MoE Expert Models: A Comprehensive Analysis of Three Specialized Fairness-Aware Neural Networks

Fairness Machine Learning Project

October 4, 2025

1 Introduction

This document provides a detailed mathematical and architectural analysis of the three expert models implemented in the Mixture of Experts (MoE) framework for fairness-aware machine learning. The system consists of three specialized neural network experts, each designed to address different aspects of fairness in machine learning: utility optimization, result-driven fairness, and procedural fairness.

2 Architecture Overview

All three experts inherit from a common `ExpertBase` class and share a fundamental Multi-Layer Perceptron (MLP) backbone. The base architecture consists of:

- Input layer: d_{input} dimensions
- Hidden layer: $d_{hidden} = 8$ dimensions (default)
- Output layer: 2 dimensions (binary classification)
- Activation: LeakyReLU
- Dropout: 0.3 (applied to hidden layer)

The forward pass of each expert follows the pattern:

$$h = \text{MLP}_{hidden}(x) \in \mathbb{R}^{d_{hidden}} \tag{1}$$

$$\ell = \text{MLP}_{output}(h) \in \mathbb{R}^2 \tag{2}$$

$$p = \text{softmax}(\ell) \in \mathbb{R}^2 \tag{3}$$

where h represents the hidden representation, ℓ the logits, and p the output probabilities.

3 Expert 1: Utility-Focused Expert

3.1 Objective

Expert 1 is designed to maximize predictive accuracy without explicit fairness constraints. It serves as a baseline utility-focused model.

3.2 Architecture

Expert 1 inherits directly from `ExpertBase` without additional components. The loss function is the standard cross-entropy loss:

$$\mathcal{L}_{E1} = \text{CrossEntropy}(p, y) = -\sum_{i=1}^{n}\sum_{c=0}^{1} y_{i,c}\log(p_{i,c}) \tag{4}$$

where $y_{i,c}$ is the one-hot encoded true label for sample i and class c, and $p_{i,c}$ is the predicted probability.

3.3 Mathematical Formulation

The complete loss computation is:

$$\mathcal{L}_{E1} = \mathcal{L}_{CE} \tag{5}$$

$$\mathcal{L}_{CE} = -\frac{1}{n}\sum_{i=1}^{n}\log(p_{i,y_i}) \tag{6}$$

where y_i is the true class label for sample i.

4 Expert 2: Result-Driven Fairness Expert

4.1 Objective

Expert 2 addresses result-driven fairness by incorporating both representation alignment and demographic parity/equal opportunity constraints.

4.2 Architecture

Expert 2 extends `ExpertBase` with two additional hyperparameters:

- λ_{rep}: Weight for representation alignment loss
- λ_{fair}: Weight for fairness loss

4.3 Mathematical Formulation

The total loss function combines three components:

$$\mathcal{L}_{E2} = \mathcal{L}_{CE} + \lambda_{rep}\cdot\mathcal{L}_{rep} + \lambda_{fair}\cdot\mathcal{L}_{fair} \tag{7}$$

4.3.1 Representation Alignment Loss

The representation alignment loss minimizes the distance between group mean embeddings:

$$\mu_0 = \frac{1}{n_0}\sum_{i:s_i=0} h_i \tag{8}$$

$$\mu_1 = \frac{1}{n_1}\sum_{i:s_i=1} h_i \tag{9}$$

$$\mathcal{L}_{rep} = ||\mu_0 - \mu_1||_2^2 \tag{10}$$

where $s_i \in \{0, 1\}$ is the sensitive attribute for sample i, and n_0, n_1 are the counts of samples in each group.

