A Responsible Softmax Layer in Deep Learning

Ryan Coatney

University of Arizona

18 June 2020

Overview

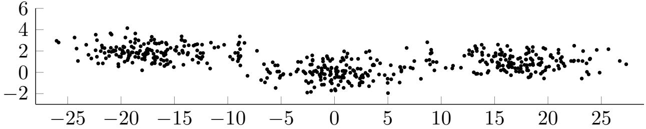
Clustering and Classification

Dynamic Responsibility

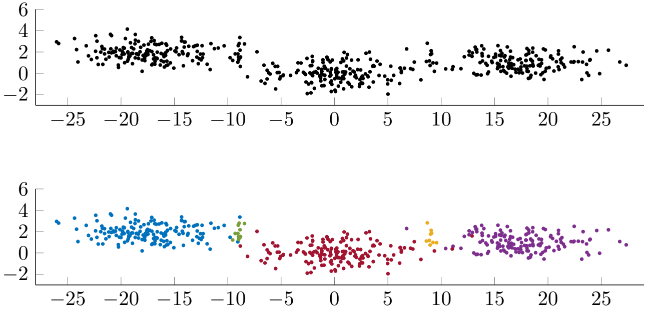
Responsible Softmax

Basic Experiments

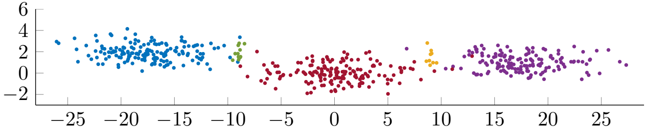
Clustering



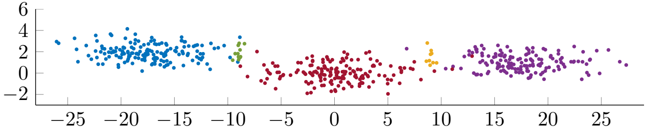
Clustering

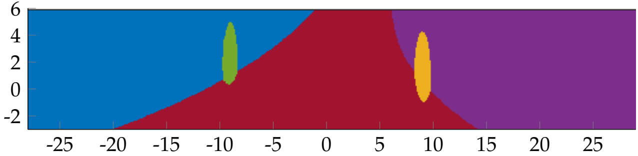


Classification



Classification





K-means Algorithm

Start with data $\mathcal{X} = \{\boldsymbol{x}^1, \ldots, \boldsymbol{x}^N\}$ and K starting 'means', $\{\boldsymbol{m}_1, \ldots, \boldsymbol{m}_K\}$.

Assignment: For each data point, $\boldsymbol{x}^n$, set $\hat{k}_n = \arg\min_k d(\boldsymbol{x}^n, \boldsymbol{m}_k)$. Set $\rho_i^n = \delta_i^{\hat{k}_n}$ (Hard responsibility).

Update: Let $N_k = \sum_{n=1}^N \rho_k^n$ and

$$\boldsymbol{m}_k^{new} = \frac{\sum_{n=1}^N \rho_k^n \boldsymbol{x}^{(n)}}{N_k}.$$

If $d(\boldsymbol{m}_k, \boldsymbol{m}_k^{new})$ is small for every k, stop. Otherwise set $\boldsymbol{m}_k = \boldsymbol{m}_k^{new}$ and repeat. See MacKay (2002) for more discussion.

K-means Algorithm

Start with data $\mathcal{X} = \{\boldsymbol{x}^1, \ldots, \boldsymbol{x}^N\}$ and K starting 'means', $\{\boldsymbol{m}_1, \ldots, \boldsymbol{m}_K\}$.

Assignment: For each data point, $\boldsymbol{x}^n$, set $\hat{k}_n = \arg\min_k d(\boldsymbol{x}^n, \boldsymbol{m}_k)$. Set $\rho_i^n = \delta_i^{\hat{k}_n}$ (Hard responsibility).

Update: Let $N_k = \sum_{n=1}^N \rho_k^n$ and

$$\boldsymbol{m}_k^{new} = \frac{\sum_{n=1}^N \rho_k^n \boldsymbol{x}^{(n)}}{N_k}.$$

If $d(\boldsymbol{m}_k, \boldsymbol{m}_k^{new})$ is small for every k, stop. Otherwise set $\boldsymbol{m}_k = \boldsymbol{m}_k^{new}$ and repeat. See MacKay (2002) for more discussion.

K-means Algorithm

Start with data $\mathcal{X} = \{\boldsymbol{x}^1, \ldots, \boldsymbol{x}^N\}$ and K starting 'means', $\{\boldsymbol{m}_1, \ldots, \boldsymbol{m}_K\}$.

Assignment: For each data point, $\boldsymbol{x}^n$, set $\hat{k}_n = \arg\min_k d(\boldsymbol{x}^n, \boldsymbol{m}_k)$. Set $\rho_i^n = \delta_i^{\hat{k}_n}$ (Hard responsibility).

Update: Let $N_k = \sum_{n=1}^N \rho_k^n$ and

$$\boldsymbol{m}_k^{new} = \frac{\sum_{n=1}^N \rho_k^n \boldsymbol{x}^{(n)}}{N_k}.$$

If $d(\boldsymbol{m}_k, \boldsymbol{m}_k^{new})$ is small for every k, stop. Otherwise set $\boldsymbol{m}_k = \boldsymbol{m}_k^{new}$ and repeat. See MacKay (2002) for more discussion.

K-means Algorithm

Start with data $\mathcal{X} = \{\boldsymbol{x}^1, \ldots, \boldsymbol{x}^N\}$ and K starting 'means', $\{\boldsymbol{m}_1, \ldots, \boldsymbol{m}_K\}$.

Assignment: For each data point, $\boldsymbol{x}^n$, set $\hat{k}_n = \arg\min_k d(\boldsymbol{x}^n, \boldsymbol{m}_k)$. Set $\rho_i^n = \delta_i^{\hat{k}_n}$ (Hard responsibility).

Update: Let $N_k = \sum_{n=1}^N \rho_k^n$ and

$$\boldsymbol{m}_k^{new} = \frac{\sum_{n=1}^N \rho_k^n \boldsymbol{x}^{(n)}}{N_k}.$$

If $d(\boldsymbol{m}_k, \boldsymbol{m}_k^{new})$ is small for every k, stop. Otherwise set $\boldsymbol{m}_k = \boldsymbol{m}_k^{new}$ and repeat. See MacKay (2002) for more discussion.

K-means Algorithm

Start with data $\mathcal{X} = \{\boldsymbol{x}^1, \dots, \boldsymbol{x}^N\}$ and K starting 'means', $\{\boldsymbol{m}_1, \dots, \boldsymbol{m}_K\}$.

