Learning in Gated Neural Networks

Ashok Vardhan Makkuva

University of Illinois at Urbana-Champaign

Gated Recurrent Neural Networks

- Well-known examples: LSTM and GRU
- State-of-the-art results in many challenging ML tasks

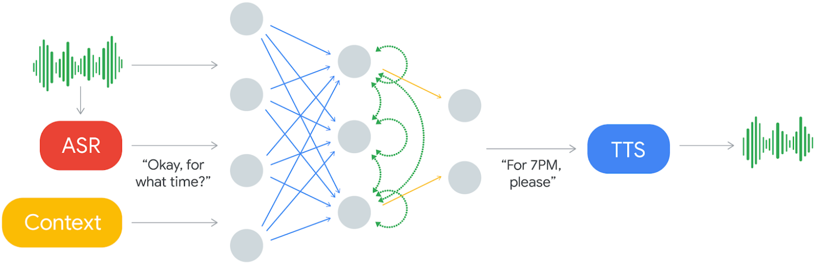
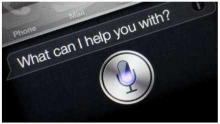


Figure: Google Duplex

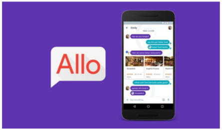
Demo

Siri, Alexa and more...

- Language translation
- Speech recognition

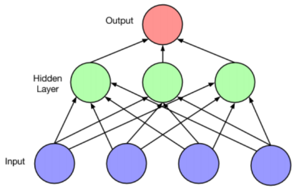


- Phrase completion

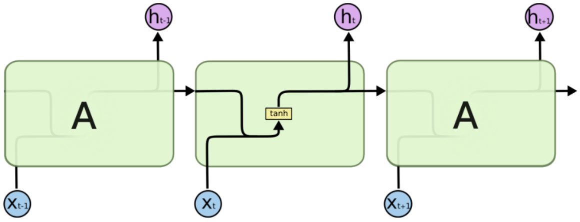


NNs and RNNs

- Feed-forward neural networks



- Recurrent neural networks (Vanilla)



Gated RNNs

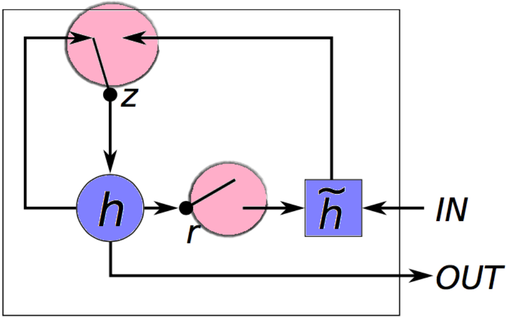
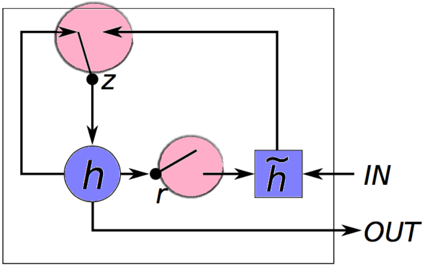


Figure: Gated Recurrent Unit (GRU)

Key features:

- **Gating** mechanism
- Non-linear 'switching' dynamical systems
- Long term memory

GRU



- **Gates:** $z_t, r_t \in [0,1]^d$ depend on the input x_t and the past h_{t-1}
- **States:** $h_t, \tilde{h}_t \in \mathbb{R}^d$

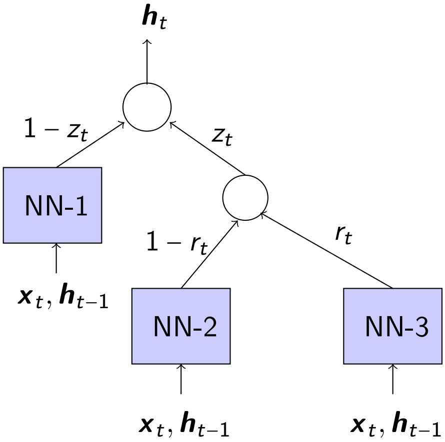
Update equations for each t:

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t$$

$$\tilde{h}_t = f(Ax_t + r_t \odot Bh_{t-1})$$

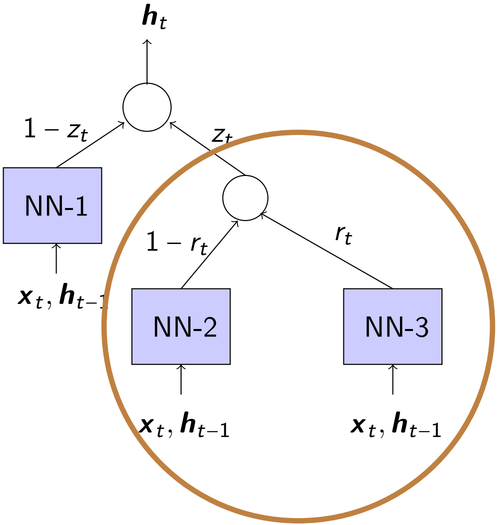
Building blocks of GRU

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot (1 - r_t) \odot f(Ax_t) + z_t \odot r_t \odot f(Ax_t + Bh_{t-1})$$