4.3.2 Fairness Loss

The fairness loss combines demographic parity (DP) and equal opportunity (EO) gaps:

$$\mathcal{L}_{fair} = \frac{\mathcal{L}_{DP} + \mathcal{L}_{EO}}{2} \tag{11}$$

Demographic Parity Gap:

$$p_0 = \frac{1}{n_0} \sum_{i:s_i=0} p_{i,1} \tag{12}$$

$$p_1 = \frac{1}{n_1} \sum_{i:s_i=1} p_{i,1} \tag{13}$$

$$\mathcal{L}_{DP} = |p_0 - p_1| \tag{14}$$

Equal Opportunity Gap:

$$p_0^{pos} = \frac{1}{n_0^{pos}} \sum_{i:s_i=0,y_i=1} p_{i,1} \tag{15}$$

$$p_1^{pos} = \frac{1}{n_1^{pos}} \sum_{i:s_i=1,y_i=1} p_{i,1} \tag{16}$$

$$\mathcal{L}_{EO} = |p_0^{pos} - p_1^{pos}| \tag{17}$$

where $p_{i,1}$ is the probability of positive class for sample i, and n_0^{pos}, n_1^{pos} are the counts of positive samples in each group.

5 Expert 3: Procedural Fairness Expert

5.1 Objective

Expert 3 addresses procedural fairness through attention alignment and adversarial debiasing techniques.

5.2 Architecture

Expert 3 extends `ExpertBase` with:

- $\lambda_{attention}$: Weight for attention fairness loss
- λ_{adv}: Weight for adversarial debiasing loss
- Gradient Reversal Layer (GRL) for adversarial training
- Two adversarial networks: one for hidden representations, one for logits

5.3 Mathematical Formulation

The total loss function is:

$$\mathcal{L}_{E3} = \mathcal{L}_{CE} + \lambda_{attention} \cdot \mathcal{L}_{attention} + \lambda_{adv} \cdot \mathcal{L}_{adv} \tag{18}$$

5.3.1 Attention Fairness Loss

The attention fairness loss uses Jensen-Shannon Divergence (JSD) to measure the difference in attention distributions between groups:

$$\mathcal{L}_{attention} = \text{JSD}(A_0, A_1) \tag{19}$$

$$\text{JSD}(A_0, A_1) = \frac{1}{2}\text{KL}(A_0||M) + \frac{1}{2}\text{KL}(A_1||M) \tag{20}$$

where $M = \frac{1}{2}(A_0 + A_1)$ is the average distribution, and A_0, A_1 are the mean attention weights for each group.

The attention weights are extracted using interpretability methods (SHAP, Integrated Gradients, LIME, or Gradient SHAP) and normalized:

$$A_{i,j} = \frac{|\text{attribution}_{i,j}|}{\sum_{k=1}^{d} |\text{attribution}_{i,k}| + \epsilon} \tag{21}$$

5.3.2 Adversarial Debiasing Loss

The adversarial loss prevents the model from learning sensitive attribute information:

$$\mathcal{L}_{adv} = \frac{\mathcal{L}_{adv}^{hidden} + \mathcal{L}_{adv}^{logits}}{2} \tag{22}$$

Hidden Representation Adversarial Loss:

$$h_{adv} = \text{GRL}(h) \tag{23}$$

$$\ell_{adv}^{hidden} = \text{MLP}_{adv}^{hidden}(h_{adv}) \tag{24}$$

$$\mathcal{L}_{adv}^{hidden} = \text{CrossEntropy}(\ell_{adv}^{hidden}, s) \tag{25}$$

Logits Adversarial Loss:

$$p_{adv} = \text{GRL}(p) \tag{26}$$

$$\ell_{adv}^{logits} = \text{MLP}_{adv}^{logits}(p_{adv}) \tag{27}$$

$$\mathcal{L}_{adv}^{logits} = \text{CrossEntropy}(\ell_{adv}^{logits}, s) \tag{28}$$

5.3.3 Gradient Reversal Layer

The GRL implements adversarial training by reversing gradients during backpropagation:

$$\text{GRL}(x) = x \quad \text{(forward pass)} \tag{29}$$

$$\frac{\partial \text{GRL}(x)}{\partial x} = -\lambda \quad \text{(backward pass)} \tag{30}$$

6 Implementation Details

6.1 MLP Backbone

The shared MLP backbone uses the following architecture:

- Linear layer: $d_{input} \to d_{hidden}$
- LeakyReLU activation
- Dropout (0.3)
- Linear layer: $d_{hidden} \to 2$