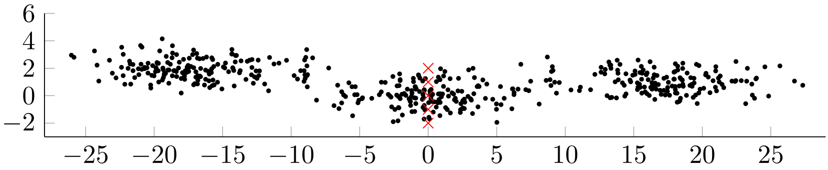
Assignment: For each data point, $\boldsymbol{x}^n$, set $\hat{k}_n = \arg\min_k d(\boldsymbol{x}^n, \boldsymbol{m}_k)$. Set $\rho_i^n = \delta_i^{\hat{k}_n}$ (Hard responsibility).

Update: Let $N_k = \sum_{n=1}^N \rho_k^n$ and

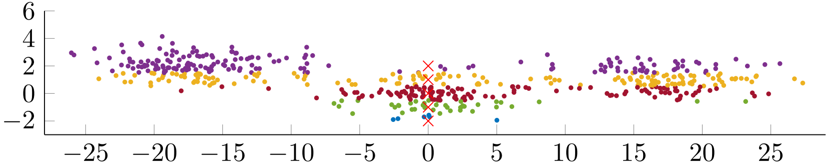
$$\boldsymbol{m}_k^{new} = \frac{\sum_{n=1}^N \rho_k^n \boldsymbol{x}^{(n)}}{N_k}.$$

If $d(\boldsymbol{m}_k, \boldsymbol{m}_k^{new})$ is small for every k, stop. Otherwise set $\boldsymbol{m}_k = \boldsymbol{m}_k^{new}$ and repeat. See MacKay (2002) for more discussion.