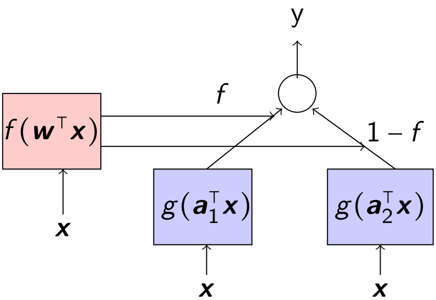
Building blocks of GRU

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot (1 - r_t) \odot f(Ax_t) + z_t \odot r_t \odot f(Ax_t + Bh_{t-1})$$



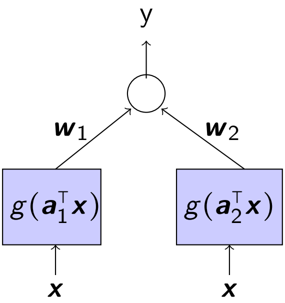
Mixture-of-Experts: Building blocks of GRU

- Jacobs, Jordan, Nowlan and Hinton, 1991

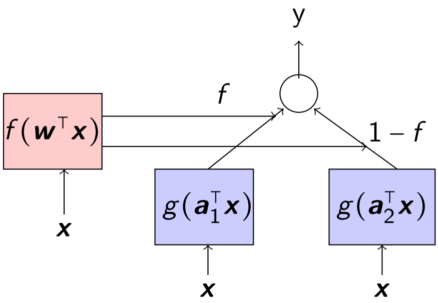


$f = \text{sigmoid}, \ g = \text{linear}, \text{tanh}, \text{ReLU}$

MoE as gated feed-forward network



(a) 2-node NN



(b) 2-MoE

MoE: Modern relevance

- Outrageously large neural networks

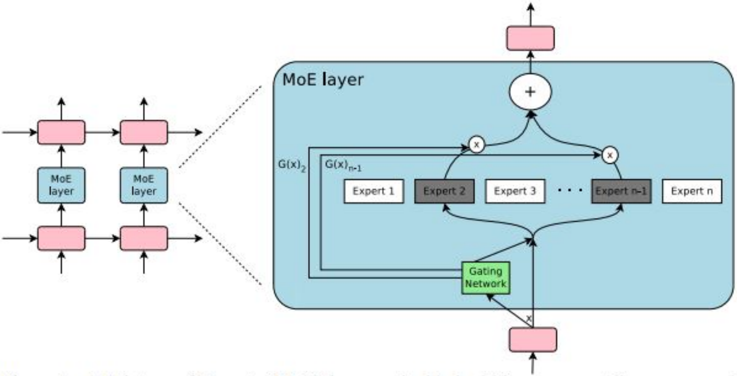


Figure 1: A Mixture of Experts (MoE) layer embedded within a recurrent language model. In this case, the sparse gating function selects two experts to perform computations. Their outputs are modulated by the outputs of the gating network.

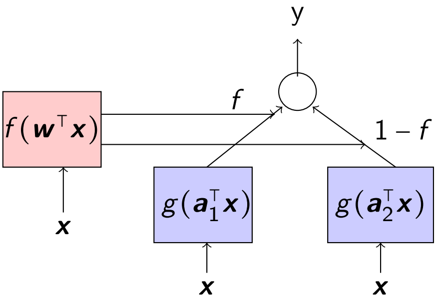
What is known about MoE?

Adaptive mixtures of local experts RA Jacobs, MI Jordan, SJ Nowlan, GE Hinton Neural computation 3 (1), 79-87	3663	1991
Sharing clusters among related groups: Hierarchical Dirichlet processes YW Teh, MI Jordan, MJ Beal, DM Blei Advances in neural information processing systems, 1385-1392	3273	2005
Hierarchical mixtures of experts and the EM algorithm MI Jordan, RA Jacobs Neural computation 6 (2), 181-214	3090	1994

- No provable learning algorithms for parameters[1] ☹

[1] 20 years of MoE, MoE: a literature survey

Open problem for 25+ years



$$\Leftrightarrow P_{y|\boldsymbol{x}} = f(\boldsymbol{w}^\top \boldsymbol{x}) \cdot \mathcal{N}(y|g(\boldsymbol{a}_1^\top \boldsymbol{x}), \sigma^2) + (1 - f(\boldsymbol{w}^\top \boldsymbol{x})) \cdot \mathcal{N}(y|g(\boldsymbol{a}_2^\top \boldsymbol{x}), \sigma^2)$$

Open question

Given n i.i.d. samples $(\boldsymbol{x}^{(i)}, y^{(i)})$, does there exist an efficient learning algorithm with provable theoretical guarantees to learn the regressors $\boldsymbol{a}_1, \boldsymbol{a}_2$ and the gating parameter $\boldsymbol{w}$?