6.2 Adversarial Networks

Expert 3 includes two adversarial networks:

Hidden Adversarial Network:

- Input: d_{hidden} dimensions
- Hidden: $\max(4, d_{hidden}/2)$ dimensions
- Output: 2 dimensions (sensitive attribute prediction)

Logits Adversarial Network:

- Input: 2 dimensions (logits)
- Hidden: 8 dimensions
- Output: 2 dimensions (sensitive attribute prediction)

7 Training and Optimization

Each expert is trained independently with its specialized loss function. The training process involves:

1. Forward pass through the MLP backbone
2. Computation of specialized loss components
3. Backpropagation with appropriate gradient modifications (e.g., GRL for Expert 3)
4. Parameter updates using standard optimization algorithms

8 Gating Network

8.1 Objective

The gating network serves as the decision-making component that dynamically selects which expert(s) to use for each input sample. It learns to route inputs to the most appropriate expert based on the current state and expert predictions.

8.2 Architecture

The gating network is a two-layer neural network with the following structure:

- Input layer: $d_{input} + 3 \times d_{classes} + 3 + 1$ dimensions
- Hidden layer: $d_{hidden} = 16$ dimensions (default)
- Output layer: 3 dimensions (one for each expert)
- Activation: ReLU
- Temperature parameter: $\tau = 1.0$ for softmax scaling

8.3 State Representation

The gating network receives a rich state representation that includes:

$$\text{state} = [x, p_1, p_2, p_3, \text{conf}_1, \text{conf}_2, \text{conf}_3, \text{disagree}] \tag{31}$$

where:

- $x \in \mathbb{R}^{d_{input}}$: Original input features
- $p_1, p_2, p_3 \in \mathbb{R}^{d_{classes}}$: Expert probability predictions
- $\text{conf}_1, \text{conf}_2, \text{conf}_3 \in \mathbb{R}$: Expert confidence scores (max probability)
- $\text{disagree} \in \mathbb{R}$: Average pairwise disagreement between experts

8.4 Mathematical Formulation

8.4.1 Forward Pass

The gating network computes expert selection probabilities as:

$$\ell_{gate} = \text{MLP}_{gate}(\text{state}) \in \mathbb{R}^3 \tag{32}$$

$$g = \text{softmax}(\ell_{gate}/\tau) \in \mathbb{R}^3 \tag{33}$$

where g_i represents the probability of selecting expert i.

8.4.2 Expert Disagreement

The disagreement metric measures how much experts differ in their predictions:

$$d_{ij} = \frac{1}{d_{classes}} \sum_{k=1}^{d_{classes}} |p_{i,k} - p_{j,k}| \tag{34}$$

$$\text{disagree} = \frac{d_{12} + d_{13} + d_{23}}{3} \tag{35}$$

8.4.3 Confidence Scores

Expert confidence is measured as the maximum probability across classes:

$$\text{conf}_i = \max_k p_{i,k} \tag{36}$$

8.5 Training Algorithm

The gating network is trained using a reinforcement learning approach with the following components:

8.5.1 Reward Function

The reward function balances utility improvement and fairness improvement:

$$R = (U_2 - U_1) - (F_2 - F_1) \tag{37}$$

where:

- U_1, F_1: Baseline utility and fairness from Expert 1
- U_2, F_2: Utility and fairness from the gating network's mixture

8.5.2 Policy Loss

The policy loss uses REINFORCE with baseline subtraction:

$$\mathcal{L}_{policy} = -\frac{1}{n} \sum_{i=1}^{n} w_i \log g_{i,a_i} \cdot (R - b) \tag{38}$$

$$w_i = \frac{\text{disagree}_i}{\text{disagree}_{mean} + \epsilon} \tag{39}$$

where a_i is the selected expert for sample i, b is the moving average baseline, and w_i is a disagreement-based weighting factor.