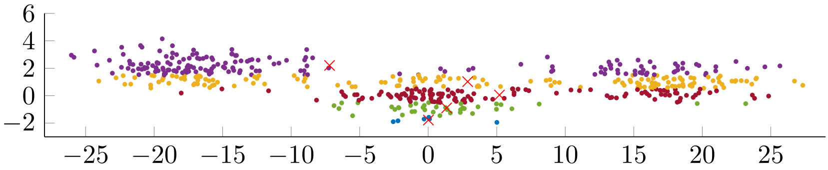
K-means Example



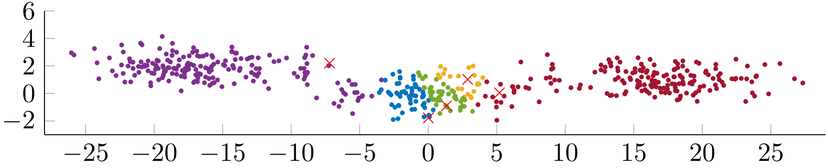
K-means Example



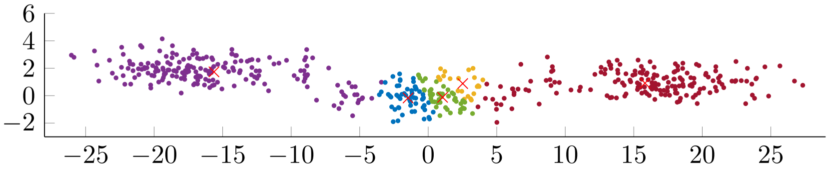
K-means Example



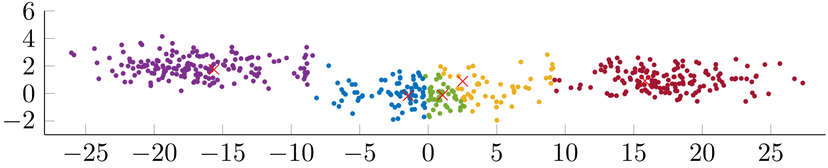
K-means Example



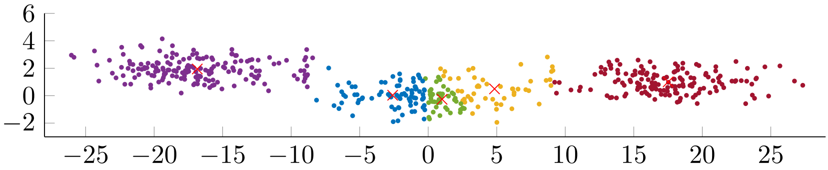
K-means Example



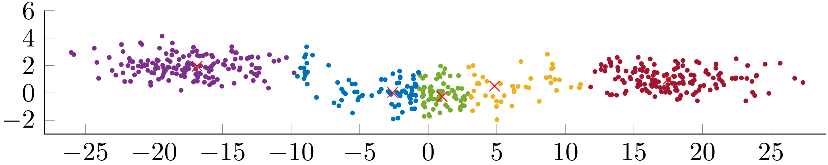
K-means Example



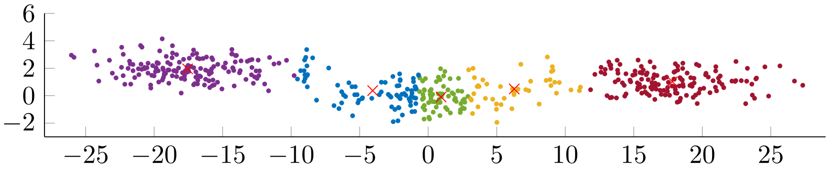
K-means Example



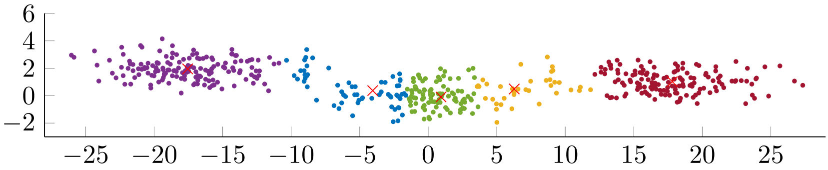
K-means Example



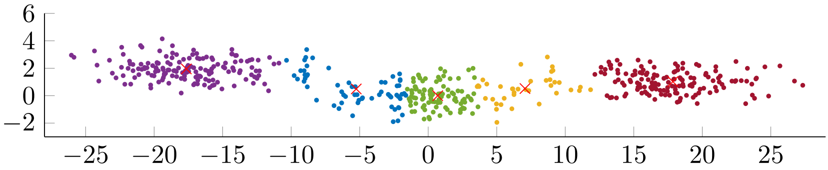
K-means Example



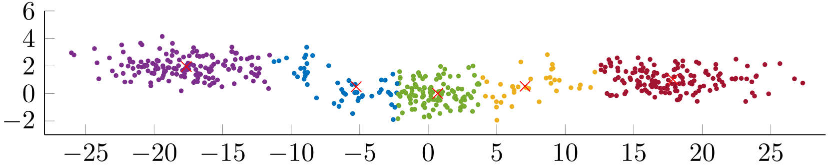
K-means Example



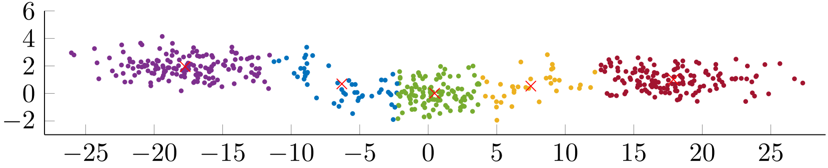
K-means Example



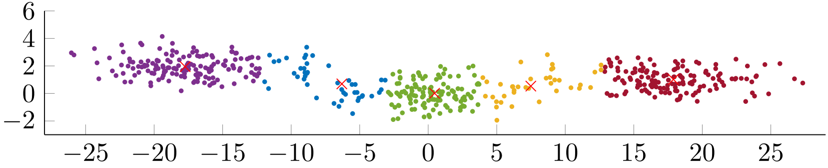
K-means Example



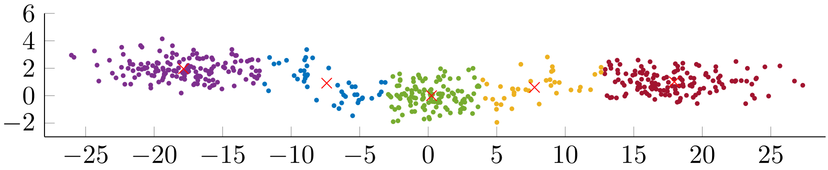
K-means Example



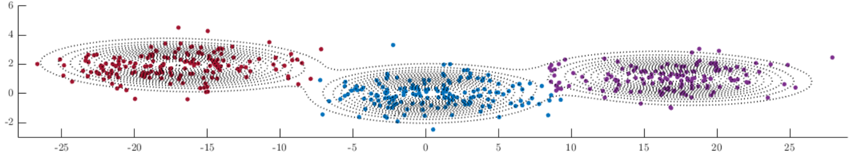
K-means Example



K-means Example

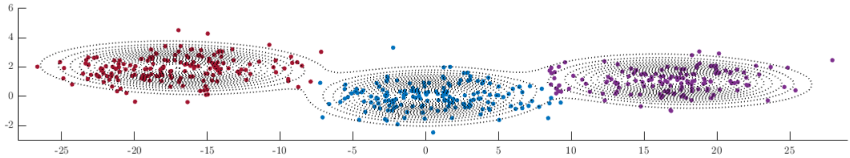


EM Algorithm on Gaussian Mixture

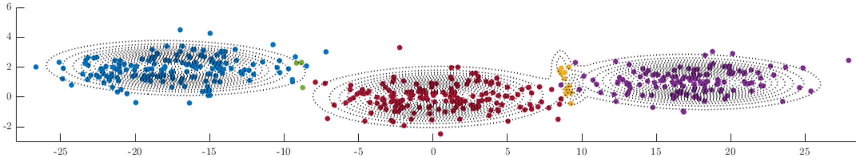


(a) AIC≈ 5030, BIC≈ 5102

EM Algorithm on Gaussian Mixture

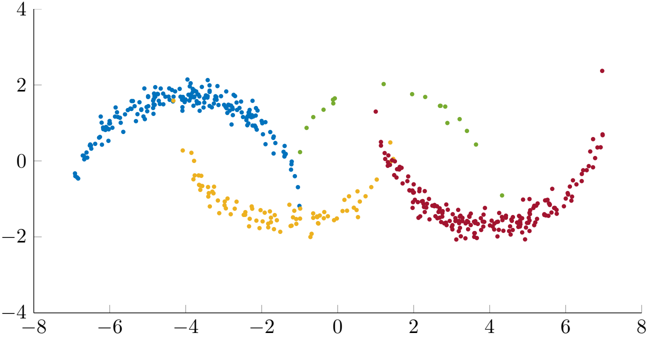


(a) AIC≈ 5030, BIC≈ 5102

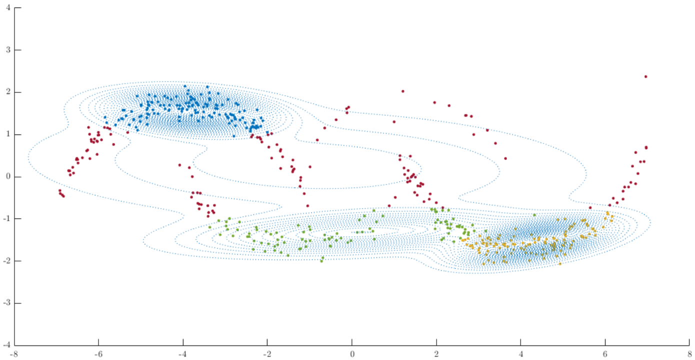


(b) AIC≈ 4994, BIC≈ 5116

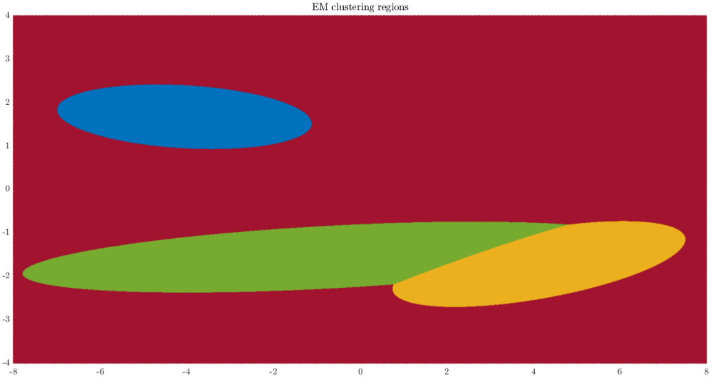
EM Algorithm on Non-Gaussian Data



EM Algorithm on Non-Gaussian Data



EM Algorithm on Non-Gaussian Data



Overview

Clustering and Classification

Dynamic Responsibility

Responsible Softmax

Basic Experiments

Responsibility Requirements

Necessary items

- Data $\{\mathcal{X}, \mathcal{T}\} = (\boldsymbol{x}^n, t^n)\ n = 1, \ldots, N$
- Distributions $f_k(\boldsymbol{x}, \boldsymbol{\theta}_k)\ k = 1, \ldots, K$
- Parameter matrix $F = \left(f_i(\boldsymbol{x}^j, \boldsymbol{\theta}_i)\right)_i^j = (F_i^j)$
- Mixture probabilities $\boldsymbol{\pi}_0 = (\pi_1, \ldots, \pi_K) \in S_K$

Definition (Probability Simplex)

$$S_K := \left\{ \{\pi_k\}_{k=1}^K : 0 \leq \pi_k \leq 1; \sum_{k=1}^K \pi_k = 1 \right\}.$$

Responsibility Requirements

Necessary items

- Data $\{\mathcal{X}, \mathcal{T}\} = (\boldsymbol{x}^n, t^n)\ n = 1, \dots, N$
- Distributions $f_k(\boldsymbol{x}, \boldsymbol{\theta}_k)\ k = 1, \dots, K$
- Parameter matrix $F = \big(f_i(\boldsymbol{x}^j, \boldsymbol{\theta}_i)\big)_i^j = (F_i^j)$
- Mixture probabilities $\boldsymbol{\pi}_0 = (\pi_1, \dots, \pi_K) \in S_K$