Traditional loss functions

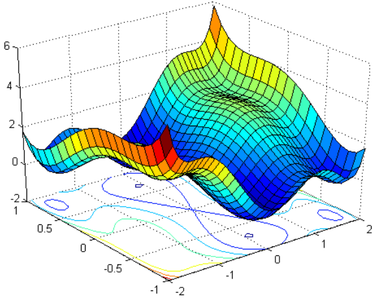
Loss functions:

- Log-likelihood loss

$$L = \log\left(f(\boldsymbol{w}^\top \boldsymbol{x}) \cdot e^{-\frac{\|y - g(\boldsymbol{a}_1^\top \boldsymbol{x})\|^2}{2\sigma^2}} + (1 - f(\boldsymbol{w}^\top \boldsymbol{x})) \cdot e^{-\frac{\|y - g(\boldsymbol{a}_2^\top \boldsymbol{x})\|^2}{2\sigma^2}} \right)$$

- L_2-loss

$$L = \left(y - \left(f(\boldsymbol{w}^\top \boldsymbol{x}) g(\boldsymbol{a}_1^\top \boldsymbol{x}) + (1 - f(\boldsymbol{w}^\top \boldsymbol{x})) g(\boldsymbol{a}_2^\top \boldsymbol{x})\right)\right)^2$$



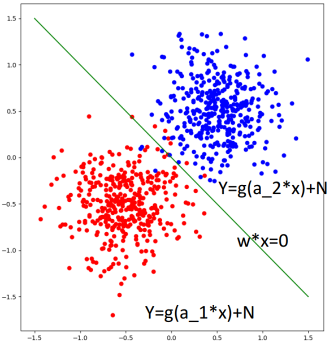
Traditional algorithms

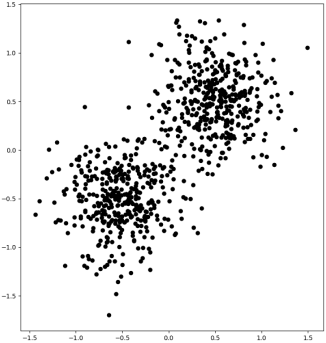
Algorithms: EM, Gradient descent, and their variants

- Practical: Often get stuck in local optima
- Theoretical: Loss surface is hard to analyze because of coupling of $\boldsymbol{w}$ and $(\boldsymbol{a}_1, \boldsymbol{a}_2)$. Just understood for far simpler problem of Gaussian mixtures

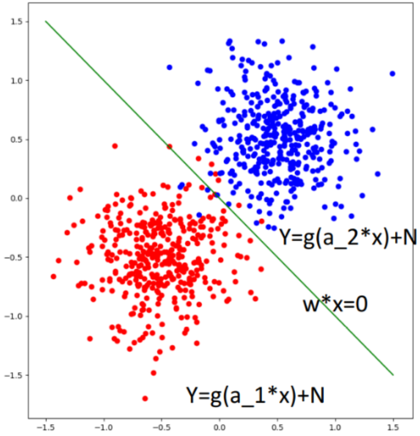
Modular structure

Mixture of classification ($\boldsymbol{w}$) and regression ($\boldsymbol{a}_1, \boldsymbol{a}_2$) problems





Key observation



Key observation

If we know the regressors, learning the gating parameter is easy and vice-versa. How to break the gridlock?

Focus of this talk: Breaking the gridlock

- **First** learning guarantees for MoE
- Two novel approaches to learn the parameters:

Method 1: Algorithms

We propose a novel algorithm with first recoverable guarantees

Method 2: Optimization framework

We design a non-trivial loss function on which traditional algorithms like GD converge to true parameters

- Both approaches work with **global initializations**
  - restriction: $\boldsymbol{x}$ is Gaussian

Generalizability

k-MoE

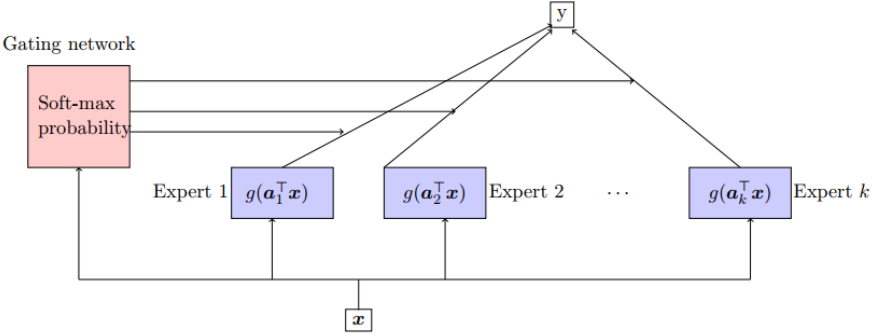


Figure 1: Architecture for k-MoE

Generalizability

Hierarchical mixture of experts (HME)

