STOR 390 Final Project: Balancing the Scales: A Philosophical Approach to Advancing Fairness in Imbalanced Data

Tianrui Ye

2024-05-03

Introduction

With the wide application of machine learning, the fairness of machine learning models has received increasing attention. A fair model should be able to give unbiased predictions for different groups of people without discriminating against individuals based on their sensitive attributes such as gender and race. Much existing research has been devoted to the development of machine learning algorithms that strike a balance between accuracy and fairness.

However, we note that most of the existing fairness algorithms have been tested and developed on relatively balanced datasets. In the real world, training data is often highly unbalanced, with the minority group having a much smaller amount of data than the mainstream group. In such extremely unbalanced situations, minorities are likely to be underrepresented, leading to bias and discrimination against them in the trained models. Unfortunately, existing fairness algorithms do not address this problem well.

A recent work, FPFL , proposes a federated learning framework to train both privacy-preserving and fair models in distributed environments. FPFL adopts a two-stage training process, in which each participant locally trains a model with fairness constrained loss, and then aggregates the models using a differential privacy method. Then the models are aggregated by a differential privacy method. This decoupling approach can alleviate the fairness problem caused by extreme data imbalance to some extent. However, we find that the constraint used to guarantee fairness in the first stage does not have enough robustness to the extreme unbalanced data, and through experiments on some highly unbalanced datasets, we find that although the overall fairness of the model obtained by FPFL is improved, the bias to minorities is still very large, and even the The model refuses to give predictive judgment to minorities.

The purpose of this paper is to deeply analyze the challenges faced by fairness algorithms in extremely unbalanced data environments, and try to propose some solutions. We believe that in order to design fairness machine learning algorithms that can deal with extreme data imbalance, it is necessary to firstly consider the problem from the data and optimization perspectives, such as assigning more weights to minorities and paying more attention to minorities when calculating fairness metrics. Secondly, more robust optimization algorithms need to be developed to avoid the failure of the fairness constraint when the data are extremely skewed. Finally, when evaluating the fairness of the model, it is also necessary to pay more attention to the performance of the model on minorities and establish more comprehensive and detailed evaluation indexes.

Translated with www.DeepL.com/Translator (free version)

Analysis of Methods

Methodolody

According to the paper, the loss function L_1 used in Phase 1 consists of two parts: the regular cross-entropy loss l_{CE} and the fairness loss l_k. These two parts are combined using a Lagrange multiplier λ. The cross-entropy loss l_{CE} is defined as:

$$l_{CE}(h_{\phi_i}, X, Y) = \mathbb{E}_{(x,y)\sim(X,Y)}\left[-y_i \log(h\phi_i(x)) - (1-y)\log(1-h_{\phi_i}(x))\right]$$

where h_{ϕ_i} denotes the model trained by the i-th agent with parameters ϕ_i. The fairness loss l_k depends on the chosen fairness metric. The paper considers two fairness metrics: demographic parity and equalized odds. If demographic parity is used, l_k is defined as:

$$l_{DemP}(h_{\phi_i}, X, A) = |\mathbb{E}[h_{\phi_i}(x)|A=a] - \mathbb{E}[h_{\phi_i}(x)]|$$

If equalized odds is used, l_k is defined as:

$$l_{EO}(h_{\phi_i}, X, A, Y) = |\mathbb{E}[h_{\phi_i}(x)|A=a, y] - \mathbb{E}[h_{\phi_i}(x)|y]|$$

Therefore, the overall loss function L_1 in Phase 1 is:

$$L_1(h_{\phi_i}, X, A, Y) = l_{CE} + \lambda l_k, \quad k \in DemP, EO$$

The goal of the model is to minimize L_1, i.e.:

$$\min_{\phi} \max_{\lambda} L_1(\cdot)$$

This is the specific form of the loss function used in Phase 1 of the paper. The introduction of the fairness loss l_k ensures that the model strives to achieve fairness while pursuing accuracy, resulting in a model that balances the two objectives.