Definition (Probability Simplex)

$$S_K := \left\{ \{\pi_k\}_{k=1}^K : 0 \le \pi_k \le 1; \sum_{k=1}^K \pi_k = 1 \right\}.$$

Bayes' Rule Estimation of Mixture Probabilities

$$\begin{aligned} P(t^n = k|\boldsymbol{x}^n, \boldsymbol{\Theta}) &= \frac{P(\boldsymbol{x}^n|t^n = k, \boldsymbol{\Theta})P(t^n = k|\boldsymbol{\Theta})}{P(\boldsymbol{x}^n|\boldsymbol{\Theta})} \\ &= \frac{f_k(\boldsymbol{x}^n, \boldsymbol{\theta}_k)\pi_k}{\sum_i \pi_i f_i(\boldsymbol{x}^n, \boldsymbol{\theta}_i)} \end{aligned}$$

Responsibility

Start with rational maps

$$r_i(\boldsymbol{\pi}) = \frac{1}{N}\sum_n \frac{\pi_i f_i(\boldsymbol{x}^n, \boldsymbol{\theta}_k)}{\sum_k \pi_k f_k(\boldsymbol{x}^n, \boldsymbol{\theta}_k)} \quad i = 1, \ldots, K$$

Definition (Responsibility Map)

$$R : S_K \to S_K : R(\pi_1, \pi_2, \ldots, \pi_K) = (r_1(\boldsymbol{\pi}), r_2(\boldsymbol{\pi}), \ldots, r_K(\boldsymbol{\pi})). \tag{2.1}$$

When necessary, write $R_F(\boldsymbol{\pi})$ to emphasize dependence on $K \times N$ parameter matrix F.

Responsibility

Start with rational maps

$$r_i(\boldsymbol{\pi}) = \frac{1}{N}\sum_n \frac{\pi_i f_i(\boldsymbol{x}^n, \boldsymbol{\theta}_k)}{\sum_k \pi_k f_k(\boldsymbol{x}^n, \boldsymbol{\theta}_k)} \quad i = 1, \ldots, K$$

Definition (Responsibility Map)

$$R : S_K \to S_K : R(\pi_1, \pi_2, \ldots, \pi_K) = (r_1(\boldsymbol{\pi}), r_2(\boldsymbol{\pi}), \ldots, r_K(\boldsymbol{\pi})). \tag{2.1}$$

When necessary, write $R_F(\boldsymbol{\pi})$ to emphasize dependence on $K \times N$ parameter matrix F.

Dynamic Responsibility

Algorithm 1 Dynamic Responsibility Algorithm

Require: F a $K \times N$ matrix
Require: $\boldsymbol{\pi}_0$, ϵ ▷ ϵ creates halt condition
procedure Iteration$(F, \boldsymbol{\pi}_0, \epsilon)$
  $n \leftarrow 1$, $\boldsymbol{\pi}_n \leftarrow R_F(\boldsymbol{\pi}_0)$
  $orbit \leftarrow \{\boldsymbol{\pi}_0, \boldsymbol{\pi}_1\}$
  while $|\boldsymbol{\pi}_n - \boldsymbol{\pi}_{n-1}| > \epsilon |\boldsymbol{\pi}_n|$ **do**
    $\boldsymbol{\pi}_{n+1} \leftarrow R_F(\boldsymbol{\pi}_n)$
    $orbit \leftarrow \{\boldsymbol{\pi}_0, \ldots, \boldsymbol{\pi}_{n+1}\}$
    $n \leftarrow n + 1$
  end while
  return $orbit$ ▷ at this point $\boldsymbol{\pi}_{n-1} \approx \hat{\boldsymbol{\pi}}$
end procedure

Lyapunov function

For $\boldsymbol{\pi} \in \mathbb{R}_{+}^{K}$ the positive orthant of $\mathbb{R}^K$, let

$$\ell_F(\boldsymbol{\pi}) = \frac{1}{N} \sum_{n=1}^{N} \log \left(\sum_{k=1}^{K} \pi_k F_k^n \right)$$

Lemma

$-\ell_F(\boldsymbol{\pi})$ is a Lyapunov function for dynamic responsibility. In other words,

$$\ell_F(R_F(\boldsymbol{\pi})) \geq \ell_F(\boldsymbol{\pi})$$

With equality if and only if $R_F(\boldsymbol{\pi}) = \boldsymbol{\pi}$.

Note that if F has full rank, $-\ell_F$ is *strictly* convex.

Lyapunov function

For $\boldsymbol{\pi} \in \mathbb{R}_+^K$ the positive orthant of $\mathbb{R}^K$, let

$$\ell_F(\boldsymbol{\pi}) = \frac{1}{N} \sum_{n=1}^{N} \log \left(\sum_{k=1}^{K} \pi_k F_k^n \right)$$

Lemma

$-\ell_F(\boldsymbol{\pi})$ is a Lyapunov function for dynamic responsibility. In other words,

$$\ell_F(R_F(\boldsymbol{\pi})) \geq \ell_F(\boldsymbol{\pi})$$

With equality if and only if $R_F(\boldsymbol{\pi}) = \boldsymbol{\pi}$.

Note that if F has full rank, $-\ell_F$ is *strictly* convex.

Lyapunov function

For $\boldsymbol{\pi} \in \mathbb{R}_+^K$ the positive orthant of $\mathbb{R}^K$, let

$$\ell_F(\boldsymbol{\pi}) = \frac{1}{N} \sum_{n=1}^{N} \log \left(\sum_{k=1}^{K} \pi_k F_k^n \right)$$

Lemma

$-\ell_F(\boldsymbol{\pi})$ is a Lyapunov function for dynamic responsibility. In other words,

$$\ell_F(R_F(\boldsymbol{\pi})) \geq \ell_F(\boldsymbol{\pi})$$

With equality if and only if $R_F(\boldsymbol{\pi}) = \boldsymbol{\pi}$.

Note that if F has full rank, $-\ell_F$ is *strictly* convex.

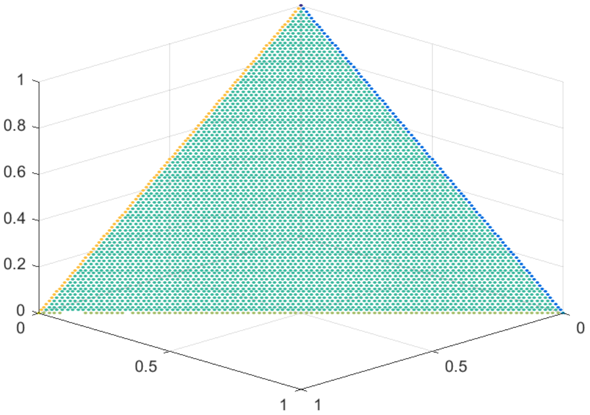
Main Theorem

Theorem (Convergence of dynamic responsibility)

If F has full rank, and $\boldsymbol{\pi}_0 \in \operatorname{Int} S_K$ then the orbit $\boldsymbol{\pi}^n = R_F^n(\boldsymbol{\pi}_0)$ converges to $\hat{\boldsymbol{\pi}}_F$, the unique maximizing fixed point of $\ell_F(\boldsymbol{\pi})$ on S_K. Moreover, $\hat{\boldsymbol{\pi}}_F$ depends differentiably on F.

