Gaussian mixture layers for neural networks

Sinho Chewi Philippe Rigollet Yuling Yan
presenter: Shen Yuan

中國人民大學
RENMIN UNIVERSITY OF CHINA

高瓴人工智能学院
Gaoling School of Artificial Intelligence

Outline

Outline

Introduction

The main contribution is the proposal of an alternative to the fully connected layer, namely the Gaussian Mixture (GM) layer.

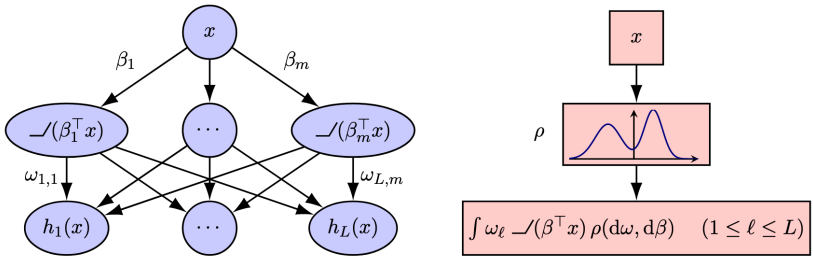

Figure 1: A GM layer (right) can act as a replacement for a fully connected layer (left).

Outline

Review of Mean-Field Theory

For the 2-layer fully connected neural network, a width-m neural network computes a function $h_{\boldsymbol{\omega},\boldsymbol{\beta}} : \mathbb{R}^d \to \mathbb{R}$ of the form

$$h_{\boldsymbol{\omega},\boldsymbol{\beta}}(x) = \frac{1}{m} \sum_{j=1}^{m} \omega_j \mathrm{ReLU}(\langle \beta_j, x \rangle) \tag{1}$$

where $(\omega_j, \beta_j) \in \mathbb{R} \times \mathbb{R}^d$ are trainable weights, for each neuron $j \in [m]$.
Let $\rho^{(m)}$ denote the empirical distribution of the weights, $\rho^{(m)} = m^{-1} \sum_{j=1}^{m} \delta_{(\omega_j, \beta_j)}$, then we can write $h_{\boldsymbol{\omega},\boldsymbol{\beta}}(x) = \int \omega \mathrm{ReLU}(\langle \beta, x \rangle) \rho^{(m)}(\mathrm{d}\omega, \mathrm{d}\beta)$.
We can view h as being parameterized by a probability measure ρ:

$$h_\rho(x) = \int \omega \mathrm{ReLU}(\langle \beta, x \rangle) \rho(\mathrm{d}\omega, \mathrm{d}\beta) \tag{2}$$

Review of Mean-Field Theory

Consider a loss objective $\mathcal{L}$ and minimize the objective $\mathcal{L}(h_{\boldsymbol{\omega},\boldsymbol{\beta}})$ by following the gradient flow for the weights $\{\omega_j, \beta_j\}_{j\in[m]}$.

This gives rise to a curve of parameters $(\boldsymbol{\omega}(t), \boldsymbol{\beta}(t))_{t\geq 0}$, and corresponding empirical measures $\rho^{(m)}(t) = m^{-1}\sum_{j=1}^{m} \delta_{(\omega_j(t),\beta_j(t))}$.

The insight of mean-field theory is that the evolution of the empirical measures can be described as the (time-rescaled) gradient flow of the loss function $\rho \mapsto \mathcal{L}(h_\rho)$ over the space of probability measures, equipped with the Wasserstein metric from optimal transport, initialized at $\rho^{(m)}(0)$.

Review of Mean-Field Theory

The advantage of this reformulation is that it admits a well-defined limit as $m \to \infty$: if $\rho^{(m)}(0) \to \rho(0)$, then the curve of measures $(\rho^{(m)}(t))_{t \geq 0}$ converges to the Wasserstein gradient flow of $\rho \mapsto \mathcal{L}(h_\rho)$, initialized at $\rho(0)$.

To summarize: the training dynamics of a finite-width neural network correspond to a Wasserstein gradient flow, initialized at (and remaining through its trajectory) an empirical measure, but the Wasserstein gradient flow picture is more general because it allows for flows of continuous measures.
Unfortunately, in the latter case, the Wasserstein gradient flow does not readily lend itself to tractable implementation.