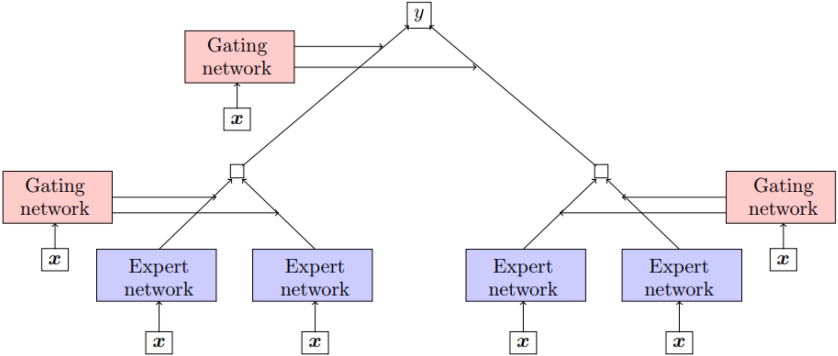


Figure 2: A two-level hierarchical mixture of experts

Method 1: Design of algorithms

Algorithmic approach: An overview

Recall the model for MoE:

$$P_{y|\boldsymbol{x}} = f(\boldsymbol{w}^\top \boldsymbol{x}) \cdot \mathcal{N}(y|g(\boldsymbol{a}_1^\top \boldsymbol{x}), \sigma^2) + (1 - f(\boldsymbol{w}^\top \boldsymbol{x})) \cdot \mathcal{N}(y|g(\boldsymbol{a}_2^\top \boldsymbol{x}), \sigma^2)$$

- We learn $(\boldsymbol{a}_1, \boldsymbol{a}_2)$ and $\boldsymbol{w}$ separately
- First recover $(\boldsymbol{a}_1, \boldsymbol{a}_2)$ without knowing $\boldsymbol{w}$ at all
- Later learn $\boldsymbol{w}$ using traditional methods like EM
- Global consistency guarantees (population setting)

Learning regressors without gating

Model for MoE:

$$P_{y|\boldsymbol{x}} = f(\boldsymbol{w}^\top \boldsymbol{x}) \cdot \mathcal{N}(y|g(\boldsymbol{a}_1^\top \boldsymbol{x}), \sigma^2) + (1 - f(\boldsymbol{w}^\top \boldsymbol{x})) \cdot \mathcal{N}(y|g(\boldsymbol{a}_2^\top \boldsymbol{x}), \sigma^2)$$

Without gating:

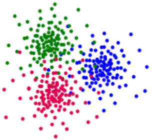
$$P_{y|\boldsymbol{x}} = p \cdot \mathcal{N}(y|g(\boldsymbol{a}_1^\top \boldsymbol{x}), \sigma^2) + (1 - p) \cdot \mathcal{N}(y|g(\boldsymbol{a}_2^\top \boldsymbol{x}), \sigma^2)$$

- Mixture of generalized linear models (GLMs)!
  - How do we learn $\boldsymbol{a}_1$ and $\boldsymbol{a}_2$ without knowing p?
  - Method of moments

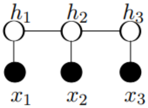
Tensor methods in latent variable models

- Anandkumar, Ge, Hsu, Kakade, and Telgarsky 2014

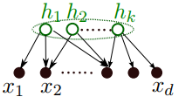
GMM



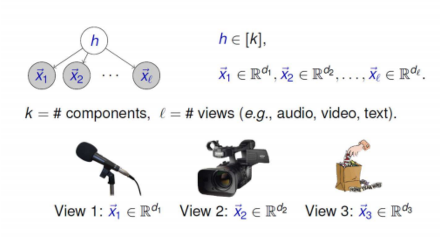
HMM



ICA



Multiview and Topic Models



Tensor methods in GLMs

Mixture of linear regression

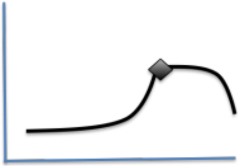
Noiseless – Yi et al '16
Local guarantee for noisy case:
Balakrishnan, Wainwright, Yu '17

Mixture of GLM (generalized linear model)

Sedghi, Janzamin and Anandkumar
AISTATS '16

Mixture of Experts

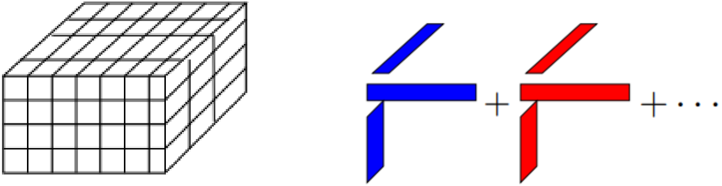
Open



Main approach

- Basic idea: Construct a **third-order super-symmetric** tensor from data such that

$$\mathbb{E}(\psi(X,Y)) = \sum_i \boldsymbol{a}_i \otimes \boldsymbol{a}_i \otimes \boldsymbol{a}_i \Rightarrow \boldsymbol{a}_i \text{ can be recovered}$$



- How do we construct ψ?
  - Stein's lemma

Stein's lemma 101

Stein's lemma

For $f : \mathbb{R}^d \to \mathbb{R}$ and $\boldsymbol{x} \sim \mathcal{N}(0, I_d)$,

$$\mathbb{E}[f(\boldsymbol{x}) \cdot \boldsymbol{x}] = \mathbb{E}[\nabla_{\boldsymbol{x}} f(\boldsymbol{x})] \in \mathbb{R}^d.$$