Reproduction of the results

The original algorithm was implemented in a Python environment, and the source code is publicly available on GitHub. However, due to the limited extensibility of R compared to Python, certain methods and specific logic cannot be fully reproduced in R. Additionally, the dataset used in the original paper was not made publicly available. Therefore, in the following analysis and argumentation, I will use simulated data.

I simulated both normal and imbalanced datasets to evaluate the algorithm's performance. The following is a detailed description of the generated datasets:

Normal Data: I simulated 20 normal variables, each following a normal distribution $N(0, 1)$. There are also two protected attributes, `prot_attr1` and `prot_attr2`, each taking values of 0 and 1, simulating sensitive attributes such as gender. The proportion of 0 and 1 for each protected attribute is 0.5. Finally, there are two dependent variables, `y0` and `y1`, representing negative and positive labels, respectively. The proportion of positive and negative classes is also 0.5. I generated a total of 1,500 rows for the training set and 400 rows for the test set. The first five rows of the training set are shown below.

```
##            V1         V2         V3         V4         V5         V6
## 1 -0.05638101 -1.5703349  2.6492686  1.2536632  1.3188571  1.2342645
## 2  1.36159940  1.4318624  1.2074195 -0.9568123 -1.9001631  0.1977326
## 3 -0.64544797  1.8013072  0.1532414 -1.0341349  1.5682651 -0.3806339
## 4  1.27409368 -2.2614696 -1.3277937  0.3481366  0.4699932 -1.5020123
## 5  0.52938722  0.1104926  0.7392958  2.1852818 -0.7147473 -0.4343615
## 6 -1.95089193 -0.5738134  1.2779257 -0.3630975 -0.8893561  1.6647014
##            V7         V8         V9        V10         V11        V12
## 1 -0.79723331 -0.2622925 -0.7319781  0.3093455  0.42480724  0.4974768
## 2  0.07915213  0.2407643  0.7237776 -0.6014077 -0.06312186  0.1556090
## 3  0.20619544 -0.3333122 -0.3128863  0.7563316 -0.59017539  0.2580254
## 4 -2.34957803  0.2279061 -0.2229760 -1.1096897 -1.34732776 -1.7031347
## 5  0.27381932 -1.2933390  0.2239728 -0.8299212 -0.99645451  0.6278548
## 6  0.01662230 -2.1529426  1.5067097  2.1747693  1.38615960  0.6169005
##           V13        V14        V15        V16        V17         V18        V19
## 1 -0.2265761 -1.5873820 -0.2256479 -0.8320309 -0.5201121  1.67760164 0.55191911
## 2  1.0521911 -0.3206948 -0.4508707  1.4527417  0.4551920 -1.65926068 0.06277017
## 3  0.2761261  1.2659089 -0.3783535  0.3260779  0.4224935 -0.04113885 0.61872142
## 4 -0.5720122  0.4996900 -0.2859831  0.8236425 -1.1074068 -1.37808816 1.85494920
## 5  0.6641331 -2.2534067  0.9940283  0.6224831  0.2586428 -0.63225123 0.22855142
## 6  0.5398842 -0.5844866 -0.2602651 -0.6272756 -0.2494152  1.85370024 0.07344137
##           V20 y0 y1 prot_attr1 prot_attr2
## 1 -1.6290361  1  0          0          1
## 2 -0.4877941  0  1          0          0
## 3  0.2620492  1  0          1          1
## 4  2.7168748  0  1          1          0
## 5  1.6054364  0  1          0          1
## 6 -1.4649387  1  0          0          1
```