Outline

The Gaussian Mixture (GM) Layer

Here, we restrict the measure ρ to be a Gaussian mixture with K components:

$$\rho = \rho_{\boldsymbol{\mu},\boldsymbol{\Sigma}} := \frac{1}{K}\sum_{k=1}^{K}\mathcal{N}(\mu_k, \sigma_k),\ \ \mu_k \in \mathbb{R}^{d+1},\ \ \Sigma_k \in \mathbb{R}^{(d+1)\times(d+1)}. \tag{3}$$

We use the short-hand notation $h_{\boldsymbol{\mu},\boldsymbol{\Sigma}} := h_{\rho_{\boldsymbol{\mu},\boldsymbol{\Sigma}}}$.
The use of mixture modelling to model the distribution over neurons is motivated by the empirical observation that for many problems, the neurons tend to cluster.

The Gaussian Mixture (GM) Layer

When we move to Gaussian mixtures, we effectively replace the particles $\theta_j = (\omega_j, \beta_j)$ with "Gaussian particles" $\mathcal{N}(\mu_k, \Sigma_k)$, which are themselves distributions over θ but can be viewed simply as (μ_k, Σ_k) pairs.

In the case $K = 1$, it turns out that the Wasserstein gradient flow restricted to Gaussian measures is implemented simply by evolving the parameters (μ, C) via the (Euclidean) gradient flow, where $\Sigma = CC^\top$.

The Gaussian Mixture (GM) Layer

When $K \geq 2$, it is no longer possible to follow the Wasserstein gradient flow restricted to Gaussian mixtures.

However, we can instead follow a Wasserstein gradient flow for the mixing measure $\nu := \frac{1}{K}\sum_{k=1}^{K} \delta_{(\mu_k, \Sigma_k)}$ over the Bures–Wasserstein space.

Theorem 1 (Informal) *Let $\mathfrak{L}$ be a loss function over the space of probability measures. Then, the Gaussian mixture gradient flow for $\mathfrak{L}$ is equivalent to the Euclidean gradient flow of the objective $\mathfrak{L}(h_{\boldsymbol{\mu},\boldsymbol{\Sigma}})$ with respect to the parameters $(\boldsymbol{\mu}, \boldsymbol{C})$, where $\Sigma_k = C_k C_k^\top$ for each $k \in [K]$.*

Outline

Parameterization

$$\rho = \rho_{\boldsymbol{\mu},\boldsymbol{\Sigma}} := \frac{1}{K}\sum_{k=1}^{K}\mathcal{N}(\mu_k, \sigma_k),\ \ \mu_k \in \mathbb{R}^{d+1},\ \ \Sigma_k \in \mathbb{R}^{(d+1)\times(d+1)}.$$

The most straightforward way to parameterize the function $h : \mathbb{R}^d \to \mathbb{R}^L$ is via Equation 2 and 3, where now $(\omega, \beta) \in \mathbb{R}^L \times \mathbb{R}^d$, i.e., the Gaussian mixture $\rho_{\boldsymbol{\mu},\boldsymbol{\Sigma}}$ is a distribution over $\mathbb{R}^{d+L}$. In other words,

$$h_{\boldsymbol{\mu},\boldsymbol{\Sigma}}(x) = K^{-1}\sum_{k\in[K]}\mathbb{E}_{(\omega,\beta)\sim\mathcal{N}(\mu_k,\Sigma_k)}[\omega\mathrm{ReLU}(\langle\beta, x\rangle)]. \tag{4}$$

However, the number of parameters becomes $\Theta((d+L)^2K)$, which is prohibitively large.

Reducing the number of parameters

To reduce the number of parameters, we propose to incorporate sparsity into the model by considering only diagonal covariance matrices for $\beta : \beta \sim \mathcal{N}(\mu^\beta, \text{diag}(\sigma^2))$ where $\text{diag}(\sigma^2)$ is a diagonal matrix whose entries are given by the vector $\sigma^2 = \sigma \odot \sigma \in \mathbb{R}^d$. Moreover, for jointly Gaussian (ω, β), the conditional mean of ω given β is affine: $\mathbb{E}[\omega|\beta] = U\beta + v$. Since Equation 4 only requires to know the conditional expectation $\mathbb{E}[\omega|\beta]$, we model each component of the Gaussian mixture as

$$\beta \sim \mathcal{N}(\mu^\beta, \text{diag}(\sigma^2)), \ \mathbb{E}[\omega|\beta] = U\beta + v. \tag{5}$$

Reducing the number of parameters

The trainable parameters for such a component are $\mu \in \mathbb{R}^d$, $\sigma \in \mathbb{R}^d$, $U \in \mathbb{R}^{L \times d}$, $v \in \mathbb{R}^L$. Hence, the parameters for the full Gaussian mixture are $\boldsymbol{\theta} := (\boldsymbol{\mu}^\beta, \boldsymbol{\sigma}, \boldsymbol{U}, \boldsymbol{v}) = \{(\mu_k^\beta, \sigma_k, U_k, v_k)\}_{k \in [K]}$. This leads to

$$h_{\boldsymbol{\theta}}(x) = K^{-1} \sum_{k \in [K]} \mathbb{E}_{\beta \sim \mathcal{N}(\mu_k^\beta, \mathrm{diag}(\sigma^2))} [(U\beta + v)\mathrm{ReLU}(\langle \beta, x \rangle)]. \tag{6}$$

This use of Equ. 6 cuts down the number of parameters to $\Theta(dKL)$.