Non-linear regression using Stein's lemma: If $y = g(\boldsymbol{a}_1^\top \boldsymbol{x}) + N$, then

$$\begin{aligned} \underbrace{\mathbb{E}[y \cdot \boldsymbol{x}]}_{\text{Estimated from samples}} &= \mathbb{E}[g(\boldsymbol{a}_1^\top \boldsymbol{x}) \cdot \boldsymbol{x}] + \underbrace{\mathbb{E}[N \cdot \boldsymbol{x}]}_{=0} \\ &= \mathbb{E}[\nabla_x g(\boldsymbol{a}_1^\top \boldsymbol{x})] \\ &\propto \boldsymbol{a}_1 \end{aligned}$$

Mixture of GLMs: Stein's lemma 101

- Recall, for mixture of GLMs:

$$P_{y|\boldsymbol{x}} = p \cdot \mathcal{N}(y|g(\boldsymbol{a}_1^\top \boldsymbol{x}), \sigma^2) + (1-p) \cdot \mathcal{N}(y|g(\boldsymbol{a}_2^\top \boldsymbol{x}), \sigma^2)$$

- From Stein's lemma,

$$\mathbb{E}[y \cdot \boldsymbol{x}] \propto p \cdot \boldsymbol{a}_1 + (1-p) \cdot \boldsymbol{a}_2.$$

- Not unique in $\boldsymbol{a}_1$ and $\boldsymbol{a}_2$
- How can we ensure uniqueness?

Stein's lemma 102

2nd order Stein's lemma

$$\mathbb{E}[f(\boldsymbol{x}) \cdot \underbrace{(\boldsymbol{x}\boldsymbol{x}^\top - I)}_{\mathcal{S}_2(\boldsymbol{x})}] = \mathbb{E}[\nabla_{\boldsymbol{x}}^{(2)} f(\boldsymbol{x})] \in \mathbb{R}^{d \times d}.$$

- Mixture of GLMs:

$$P_{y|\boldsymbol{x}} = p \cdot \mathcal{N}(y|g(\boldsymbol{a}_1^\top \boldsymbol{x}), \sigma^2) + (1-p) \cdot \mathcal{N}(y|g(\boldsymbol{a}_2^\top \boldsymbol{x}), \sigma^2)$$
$$\Rightarrow \mathbb{E}[y \cdot (\boldsymbol{x}\boldsymbol{x}^\top - I)] \propto 2p \cdot \boldsymbol{a}_1 \boldsymbol{a}_1^\top + 2(1-p) \cdot \boldsymbol{a}_2 \boldsymbol{a}_2^\top.$$

- Not unique!
- How can we ensure uniqueness?

Stein's lemma 103

3rd order Stein's lemma

$$\mathbb{E}[f(\boldsymbol{x}) \cdot \mathcal{S}_3(\boldsymbol{x})] = \mathbb{E}[\nabla_{\boldsymbol{x}}^{(3)} f(\boldsymbol{x})] \in \mathbb{R}^{d \times d \times d}$$

- Score transformation $\mathcal{S}_3(\boldsymbol{x}) = \boldsymbol{x} \otimes \boldsymbol{x} \otimes \boldsymbol{x} - \sum_{i \in [d]} \mathrm{sym}(\boldsymbol{x} \otimes \boldsymbol{e}_i \otimes \boldsymbol{e}_i)$

- Mixture of GLMs:

$$P_{y|\boldsymbol{x}} = p \cdot \mathcal{N}(y | g(\boldsymbol{a}_1^\top \boldsymbol{x}), \sigma^2) + (1-p) \cdot \mathcal{N}(y | g(\boldsymbol{a}_2^\top \boldsymbol{x}), \sigma^2)$$
$$\Rightarrow \mathbb{E}[y \cdot \mathcal{S}_3(\boldsymbol{x})] \propto p \cdot \boldsymbol{a}_1 \otimes \boldsymbol{a}_1 \otimes \boldsymbol{a}_1 + (1-p) \cdot \boldsymbol{a}_2 \otimes \boldsymbol{a}_2 \otimes \boldsymbol{a}_2.$$

- Unique! (by Kruskal's theorem)
- Can we extend this to MoE?

MoE: Stein's lemma

- For MoE, $p = p(x) = f(\boldsymbol{w}^\top \boldsymbol{x})$ since

$$P_{y|\boldsymbol{x}} = f(\boldsymbol{w}^\top \boldsymbol{x}) \cdot \mathcal{N}(y|g(\boldsymbol{a}_1^\top \boldsymbol{x}), \sigma^2) + (1 - f(\boldsymbol{w}^\top \boldsymbol{x})) \cdot \mathcal{N}(y|g(\boldsymbol{a}_2^\top \boldsymbol{x}), \sigma^2)$$

- Can we use Stein's lemma to learn $\boldsymbol{a}_1$ and $\boldsymbol{a}_2$?
- Natural attempt:

$$\mathbb{E}[y \cdot S_3(\boldsymbol{x})] = \boldsymbol{a}_1 \otimes \boldsymbol{a}_1 \otimes \boldsymbol{a}_1 + \boldsymbol{w} \otimes \boldsymbol{a}_1 \otimes \boldsymbol{w} + \ldots + \boldsymbol{a}_1 \otimes \boldsymbol{a}_1 \otimes \boldsymbol{w} + \ldots$$

  Not a super-symmetric tensor

- Can we construct a super-symmetric tensor for MoE?