Imbalanced Data: I also simulated 20 normal variables, each following a normal distribution $N(0, 1)$. I added two extreme variables: `rbinom(n, 1, 0.01) * rnorm(n, mean = 100, sd = 25)` and `rbinom(n, 1, 0.99) * rnorm(n, mean = -100, sd = 25)`. The former is mostly 0, with approximately 1% of the values following $N(100, 25)$. The latter has approximately 1% probability of being 0, with about 99% of the values following $N(-100, 25)$. These two columns are used to simulate the occurrence of extreme or rare events [1]. I also created two protected attributes, simulating extremely imbalanced situations. In `prot_attr1`, 95% of the data is 0, and 5% of the data is 1. In `prot_attr2`, 10% of the data is 0, and 90% of the data is 1. The dependent variables `y0` and `y1` were also created with imbalanced attributes. In the current dataset, 10% of the data in `y1` is 1, and correspondingly, 10% of the data in `y0` is 0. I similarly simulated 1,500 rows for the training set and 400 rows for the test set. The first five rows of the training set are shown below.

```
##           V1          V2          V3         V4          V5         V6
## 1  0.2972714  0.95392350 -0.01096755  1.2290392 -1.37901928 -0.6220680
## 2 -2.7525572  0.01657056 -0.05974235 -1.4331663  0.60805995 -1.9137246
## 3  1.2274787  0.88116840 -2.21766632  0.9812427  0.43150728  1.2127641
## 4  0.2155626  0.57132273 -1.35085215 -0.8268766 -0.09809138  1.1339323
## 5  0.1898034 -2.06536302  0.10091068  0.1871031  0.02909244  2.2729207
## 6  1.2267621 -0.91829975  0.06720970  1.6350131  1.00406900  0.9069925
##             V7         V8         V9         V10         V11         V12
## 1 -0.007451788 -1.9083017  0.9332323  0.5233308 -0.3693568 -1.0482217
## 2  1.598455640 -1.0237727 -0.5266425 -1.5834954 -0.2238844 -0.7853943
## 3 -0.023270006  0.3951418  1.0919152  0.6280403 -0.1183283 -2.1838339
## 4 -1.355309983 -1.3197462 -1.1103685  0.6243135  0.5522630  1.1327513
## 5  0.496950593 -0.6317731  1.4124645 -1.3652616 -1.5929399  0.4996753
## 6 -2.483980853 -0.1254277 -0.8010001  0.7611317 -0.4685315  1.1349301
##          V13         V14         V15         V16         V17         V18
## 1 -0.6990145  0.89003993  0.17111647 -0.1544404  0.5020350 -1.0347070
## 2  0.3271934 -1.10859304  0.94006023 -1.2930744 -0.7530842 -0.1879283
## 3 -1.4334262 -1.33428489  1.40688127  1.0103506  0.6496366 -1.1589409
## 4 -0.4552427  0.71647306 -0.07902413  0.9103094 -0.9512831  0.1861893
## 5 -0.3015384  0.03121414 -0.53985378 -0.5330411  0.5645748  0.5042631
## 6  1.1426661 -0.52998483 -0.68266001 -0.5993410 -1.8410416  0.6368495
##           V19        V20 extreme_feature1 extreme_feature2 y0 y1 prot_attr1
## 1 -0.15053127  0.8279983                0        -87.02014  1  0          0
## 2  0.62015379  1.8888499                0        -68.23114  0  1          0
## 3  1.14901173 -0.2023877                0       -135.13288  0  1          0
## 4 -0.31626568  0.8765159                0        -85.57520  1  0          0
## 5 -0.26999017  0.1698082                0       -112.12617  1  0          0
## 6 -0.06841431  0.7055777                0       -140.53114  1  0          0
##   prot_attr2
## 1          1
## 2          1
## 3          1
## 4          1
## 5          1
## 6          1
```

```
## # A tibble: 2 × 5
##   Data_Type       Accuracy  DemP equal_opportunity `F1 equalized_odds`
##   <chr>              <dbl> <dbl>             <dbl>               <dbl>
## 1 Normal_Data         0.8   0.18              0.15                0.14
## 2 Imbalanced_Data     0.87  0.23              0.83                0.61
```

From the table, it is evident that all fairness metrics for the Imbalanced Data are significantly higher compared to the Normal Data, indicating that the Fair-SGD algorithm presented in the current literature is no longer fair when faced with Imbalanced Data. Furthermore, there is a severe issue within the Imbalanced Data, as shown in the confusion matrix printed below. It can be observed that the model predicts all instances as Class 1, suggesting that the model has not learned the features of Class 2 effectively.

```
## 13/13 - 0s - 53ms/epoch - 4ms/step
```

```
##       Predicted
## Actual   1   2
##      1 365   1
##      2  34   0
```

The poor performance of the Fair-SGD algorithm on the Imbalanced Data highlights the limitations of current fairness-aware machine learning methods when dealing with class imbalance. The algorithm's inability to learn the features of the minority class (Class 2) leads to a biased model that favors the majority class (Class 1), resulting in a high number of false negatives.