An additional benefit is that parametrization by $\boldsymbol{\sigma}$ automatically maintains the positive semidefiniteness of the covariances during training, without recourse to costly projection steps.

Stacking GM layers

Since the input and output dimensions are arbitrary, this construction can be readily composed with other types of layers—including GM layers themselves—in order to build up deep neural network architectures.

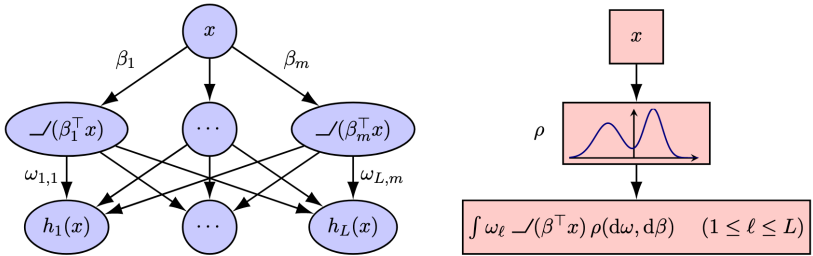

Figure 1: A GM layer (right) can act as a replacement for a fully connected layer (left).

Outline

Setting

- **Dataset:** MNIST and Fashion-MNIST. Both datasets consist of 60,000 training examples and 10,000 test examples, where each example is a 28×28 grayscale image, associated with a label from one of 10 classes.
- **Setup:** We consider $K \in \{5, 10, 20\}$ for different experiments. For a GM layer with parameters μ^β, σ, U and v, we initialize the entries of μ^β, U and v i.i.d. from $\mathcal{N}(0, \gamma^2)$, and the entries of σ all equal to γ, for some $\gamma > 0$. For most of the experiments we fix $\gamma = 1/2$ unless otherwise mentioned. We train the network using SGD with batch size 64 and fixed learning rate 1 for the parameter σ and 0.1 for all other parameters.

Test error

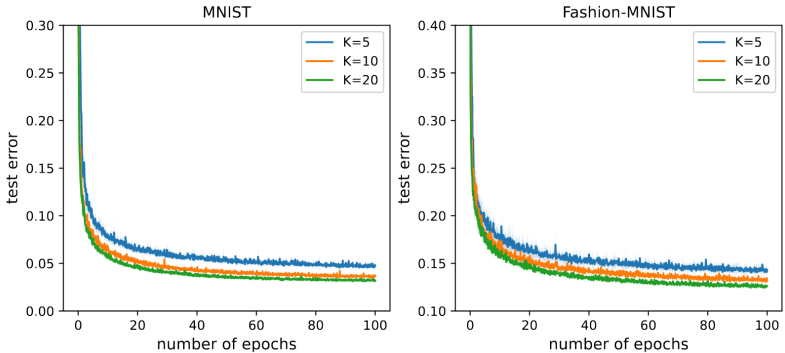

Figure 2: Test error (with error bars) for a GM Network with $K = 5, 10, 20$ number of components vs. the number of epochs. The left (resp. right) panel shows the result for MNIST (resp. Fashion-MNIST) dataset. The error bars are computed over 5 independent trials.

Test error

The width of the 2-layer fully connected network is 1000. $K = 20$ for the GM network.

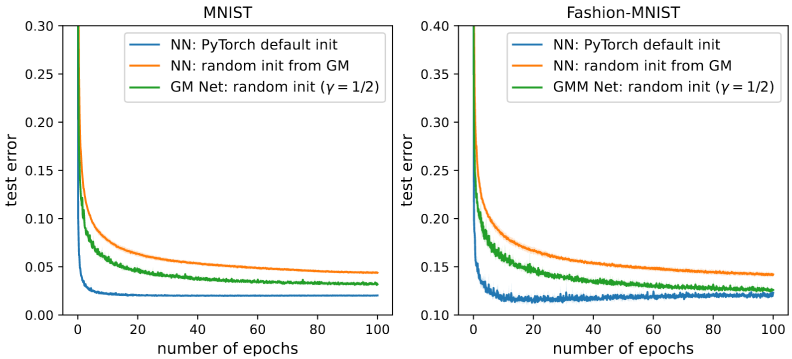

Figure 3: Test error (with error bars) for a 2-layer fully connected network using two different initialization schemes vs. the number of epochs. The left (resp. right) panel shows the results for the MNIST (resp. Fashion-MNIST) dataset. The error bars are computed over 5 independent trials.