Key insight: Hermite polynomial transformation

Suppose g =linear and $\sigma = 0$. Then

$$P_{y|\boldsymbol{x}} = f(\boldsymbol{w}^\top \boldsymbol{x}) \cdot \mathbb{1}\{y = \boldsymbol{a}_1^\top \boldsymbol{x}\} + (1 - f(\boldsymbol{w}^\top \boldsymbol{x}))\mathbb{1}\{y = \boldsymbol{a}_1^\top \boldsymbol{x}\}$$

$$\Rightarrow \mathbb{E}[y^3 - 3y|\boldsymbol{x}] = \sum_{i \in \{1,2\}} f(\boldsymbol{w}_i^\top \boldsymbol{x})((\boldsymbol{a}_i^\top \boldsymbol{x})^3 - 3(\boldsymbol{a}_i^\top \boldsymbol{x})), \quad \boldsymbol{w}_2 = -\boldsymbol{w}_1$$

Key insight: Hermite polynomial transformation

Suppose g =linear and $\sigma = 0$. Then

$$P_{y|\boldsymbol{x}} = f(\boldsymbol{w}^\top \boldsymbol{x}) \cdot \mathbb{1}\{y = \boldsymbol{a}_1^\top \boldsymbol{x}\} + (1 - f(\boldsymbol{w}^\top \boldsymbol{x}))\mathbb{1}\{y = \boldsymbol{a}_1^\top \boldsymbol{x}\}$$

$$\Rightarrow \mathbb{E}[y^3 - 3y|\boldsymbol{x}] = \sum_{i \in \{1,2\}} f(\boldsymbol{w}_i^\top \boldsymbol{x})((\boldsymbol{a}_i^\top \boldsymbol{x})^3 - 3(\boldsymbol{a}_i^\top \boldsymbol{x})), \quad \boldsymbol{w}_2 = -\boldsymbol{w}_1$$

Now applying Stein's lemma,

$$\mathbb{E}[(y^3 - 3y) \cdot \mathcal{S}_3(\boldsymbol{x})] = \mathbb{E}[\nabla_{\boldsymbol{x}}^3 \mathbb{E}[y^3 - 3y|\boldsymbol{x}]] = 3 \sum_{i \in \{1,2\}`} \boldsymbol{a}_i \otimes \boldsymbol{a}_i \otimes \boldsymbol{a}_i$$

How do cross terms like $\boldsymbol{a}_i \otimes \boldsymbol{a}_i \otimes \boldsymbol{w}$ disappear?

- Reason: $\mathbb{E}[H_3'(Z)] = \mathbb{E}[H_3''(Z)] = \mathbb{E}[H_3'''(Z)] = 0$
- $H_3(z) = z^3 - 3z$ is third-Hermite polynomial

Does this work for $\sigma \neq 0$?

Linear experts: Hermite-like-polynomials

Suppose g = linear and $\sigma \neq 0$:

$$P_{y|\boldsymbol{x}} = f(\boldsymbol{w}^\top \boldsymbol{x}) \cdot \mathcal{N}(y|\boldsymbol{a}_1^\top \boldsymbol{x}, \sigma^2) + (1 - f(\boldsymbol{w}^\top \boldsymbol{x})) \cdot \mathcal{N}(y|\boldsymbol{a}_2^\top \boldsymbol{x}, \sigma^2)$$

Super-symmetric tensor

$$\mathcal{T}_3 = \mathbb{E}[(y^3 - 3y(1+\sigma^2)) \cdot \mathcal{S}_3(\boldsymbol{x})] = 3(\boldsymbol{a}_1 \otimes \boldsymbol{a}_1 \otimes \boldsymbol{a}_1 + \boldsymbol{a}_2 \otimes \boldsymbol{a}_2 \otimes \boldsymbol{a}_2)$$

- This very much needs special linear structure. What about other non-linearities for g?

Generalization: Cubic polynomial transformations

- For a wide class of non-linearities such as g=linear, sigmoid, ReLU, etc.

$$\mathcal{T}_3 = \mathbb{E}[(y^3 + \alpha y^2 + \beta y) \cdot \mathcal{S}_3(\boldsymbol{x})] = c(\boldsymbol{a}_1 \otimes \boldsymbol{a}_1 \otimes \boldsymbol{a}_1 + \boldsymbol{a}_2 \otimes \boldsymbol{a}_2 \otimes \boldsymbol{a}_2)$$

- How do we choose α and β?
  - Solving a linear system
  - **Example:** For sigmoid,

$$\begin{bmatrix} 0.2067 & 0.2066 \\ 0.0624 & -0.0001 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} -0.1755 - 0.6199\sigma^2 \\ -0.0936 \end{bmatrix}$$

- **Key idea:** Acts like a 'Hermite' like polynomial for general g and cancels cross terms

Learning regressors: Spectral decomposition

Algorithm

- Input: Samples $(\boldsymbol{x}_i, y_i)$
- Compute $\hat{\mathcal{T}}_3 = (1/n) \sum_i H_3(y_i) \cdot \mathcal{S}_3(\boldsymbol{x}_i)$
- $\hat{\boldsymbol{a}}_1, \hat{\boldsymbol{a}}_2$ = Rank-2 decomposition on $\mathcal{T}_3$