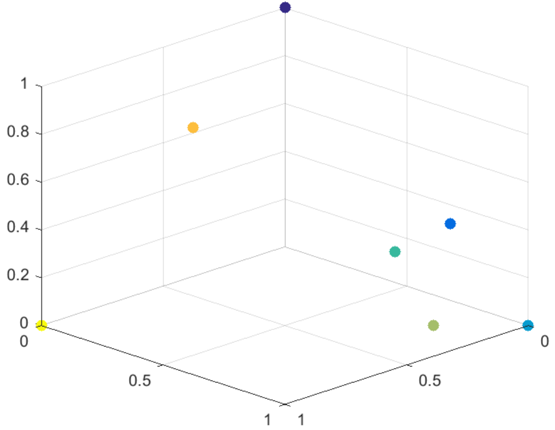
Dynamic Responsibility Example

If $F = (F_i^j)$ has linearly independent rows, the interior of S_K converges to one point.



Dynamic Responsibility Example

In this case, convergence happens very quickly. (about 5 iterations)



Overview

Clustering and Classification

Dynamic Responsibility

Responsible Softmax

Basic Experiments

Calculate F

If $F = e^{\boldsymbol{A}}$ for some $\boldsymbol{A} = (A_i^j)$ and $\mu_i = \ln(\pi_i)$, then

$$r_i(\boldsymbol{\pi}) = \frac{1}{N}\sum_n \frac{\pi_i F_i^n}{\sum_k \pi_k F_k^n} = \frac{1}{N}\sum_n \frac{\exp(A_i^n + \mu_i)}{\sum_k \exp\big(A_k^n + \mu_k\big)}$$

The softmax function is given by the Gibbs Distribution

$$\sigma_i(\boldsymbol{x}) = \frac{\exp(x_i)}{\sum_k \exp(x_k)}.$$

This establishes a connection with modern neural networks.

Calculate F

If $F = e^{\boldsymbol{A}}$ for some $\boldsymbol{A} = (A_i^j)$ and $\mu_i = \ln(\pi_i)$, then

$$r_i(\boldsymbol{\pi}) = \frac{1}{N}\sum_n \frac{\pi_i F_i^n}{\sum_k \pi_k F_k^n} = \frac{1}{N}\sum_n \frac{\exp(A_i^n + \mu_i)}{\sum_k \exp\big(A_k^n + \mu_k\big)}$$

The softmax function is given by the Gibbs Distribution

$$\sigma_i(\boldsymbol{x}) = \frac{\exp(x_i)}{\sum_k \exp(x_k)}.$$

This establishes a connection with modern neural networks.

Calculate F

If $F = e^{\boldsymbol{A}}$ for some $\boldsymbol{A} = (A_i^j)$ and $\mu_i = \ln(\pi_i)$, then

$$r_i(\boldsymbol{\pi}) = \frac{1}{N}\sum_n \frac{\pi_i F_i^n}{\sum_k \pi_k F_k^n} = \frac{1}{N}\sum_n \frac{\exp(A_i^n + \mu_i)}{\sum_k \exp\big(A_k^n + \mu_k\big)}$$

The softmax function is given by the Gibbs Distribution

$$\sigma_i(\boldsymbol{x}) = \frac{\exp(x_i)}{\sum_k \exp(x_k)}.$$

This establishes a connection with modern neural networks.

Calculate F

If $F = e^{\boldsymbol{A}}$ for some $\boldsymbol{A} = (A_i^j)$ and $\mu_i = \ln(\pi_i)$, then

$$r_i(\boldsymbol{\pi}) = \frac{1}{N}\sum_n \frac{\pi_i F_i^n}{\sum_k \pi_k F_k^n} = \frac{1}{N}\sum_n \frac{\exp(A_i^n + \mu_i)}{\sum_k \exp\big(A_k^n + \mu_k\big)}$$

The softmax function is given by the Gibbs Distribution

$$\sigma_i(\boldsymbol{x}) = \frac{\exp(x_i)}{\sum_k \exp(x_k)}.$$

This establishes a connection with modern neural networks.

Neural Network Output

Neural networks take in data, and output guesses of cluster assignments.

$$F = (F_i^j); \quad \boldsymbol{\pi}^n = R_F^n(\boldsymbol{\pi}_0); \quad \boldsymbol{\pi}^n \to \hat{\boldsymbol{\pi}} \text{ as } n \to \infty$$

$$Y(F, \hat{\boldsymbol{\pi}}) = \left(\frac{\hat{\pi}_i F_i^j}{\sum_{k=1}^K \hat{\pi}_k F_k^j} \right)_{i=1,\ldots,K}^{j=1,\ldots,N}$$

The entry Y_i^j represents the probability that $\boldsymbol{x}^j$ comes from cluster i.

For some F, it may be that $\hat{\boldsymbol{\pi}}_F \in \partial S_K$. To prevent this, stop at some finite $n = C < \infty$ and use $Y(F, \boldsymbol{\pi}^C)$ as the output.

See Neal and Hinton (1998) for inspiration.

Neural Network Output

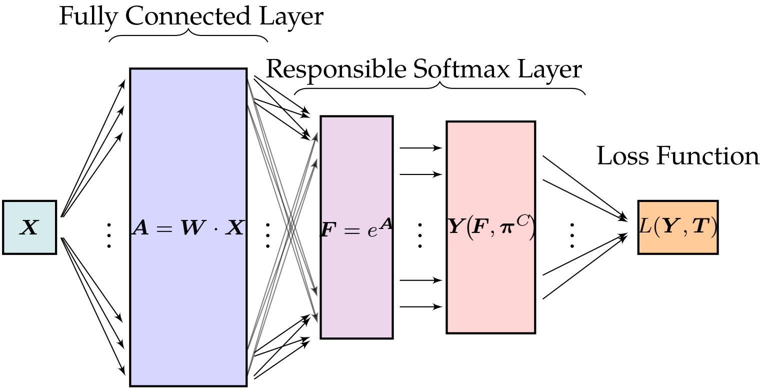
Neural networks take in data, and output guesses of cluster assignments.

$$F = (F_i^j); \quad \boldsymbol{\pi}^n = R_F^n(\boldsymbol{\pi}_0); \quad \boldsymbol{\pi}^n \to \hat{\boldsymbol{\pi}} \text{ as } n \to \infty$$