8.5.3 Entropy Regularization

To encourage exploration, entropy regularization is added:

$$\mathcal{H} = -\sum_{i=1}^{3} g_i \log g_i \tag{40}$$

8.5.4 Load Balancing

Load balancing prevents the gating network from always selecting the same expert:

$$q = \frac{\text{usage}_{ma} + \epsilon}{\sum_{j=1}^{3} (\text{usage}_{ma,j} + \epsilon)} \tag{41}$$

$$\mathcal{L}_{LB} = \text{KL}(q || \text{uniform}) = \sum_{i=1}^{3} q_i \log \frac{q_i}{1/3} \tag{42}$$

where usage_{ma} is a moving average of expert usage frequencies.

8.5.5 Total Loss

The complete gating network loss is:

$$\mathcal{L}_{gate} = \mathcal{L}_{policy} - \lambda_{entropy}\mathcal{H} + \lambda_{LB}\mathcal{L}_{LB} \tag{43}$$

8.6 Expert Mixture

The final prediction combines expert outputs using gating probabilities:

$$y_{final} = \sum_{i=1}^{3} g_i \cdot p_i \tag{44}$$

8.7 Usage Tracking

The gating network maintains a moving average of expert usage:

$$\text{usage}_{ma} = \alpha \cdot \text{usage}_{ma} + (1 - \alpha) \cdot \text{batch\_usage} \tag{45}$$

$$\text{batch\_usage}_i = \frac{1}{n} \sum_{j=1}^{n} g_{j,i} \tag{46}$$

where $\alpha = 0.9$ is the momentum parameter.

9 Training Pipeline

9.1 Two-Phase Training

The MoE system uses a two-phase training approach:

9.1.1 Phase 1: Expert Pretraining

1. Train all three experts independently for $\frac{\text{epochs}}{2}$ iterations
2. During warmup (first half), only cross-entropy loss is used
3. During second half, full fairness losses are applied
4. Experts are cached for reuse

9.1.2 Phase 2: Gating Network Training

1. Experts are frozen in evaluation mode
2. Gating network is trained for $\frac{\text{epochs}}{2}$ iterations
3. Uses REINFORCE with reward-based learning
4. Periodic expert fine-tuning every 100 epochs

9.2 Evaluation Metrics

The system evaluates performance using three main categories:

9.2.1 Utility Metrics

- Accuracy (ACC)
- F1-Score (F1)
- Area Under ROC Curve (AUC)
- Combined utility: $U = \frac{\text{ACC+F1+AUC}}{3}$

9.2.2 Result Fairness Metrics

- Demographic Parity (DP): $|\mathbb{P}(\hat{Y} = 1|S = 0) - \mathbb{P}(\hat{Y} = 1|S = 1)|$
- Equal Opportunity (EO): $|\mathbb{P}(\hat{Y} = 1|Y = 1, S = 0) - \mathbb{P}(\hat{Y} = 1|Y = 1, S = 1)|$
- Combined result fairness: $F_{res} = \frac{\text{DP+EO}}{2}$

9.2.3 Procedural Fairness Metrics

- Representation Fairness (REF)
- Value Fairness (VEF)
- Attention Jensen-Shannon Divergence (ATT_JSD)
- Combined procedural fairness: $F_{proc} = \frac{\text{REF+VEF+ATT\_JSD}}{3}$

9.2.4 Final Score

The overall performance score balances utility and fairness:

$$\text{Final Score} = U - F_{res} - F_{proc} \tag{47}$$

10 Conclusion

The three expert models provide complementary approaches to fairness in machine learning:

- **Expert 1**: Pure utility optimization without fairness constraints
- **Expert 2**: Result-driven fairness through representation alignment and statistical parity
- **Expert 3**: Procedural fairness through attention alignment and adversarial debiasing

The gating network intelligently combines these experts using a reinforcement learning approach, learning to route inputs to the most appropriate expert based on the current context and expert predictions. This multi-expert framework allows for flexible combination of different fairness approaches, enabling the system to adapt to various fairness requirements and trade-offs between accuracy and fairness.

The complete system demonstrates how different fairness paradigms can be integrated into a unified framework, providing both theoretical rigor and practical effectiveness in addressing fairness challenges in machine learning.