates

Let us assume that the gates are exclusive, that is, only one gate is active at a time. Then, $\Sigma^{xx}(p,q) = 0$ for $p \neq q$.

Then A becomes block diagonal. We can write the gradient flow for w_1 as

$$K\tau \frac{d}{dt} w_1 = -A_{11} w_1 + b_1. \tag{19}$$

Note that $A_{11} = \Sigma^{xx}(p_1, p_1)$ is always symmetric (and real). So, we can diagonalize it as $A_{11} = P\Lambda P^\top$, where the columns of P are $v_1, \dots, v_n$ and the diagonal entries of Λ are $\lambda_1, \dots, \lambda_n$. Let us introduce $u_1 = P^\top w_1$ and $c_1 = P^\top b_1$. Then,

$$K\tau \frac{d}{dt} u_1 = -\Lambda u_1 + c_1. \tag{20}$$

This ODE is solved by

$$u_1(t) = \Lambda^{-1} e^{-\frac{t}{K\tau}\Lambda + C} \mathbf{1} + \Lambda^{-1} c_1, \tag{21}$$

where C is a constant diagonal matrix that defines the initial condition. Then,

$$w_1(t) = P\left(\Lambda^{-1} e^{-\frac{t}{K\tau}\Lambda + C} \mathbf{1} + \Lambda^{-1} c\right) \tag{22}$$

$$= P\Lambda^{-1} e^{-\frac{t}{K\tau}\Lambda + C} \mathbf{1} + P\Lambda^{-1} c_1 \tag{23}$$

$$= A_{11}^{-1}\left(P e^{-\frac{t}{K\tau}\Lambda + C} \mathbf{1} + b_1\right). \tag{24}$$

Recalling A_{11} and b_1,

$$w_1(t) = (\Sigma^{xx}(p_1, p_1))^{-1}\left(P e^{-\frac{t}{\tau}\Lambda + C} \mathbf{1} + \Sigma^{yx}(p_1)^\top\right). \tag{25}$$

So,

$$w_1(\infty) = (\Sigma^{xx}(p_1, p_1))^{-1}\left(P e^{-\frac{t}{\tau}\Lambda + C} \mathbf{1} + \Sigma^{yx}(p_1)^\top\right). \tag{26}$$

As with above, this is the population solution to OLS, $(X^\top X)^{-1} X^\top y = (\langle x x^\top \rangle)^{-1}(\langle xy \rangle)$.

So, each weight matrix converges to the OLS solution on the subset of the data determined by its gate.

3.5 Redundant Gates

What if $g_1 = g_2$? Then, $b_1 = b_2$ and $A_{11} = A_{12} = A_{22}$. So,

$$K\tau \frac{d}{dt} w_1 = -A_{11}(w_1 + w_2) + b_1, \tag{27}$$

$$K\tau \frac{d}{dt} w_2 = -A_{11}(w_1 + w_2) + b_1. \tag{28}$$

Clearly, then, $w_1 - w_2$ is a constant vector. Moreover, $\frac{1}{2}(w_1 + w_2)$ evolves according to

$$\frac{K\tau}{2} \frac{d}{dt}(w_1 + w_2) = -A_{11}(w_1 + w_2) + b_1. \tag{29}$$

Writing $2\Delta = w_1 - w_2$ and $w_1 + w_2 = 2(w_1 - \Delta)$, we have

$$K\tau \frac{d}{dt} w_1 = K\tau \frac{d}{dt}(w_1 - \Delta) = -2A_{11}(w_1 - \Delta) + b_1. \tag{30}$$

We can plug this into our solution from the previous section to get

$$w_1(t) = \frac{1}{2} A_{11}^{-1} \left(P e^{-\frac{t}{K\tau}\Lambda + C} \mathbf{1} + b_1 \right). \tag{31}$$

So,

$$w_2(t) = \frac{1}{2} A_{11}^{-1} \left(P e^{-\frac{t}{K\tau}\Lambda + C} \mathbf{1} + b_1 \right) - (w_1(0) - w_2(0)). \tag{32}$$

4 Theory-driven Experiments

4.1 General Case

4.2 Single Gate

4.3 Redundant Gates

5 Next Steps

Concerns:

Questions:

Some ideas:

1. Let the gates vary during training. That is, g depends on t and x. So, Σ^{yx} and Σ^{xx} will depend on t as well. We can still solve the differential equation, and we may be able to use a single basis to diagonalize them across time (Toeplitz/circulant properties? DFT?).