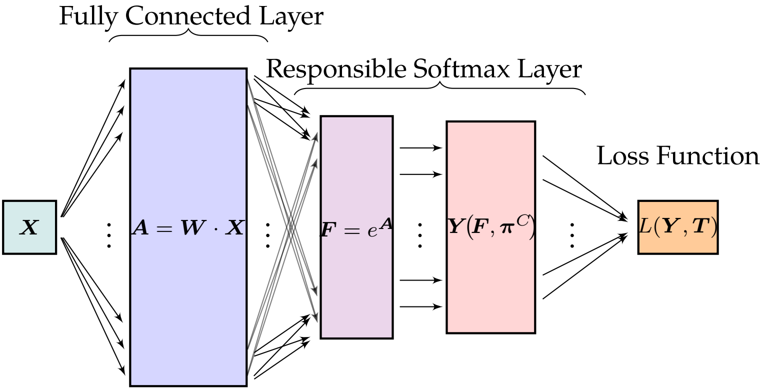
$$Y(F, \hat{\boldsymbol{\pi}}) = \left(\frac{\hat{\pi}_i F_i^j}{\sum_{k=1}^K \hat{\pi}_k F_k^j} \right)_{i=1,\ldots,K}^{j=1,\ldots,N}$$

The entry Y_i^j represents the probability that $\boldsymbol{x}^j$ comes from cluster i.
For some F, it may be that $\hat{\boldsymbol{\pi}}_F \in \partial S_K$. To prevent this, stop at some finite $n = C < \infty$ and use $Y(F, \boldsymbol{\pi}^C)$ as the output.
See Neal and Hinton (1998) for inspiration.

Layer Diagram



Layer Diagram



$$L(\boldsymbol{Y}, \boldsymbol{T}) = -\sum_n \sum_k T_k^n \log(Y_k^n)$$

Backpropagation

The goal is to use gradient descent to learn parameters of the network.

Option 1: Automatic differentiation

Option 2: Direct calculation

$$
\begin{aligned}
D\hat{\boldsymbol{\pi}}_F &= D_{\boldsymbol{\pi}}R \cdot D\hat{\boldsymbol{\pi}}_F + D_F R \\
D\hat{\boldsymbol{\pi}}_F &= (I - D_{\boldsymbol{\pi}}R)^{-1} \cdot D_F R
\end{aligned} \tag{3.1}
$$

In practice, equation (3.1) is too much. An approximation may be used instead.

Backpropagation

The goal is to use gradient descent to learn parameters of the network.

Option 1: Automatic differentiation

Option 2: Direct calculation

$$D\hat{\boldsymbol{\pi}}_F = D_{\boldsymbol{\pi}} R \cdot D\hat{\boldsymbol{\pi}}_F + D_F R$$

$$D\hat{\boldsymbol{\pi}}_F = (I - D_{\boldsymbol{\pi}} R)^{-1} \cdot D_F R \tag{3.1}$$

In practice, equation (3.1) is too much. An approximation may be used instead. $(I - D_{\boldsymbol{\pi}} R)^{-1} \approx I + DR + DR^2 + \ldots + DR^C$

Setting the Hyperparameter C

Let $a_n = d(\boldsymbol{\pi}_{n+1}, \boldsymbol{\pi}_n)$.

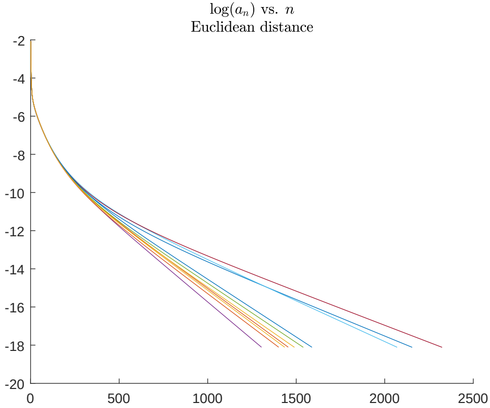


Figure: Plot of $\log(a_n)$ for several F. Each curve represents a different parameter matrix F.

Overview

Clustering and Classification

Dynamic Responsibility

Responsible Softmax

Basic Experiments

Experiments with GMM

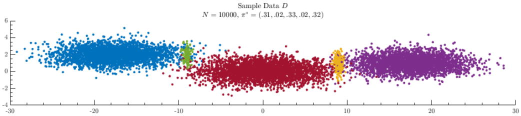
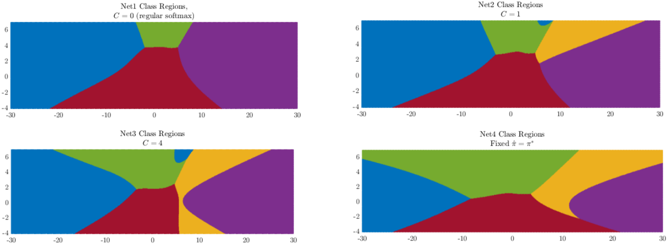


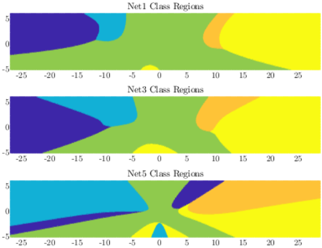
Figure: A sample of data generated from a GMM to test the responsibility softmax layer.

Experiments with GMM



Net	Classification layer
Net #1	Softmax
Net #2	Responsibility Softmax; $C = 1$
Net #3	Responsibility Softmax; $C = 4$
Net #4	Fixed Weight Softmax

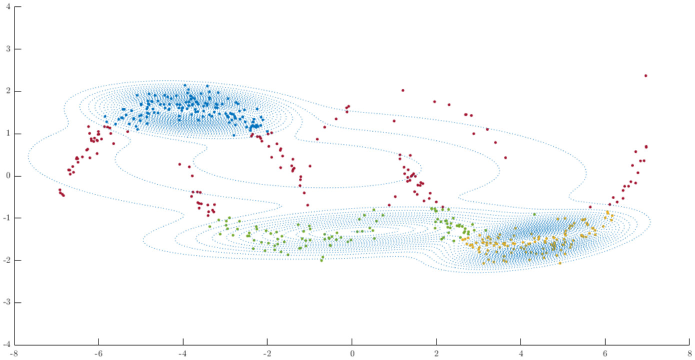
Experiments with GMM



Net	Classification layer
Net #1	Softmax
Net #2	Responsibility Softmax; $C = 1$
Net #3	Responsibility Softmax; $C = 4$
Net #4	Responsibility Softmax; $C = 8$
Net #5	Responsibility Softmax; $C = 16$
Net #6	Fixed Weight Softmax