Evolution of Gaussian components

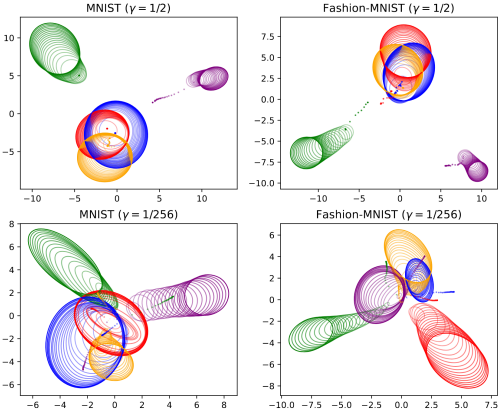

Figure 4: The evolution of the Gaussian components (marginalized over β) and weights β_j of the neurons, projected onto the top 2 PCs of the final GM distribution, across 200 epochs of training. The number of components for the GM net (resp. number of neurons in the 2-layer neural network) is $K = 5$. The projected Gaussian components are represented by their covariance ellipses centered at their means, while the projected weights of the neurons are depicted as dots. We use the same color for the evolution of the same Gaussian component and neuron, with increasing opacity as the number of epochs increases. The left (resp. right) plots show results for MNIST (resp. Fashion-MNIST), while the top (resp. bottom) plots use initialization scale $\gamma = 1/2$ (resp. $\gamma = 1/256$).

Experiments

To inspect whether the training of a network with a single GM layer is in the neural tangent kernel (NTK) a.k.a "lazy training" regime, or the mean-field regime. The "lazy training" is defined as when the distribution over the "first layer weights" β does not significantly move away from its initialization.

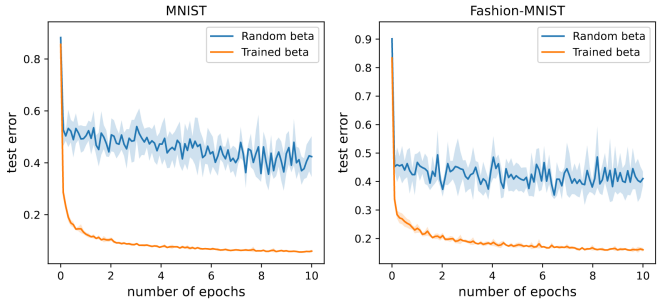

Figure 5: Test error (with error bars) for training with or without updating the marginal distribution over β vs. the number of epochs. The left (resp. right) panel shows the result for the MNIST (resp. Fashion-MNIST) dataset. The error bars are computed over 5 independent trials.

Experiments

The input dimension is $28 \times 28 = 768$. The intermediate dimension is 100.

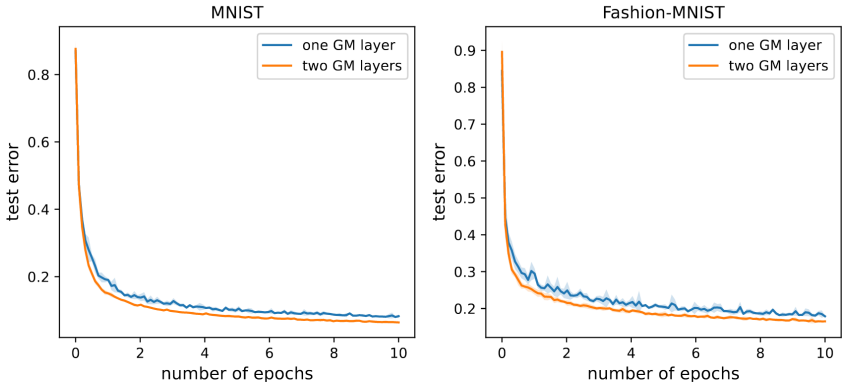

Figure 7: Test error (with error bars) for networks with one or two GM layers vs. the number of epochs. The left (resp. right) panel shows the result for the MNIST (resp. Fashion-MNIST) dataset. The error bars are computed over 5 independent trials.

Outline

Conclusion

- This paper introduced a novel layer type for neural network architectures – the Gaussian Mixture (GM) layers by using mean-field theory.
- The poor experimental results may be attributed to constructing the covariance matrix as a diagonal matrix, which ignores the interrelationships between different dimensions.