This issue is particularly concerning in the context of fairness, as the model's bias towards the majority class can lead to discriminatory outcomes for the minority class [3]. In many real-world applications, such as credit scoring or job hiring, the minority class may represent a protected group, and the model's inability to fairly assess their instances can perpetuate societal biases and result in unfair treatment [4].

The significant increase in fairness metrics for the Imbalanced Data further emphasizes the need for developing fairness-aware machine learning methods that are robust to class imbalance [5]. These methods should be able to effectively learn the features of both majority and minority classes while maintaining fairness across different subgroups.

Based on the above two points and analysis, it is clear that the current Fair-SGD model still has significant issues. My next goal is to modify and adjust the current Loss Function and model structure to address the problems of model prediction bias and fairness.

First, the issue of prediction bias needs to be addressed. This problem can be solved by resampling techniques. Here, I attempt to use the SMOTE (Synthetic Minority Over-sampling Technique) sampling method.

SMOTE (Synthetic Minority Over-sampling Technique) is a resampling method used to address imbalanced datasets. It increases the representation of the minority class by generating synthetic samples, thereby improving the classifier's sensitivity towards the minority class.

Mathematical Formulation of Custom Loss Function

The new formula I have established focuses on balancing all fairness metrics, rather than solely focusing on Demographic Parity as in the original paper. More importantly, it aims to achieve better performance on imbalanced data. I hope that the algorithm can perform well on all fairness metrics. The following is the specific formula I have constructed:

The proposed loss function consists of two main components: the classification loss and the fairness loss. The classification loss is calculated using the categorical cross-entropy, which measures the model's performance on the classification task. The fairness loss, on the other hand, is designed to minimize the differences in various fairness metrics between different subgroups.

True Positive Rate (TPR) Difference

$$\text{TPR\_diff} = |\text{TPR}_{g1} - \text{TPR}_{g2}|$$

$$\text{TPR}_g = \frac{\sum(\text{pred\_positive} \cdot \text{group\_positive})}{\sum \text{group\_positive} + \epsilon}$$

pred_positive is a mask of the predictions that are positive. group_positive is a mask of the actual positives for each protected group. #### False Positive Rate (FPR) Difference

$$\text{FPR\_diff} = |\text{FPR}g1 - \text{FPR}_{g2}|$$

$$\text{FPR}_g = \frac{\sum(\text{pred\_negative} \cdot \text{group\_negative})}{\sum \text{group\_negative} + \epsilon}$$

pred_negative is a mask of the predictions that are negative. group_negative is a mask of the actual negatives for each protected group.

Demographic Parity (DP) Difference

$$\text{DP\_diff} = |\frac{\sum(\text{pred\_positive} \cdot \text{prot\_data}_{g1})}{\sum \text{prot\_data}_{g1} + \epsilon} - \frac{\sum(\text{pred\_positive} \cdot \text{prot\_data}_{g2})}{\sum \text{prot\_data}_{g2} + \epsilon}|$$

prot_data_g refers to the mask for each protected group.

Combined Loss

The total loss is a weighted sum of the classification loss and the fairness loss, with a hyperparameter ρ controlling the balance between the two objectives. This allows for a trade-off between model accuracy and fairness, which can be adjusted based on the specific requirements of the application.

$$L = L_{CE} + \rho \cdot (\text{TPR\_diff} + \text{FPR\_diff} + \text{DP\_diff})$$

ρ is a tunable parameter that controls the contribution of the fairness component to the total loss. ϵ is a small constant added for numerical stability to avoid division by zero.

The inclusion of multiple fairness metrics in the loss function aims to address the limitations of focusing on a single metric, such as Demographic Parity. By considering TPR, FPR, and predicted positive rates, the proposed approach provides a more comprehensive assessment of fairness and is better suited for imbalanced data.