Non-Gaussian Data Set

Recall the performance of the EM algorithm on Crescent data



Non-Gaussian Data Set

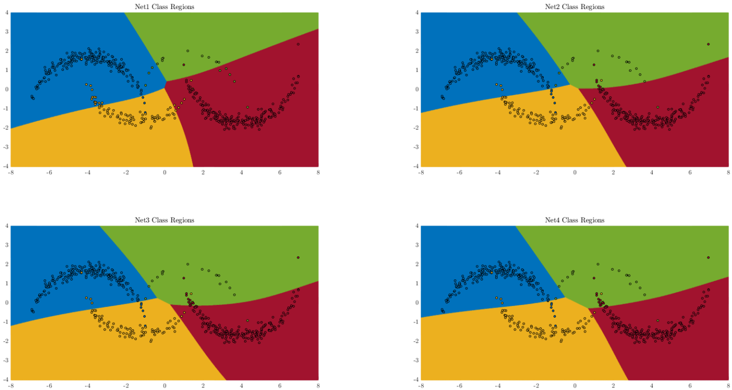
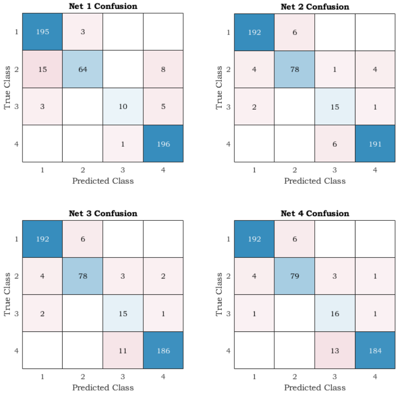


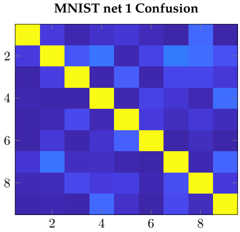
Figure: Classification regions for neural nets trained on crescent data. Hyperparametes are as in GMM example.

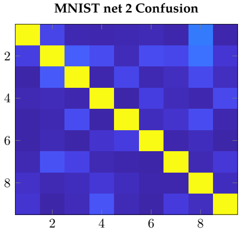
Non-Gaussian Data Set

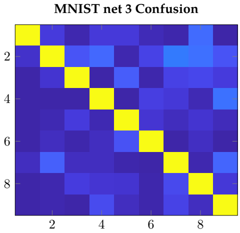


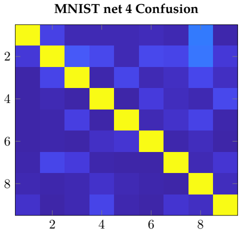
Net	Classification layer
Net #1	Softmax
Net #2	Responsibility Softmax $C = 1$
Net #3	Responsibility Softmax $C = 4$
Net #4	Fixed Weight Softmax

Experiments with MNIST









Conclusions

We have shown that:

- **Dynamic responsibility** has nice convergence properties; converges to a MLE.
- The **responsibility softmax** layer uses dynamic responsibility and gives cluster responsibilities.
- Using a responsibility softmax layer gives better results when working with imbalanced data. It also works when we do not have distributions for the mixture populations.

Future Work

Future work:

- Use responsibility softmax with other neural nets, LSTM, VAE, Deductron etc.
- Use responsibility softmax with nonparametric models (*e.g.* Gaussian processes).
- Obtain constructive bounds on convergence rates.
- Explore the relationship between hessian of ℓ_F and Fisher Information matrix.

References

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Information science and statistics. Springer, 1st ed. 2006. corr. 2nd printing edition.

Deisenroth, M. P., Faisal, A. A., and Ong, C. S. (2020). *Mathematics for Machine Learning*. Cambridge University Press.

MacKay, D. J. C. (2002). *Information Theory, Inference & Learning Algorithms*. Cambridge University Press, New York, NY, USA.

Neal, R. M. and Hinton, G. E. (1998). *A View of the EM Algorithm that Justifies Incremental, Sparse, and other Variants*, pages 355–368. Springer Netherlands, Dordrecht.

EM ALGORITHM FOR GMM

$$\mathcal{X} = \{\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^N\}$$
$$f_k(\boldsymbol{x}) \sim \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \quad k = 1, \ldots, K$$
$$p(t^n = k) = \pi_k, \quad \sum_k \pi_k = 1$$

EM ALGORITHM FOR GMM

1. **Expectation** step: Set

$$\rho_k^n = \frac{\pi_k f_k(\boldsymbol{x}^{(n)})}{\sum_{j=1}^K \pi_j f_j(\boldsymbol{x}^{(n)})}$$

2. **Maximization** step: Set

$$N_k = \sum_{n=1}^N \rho_k^n, \quad \boldsymbol{\mu}_k^{new} = \frac{1}{N_k}\sum_{n=1}^N \rho_k^n \boldsymbol{x}^{(n)}$$

$$\pi_k^{new} = \frac{N_k}{N}, \quad \boldsymbol{\Sigma}_k^{new} = \frac{1}{N_k}\sum_{n=1}^N \rho_k^n (\boldsymbol{x}^{(n)} - \boldsymbol{\mu}_k^{new})(\boldsymbol{x}^{(n)} - \boldsymbol{\mu}_k^{new})^\intercal$$

3. Repeat steps 1 and 2 until convergence.

See Bishop (2006) for more details.

Proof of Lyapunov Lemma

Lemma

The map $R_F(\boldsymbol{\pi})$ *as defined in equation* (2.1) *satisfies*

$$R_F(\boldsymbol{\pi}) = \left(\pi_i \cdot \left. \frac{\partial \ell_F}{\partial \pi_i} \right|_{\boldsymbol{\pi}} \right)_{1 \le i \le K}$$

Proof of Lyapunov Lemma

$$\begin{aligned}
\ell_F(R_F(\boldsymbol{\pi})) - \ell_F(\boldsymbol{\pi}) &= \frac{1}{N}\sum_{n=1}^{N} \log\left\{\frac{\sum_{i=1}^{K} \pi_i F_i^n \frac{\partial \ell}{\partial \pi_i}}{\sum_{k=1}^{K} \pi_k F_k^n}\right\} \\
&\geq \sum_{n=1}^{N}\sum_{i=1}^{K} \frac{1}{N} \frac{\pi_i F_i^n}{\sum_{k=1}^{K} \pi_k f_{kn}} \log\left(\frac{\partial \ell}{\partial \pi_i}\right) \\
&= \sum_{i=1}^{K}\sum_{n=1}^{N} \frac{1}{N} \frac{\pi_i F_i^n}{\sum_{k=1}^{K} \pi_k f_{kn}} \log\left(\frac{\partial \ell}{\partial \pi_i}\right) \\
&= \sum_{i=1}^{K} r_i(\boldsymbol{\pi}) \log\left(\frac{r_i(\boldsymbol{\pi})}{\pi_i}\right) \geq 0
\end{aligned}$$

Proof of Lyapunov Lemma

$$
\begin{aligned}
\ell_F(R_F(\boldsymbol{\pi})) - \ell_F(\boldsymbol{\pi}) &= \frac{1}{N}\sum_{n=1}^{N} \log\left\{\frac{\sum_{i=1}^{K} \pi_i F_i^n \frac{\partial \ell}{\partial \pi_i}}{\sum_{k=1}^{K} \pi_k F_k^n}\right\} \\
&\geq \sum_{n=1}^{N}\sum_{i=1}^{K} \frac{1}{N} \frac{\pi_i F_i^n}{\sum_{k=1}^{K} \pi_k f_{kn}} \log\left(\frac{\partial \ell}{\partial \pi_i}\right) \\
&= \sum_{i=1}^{K}\sum_{n=1}^{N} \frac{1}{N} \frac{\pi_i F_i^n}{\sum_{k=1}^{K} \pi_k f_{kn}} \log\left(\frac{\partial \ell}{\partial \pi_i}\right) \\
&= \sum_{i=1}^{K} r_i(\boldsymbol{\pi}) \log\left(\frac{r_i(\boldsymbol{\pi})}{\pi_i}\right) \geq 0
\end{aligned}
$$