Learning the gating

- Recall

$$P_{y|\boldsymbol{x}} = f(\boldsymbol{w}^\top \boldsymbol{x}) \cdot \mathcal{N}(y|\boldsymbol{a}_1^\top \boldsymbol{x}, \sigma^2) + (1 - f(\boldsymbol{w}^\top \boldsymbol{x})) \cdot \mathcal{N}(y|\boldsymbol{a}_2^\top \boldsymbol{x}, \sigma^2)$$

- If we know $\boldsymbol{a}_1$ and $\boldsymbol{a}_2$, learning $\boldsymbol{w}$ is a classification problem!
- Traditional methods:
  - EM algorithm
  - Gradient descent on log-likelihood

Theoretical contributions

- Show global convergence for existing methods
- Provide convergence rate
- Finite sample complexity
- First theoretical guarantees

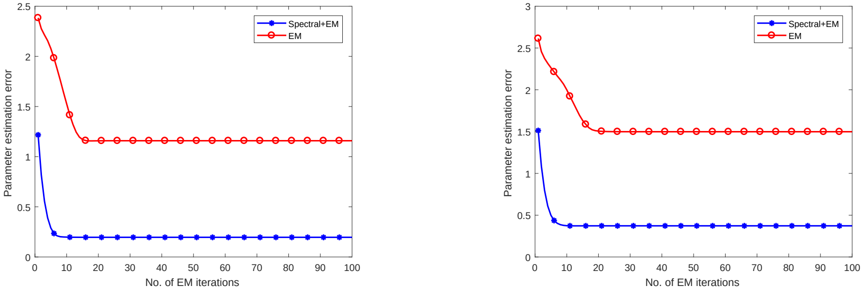
Learning the gating parameters

Suppose spectral methods give $\hat{\boldsymbol{a}}_i$ with $\|\hat{\boldsymbol{a}}_i - \boldsymbol{a}_i\|_2 \leq \sigma^2\varepsilon$

For high SNR, i.e. $\sigma < \sigma_0$, σ_0 is a dimension independent constant:

- EM iterates converge geometrically to $\hat{\boldsymbol{w}}$
- Convergence rate is a dimension-independent constant depending on σ and $\|\boldsymbol{a}_1 - \boldsymbol{a}_2\|$
- $\hat{\boldsymbol{w}}$ is ε-close to the ground truth

Comparison with EM



(a) 3 mixtures (b) 4 mixtures

Figure: Plot of parameter estimation error

Method 2: Optimization framework-loss function design

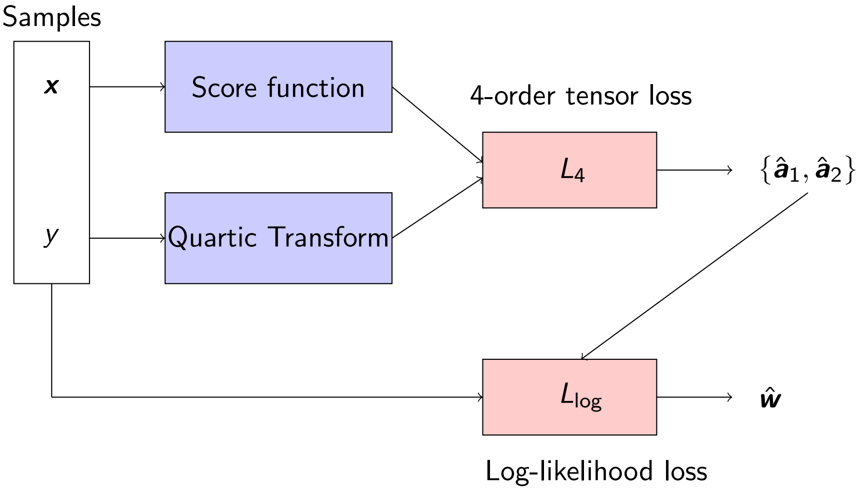
Regressors: Loss function design

$$P_{y|\boldsymbol{x}} = f(\boldsymbol{w}^\top \boldsymbol{x}) \cdot \mathcal{N}(y|g(\boldsymbol{a}_1^\top \boldsymbol{x}), \sigma^2) + (1 - f(\boldsymbol{w}^\top \boldsymbol{x})) \cdot \mathcal{N}(y|g(\boldsymbol{a}_2^\top \boldsymbol{x}), \sigma^2)$$

- Traditional approaches: l_2-loss, log-likelihood loss
  - Get stuck in local minima
  - No theoretical analysis
  - Single loss function for both $(\boldsymbol{a}_1, \boldsymbol{a}_2)$ and $\boldsymbol{w}$
- Formulation of right loss function is critical (Jacobs et. al 1991)

Theoretical contributions

- Separate loss functions L_4 and $L_{\log}$ to learn $(\boldsymbol{a}_1, \boldsymbol{a}_2)$ and $\boldsymbol{w}$



- Gradient descent on both L_4 and $L_{\log}$. What are they?