Proof of Lyapunov Lemma

$$
\begin{aligned}
\ell_F(R_F(\boldsymbol{\pi})) - \ell_F(\boldsymbol{\pi}) &= \frac{1}{N}\sum_{n=1}^{N} \log\left\{ \frac{\sum_{i=1}^{K} \pi_i F_i^n \frac{\partial \ell}{\partial \pi_i}}{\sum_{k=1}^{K} \pi_k F_k^n} \right\} \\
&\geq \sum_{n=1}^{N}\sum_{i=1}^{K} \frac{1}{N} \frac{\pi_i F_i^n}{\sum_{k=1}^{K} \pi_k f_{kn}} \log\left(\frac{\partial \ell}{\partial \pi_i}\right) \\
&= \color{red}{\sum_{i=1}^{K}\sum_{n=1}^{N}} \frac{1}{N} \frac{\pi_i F_i^n}{\sum_{k=1}^{K} \pi_k f_{kn}} \log\left(\frac{\partial \ell}{\partial \pi_i}\right) \\
&= \sum_{i=1}^{K} r_i(\boldsymbol{\pi}) \log\left(\frac{r_i(\boldsymbol{\pi})}{\pi_i}\right) \geq 0
\end{aligned}
$$

Proof of Lyapunov Lemma

$$\ell_F(R_F(\boldsymbol{\pi})) - \ell_F(\boldsymbol{\pi}) = \frac{1}{N}\sum_{n=1}^{N} \log\left\{\frac{\sum_{i=1}^{K} \pi_i F_i^n \frac{\partial \ell}{\partial \pi_i}}{\sum_{k=1}^{K} \pi_k F_k^n}\right\}$$

$$\geq \sum_{n=1}^{N}\sum_{i=1}^{K} \frac{1}{N}\frac{\pi_i F_i^n}{\sum_{k=1}^{K} \pi_k f_{kn}} \log\left(\frac{\partial \ell}{\partial \pi_i}\right)$$

$$= \sum_{i=1}^{K}\sum_{n=1}^{N} \frac{1}{N}\frac{\pi_i F_i^n}{\sum_{k=1}^{K} \pi_k f_{kn}} \log\left(\frac{\partial \ell}{\partial \pi_i}\right)$$

$$= \sum_{i=1}^{K} r_i(\boldsymbol{\pi}) \log\left(\frac{r_i(\boldsymbol{\pi})}{\pi_i}\right) \geq 0$$

Confusion for GMM data

30.253$\pm$.001	0.0	.027$\pm$.001	0.0	0.0
1.680$\pm$.000	0.0	0.0	0.0	0.0
.328$\pm$.004	0.0	32.206$\pm$.008	0.0	.706$\pm$.006
0.0	0.0	.033$\pm$.001	0.0	3.207$\pm$.001
0.0	0.0	.021$\pm$.001	0.0	31.539$\pm$.001

(a) Confusion table for GMM Net #1.

Figure: The nets were tested on a set of samples drawn independently from the training set. Values are reported as percentages for clarity. Test data sample size $N = 2500$ for all runs. Error intervals are 95% confidence standard error. An entry of 0.0 indicates that all values were zero to 3 decimal places.

Confusion for GMM data

30.165±.004	.101±.004	.014±.001	0.0	0.0
1.616±.003	.063±.003	0.0	0.0	0.0
.398±.006	.114±.003	31.739±.010	.330±.008	.659±.009
0.0	0.0	.031±.001	.333±.012	2.875±.012
0.0	0.0	.012±.000	.082±.004	31.466±.004

(a) Confusion table for GMM Net #2.

Figure: The nets were tested on a set of samples drawn independently from the training set. Values are reported as percentages for clarity. Test data sample size $N = 2500$ for all runs. Error intervals are 95% confidence standard error. An entry of 0.0 indicates that all values were zero to 3 decimal places.

Confusion for GMM data

29.897±.010	.374±.010	.009±.001	.000±.001	0.0
1.273±.011	.406±.011	.001±.001	0.0	0.0
.658±.016	.595±.017	29.916±.036	1.329±.031	.743±.018
0.0	.000±.001	.013±.001	1.221±.027	2.006±.027
0.0	0.0	0.0	.340±.009	31.220±.009

(a) Confusion table for GMM Net #3.

Figure: The nets were tested on a set of samples drawn independently from the training set. Values are reported as percentages for clarity. Test data sample size $N = 2500$ for all runs. Error intervals are 95% confidence standard error. An entry of 0.0 indicates that all values were zero to 3 decimal places.

Confusion for GMM data

26.842±.035	3.438±.035	0.0	0.0	0.0
.044±.006	1.636±.006	0.0	0.0	0.0
.075±.003	1.841±.024	28.737±.037	2.540±.025	.047±.002
0.0	0.0	.027±.001	3.122±.004	.092±.004
0.0	0.0	0.0	2.463±.023	29.097±.023

(a) Confusion table for GMM Net #4.

Figure: The nets were tested on a set of samples drawn independently from the training set. Values are reported as percentages for clarity. Test data sample size $N = 2500$ for all runs. Error intervals are 95% confidence standard error. An entry of 0.0 indicates that all values were zero to 3 decimal places.

Per class precision and recall for GMM data

GMM Net 1		
Class	Precision	Recall
1	0.936	0.999
2	0.000	0.000
3	0.998	0.966
4	0.000	0.000
5	0.979	0.999

(a) Precision and Recall table for GMM Net #1.

Figure: This table shows per class precision and recall for GMM nets trained and tested on the same data as in table 5

Per class precision and recall for GMM data

GMM Net 2		
Class	Precision	Recall
1	0.935	0.997
2	0.209	0.027
3	0.998	0.953
4	0.401	0.090
5	0.898	0.997

(a) Precision and Recall table for GMM Net #2.

Figure: This table shows per class precision and recall for GMM nets trained and tested on the same data as in table 5

Per class precision and recall for GMM data

GMM Net 3		
Class	Precision	Recall
1	0.934	0.988
2	0.279	0.194
3	0.999	0.894
4	0.385	0.364
5	0.918	0.989

(a) Precision and Recall table for GMM Net #3.

Figure: This table shows per class precision and recall for GMM nets trained and tested on the same data as in table 5

Per class precision and recall for GMM data

GMM Net 4		
Class	Precision	Recall
1	0.988	0.930
2	0.289	0.882
3	0.999	0.868
4	0.380	0.969
5	0.996	0.922

(a) Precision and Recall table for GMM Net #4.

Figure: This table shows per class precision and recall for GMM nets trained and tested on the same data as in table 5