Tensor based loss function for regressors

- For linear experts,

$$P_{y|\boldsymbol{x}} = f(\boldsymbol{w}^\top \boldsymbol{x}) \cdot \mathcal{N}(y|\boldsymbol{a}_1^\top \boldsymbol{x}, \sigma^2) + (1 - f(\boldsymbol{w}^\top \boldsymbol{x})) \cdot \mathcal{N}(y|\boldsymbol{a}_2^\top \boldsymbol{x}, \sigma^2)$$

- Stein's lemma+ 4-Hermite polynomial implies

$$\mathcal{T}_4 = \mathbb{E}[(y^4 - 6y^2(1+\sigma^2)) \cdot \mathcal{S}_4(\boldsymbol{x})] = 12(\boldsymbol{a}_1^{\otimes 4} + \boldsymbol{a}_2^{\otimes 4})$$

- If $\hat{\boldsymbol{a}}_1$ and $\hat{\boldsymbol{a}}_2$ are parameters,

$$L_4(\hat{\boldsymbol{a}}_1, \hat{\boldsymbol{a}}_2) \triangleq \sum_{j \neq k} \mathcal{T}_4(\hat{\boldsymbol{a}}_j, \hat{\boldsymbol{a}}_j, \hat{\boldsymbol{a}}_k, \hat{\boldsymbol{a}}_k) - \mu \sum_{j \in \{1,2\}} \mathcal{T}_4(\hat{\boldsymbol{a}}_j, \hat{\boldsymbol{a}}_j, \hat{\boldsymbol{a}}_j, \hat{\boldsymbol{a}}_j) + \lambda \sum_{j \in \{1,2\}} (\|\hat{\boldsymbol{a}}_j\|^2 - 1)^2$$

Landscape of L_4

Properties

- No spurious local minima: All local minima are global
- Global minima are ground truth (upto permutation and sign-flip)
- All saddle points have negative curvature
- SGD converges to approximate global minima

Why L_4?

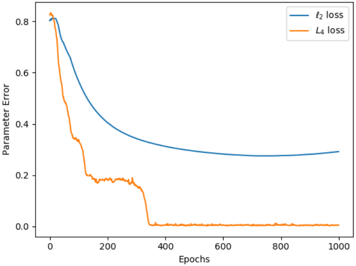
Why L_4?

- We provide a non-trivial connection to tensor based losses
- We can show that

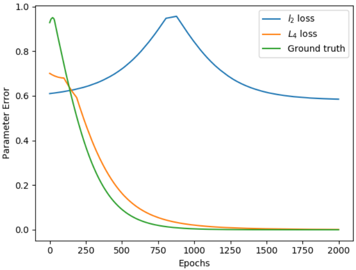
$$L_4(\hat{\boldsymbol{a}}_1, \hat{\boldsymbol{a}}_2) = 12 \sum_i \sum_{j \neq k} \langle \boldsymbol{a}_i, \hat{\boldsymbol{a}}_j \rangle^2 \langle \boldsymbol{a}_i, \hat{\boldsymbol{a}}_k \rangle^2 - 12\mu \sum_i \sum_j \langle \boldsymbol{a}_i, \hat{\boldsymbol{a}}_j \rangle^4 + \lambda \sum_j (\|\boldsymbol{a}_j\|^2 - 1)^2$$

- 4-order tensor loss
  - Landscape analysis in (Ge et. al 2018)

Empirical performance



(a) ℓ_2 vs. L_4



(b) ℓ_2 vs. $L_{\log}$

Figure: Plot of parameter estimation error

Summary

- Algorithmic innovation: First provably consistent algorithms for MoE in 25+ years
- Loss function innovation: First SGD based algorithm on novel loss functions with provably nice landscape properties
- Sample complexity: First sample complexity results for MoE
- Global convergence: Our algorithms work with global initializations

Open questions-I

Conjecture

EM algorithm recovers both the regression parameters $\boldsymbol{a}_1, \boldsymbol{a}_2$ and gating parameter $\boldsymbol{w}$ globally for 2-MoE

It is known that EM learns the true parameters globally for

- 2-symmetric mixture of Gaussians (Xu 2016, Daskalakis 2017)
- 2-symmetric mixture of linear regressions

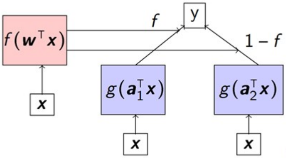
Open questions-II

- Minimax rates and optimal algorithms
- Learning algorithms for time-series?
- Generalizing to non-Gaussian inputs
  - Results: In the absence of gating, we have a loss function framework to provably learn the regressors
  - With gating?

References

- Breaking the gridlock in Mixture-of-Experts: Consistent and Efficient Algorithms
- Learning One-hidden-layer Neural Networks under General Input Distributions
- Learning in Gated Neural Networks

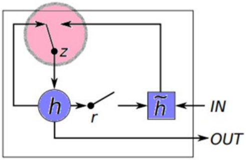
Conclusion



Mixture-of-Experts

1. Theoretical understanding ✓
2. Novel algorithms ✓

?



(b) Gated Recurrent Unit

Gated recurrent networks

1. Theoretical understanding **?**
2. Algorithms **?**

Thank you!