Updated Neural Network Structure

In addition to addressing the prediction bias issue through resampling techniques like SMOTE and redesign loss function, I also propose modifications to the neural network structure to better handle imbalanced data and improve fairness.

Input Layer:

Dense Layer: The first layer is a dense (fully connected) layer with 64 units. It uses the ReLU (Rectified Linear Unit) activation function. This layer accepts input with a shape of 22, which means it expects data with 22 features.

Normalization:

Batch Normalization: Following the first dense layer, a batch normalization layer is applied. This layer normalizes the activations of the previous layer at each batch, i.e., it applies a transformation that maintains the mean activation close to 0 and the activation standard deviation close to 1.

Hidden Layers:

Dense Layer: Another dense layer with 64 units follows, also using the ReLU activation function.

Batch Normalization: This is again followed by a batch normalization layer to help in stabilizing the learning process by normalizing the inputs of the subsequent layer.

Dense Layer: Next, there is an additional dense layer with 32 units, employing the ReLU activation function. This layer is meant to further process the features extracted by previous layers before the final output.

Output Layer:

Dense Layer: The final layer is a dense layer with 2 units, using the sigmoid activation function. This setup suggests that the model is intended for binary classification, as the sigmoid function will output a probability distribution over two classes.

New results in Imbanlanced Data

```
## # A tibble: 2 × 5
##   Data_Type              Accuracy  DemP equal_opportunity Equalized_odds
##   <chr>                     <dbl> <dbl>             <dbl>          <dbl>
## 1 Methodology_From_Paper     0.87  0.23              0.83           0.61
## 2 New_Methodolody            0.9   0.11              0.14           0.28
```

The new results demonstrate significantly improved fairness on the Imbalanced Data, greatly enhancing the performance of the original method. Through oversampling, the new approach also mitigates the issue of uneven data distribution to some extent. While increasing accuracy, it avoids the overfitting situation where all predictions belong to the same label, as observed previously. Therefore, the modifications and adjustments I made to the method have optimized the current model's issues with specific data and distributions, making the model fairer from certain perspectives.

The use of SMOTE for oversampling helps in addressing the class imbalance problem by generating synthetic samples for the minority class [1]. This ensures that the model receives a more balanced representation of both classes during training, reducing the bias towards the majority class. The improved distribution of samples leads to better generalization and prevents the model from overfitting to a single label.

Moreover, the incorporation of the custom loss function, which considers multiple fairness metrics, plays a crucial role in promoting fairness across different subgroups. By minimizing the differences in true positive rates, false positive rates, and predicted positive rates between the subgroups, the model learns to treat them more equitably. This multi-faceted approach to fairness helps in addressing various aspects of discrimination and ensures that the model's performance is consistent across different segments of the population.

In conclusion, the combination of SMOTE oversampling, custom loss function, and updated neural network structure has successfully addressed the limitations of the original method when dealing with imbalanced data. The new approach not only improves accuracy but also ensures fairness across different subgroups. The model's performance is no longer skewed towards a particular label, and it demonstrates better generalization capabilities.

Analysis of Normative Consideration

Justice and Equality of Opportunity:

The principles of justice and equality of opportunity are closely intertwined in the context of algorithmic fairness. A just society is one in which individuals have equal opportunities to pursue their life goals, regardless of their social background or circumstances. This idea is echoed in the concept of equality of opportunity, which emphasizes that individuals should be judged based on their merits and efforts, rather than arbitrary factors like race or gende.

When machine learning models are trained on imbalanced data, they risk violating these principles. If a particular group is underrepresented in the training data, the model may not adequately capture their unique characteristics and merits, leading to biased predictions that limit their opportunities. This is particularly concerning in domains like education, employment, or criminal justice, where algorithmic decisions can have far-reaching consequences for individuals' life prospects.

For example, consider a scenario where a machine learning model is used to screen job applicants. If the training data is imbalanced, with a minority group underrepresented, the model may learn to prioritize features that are more prevalent in the majority group, even if these features are not directly relevant to job performance. As a result, qualified candidates from the minority group may be unfairly disadvantaged, undermining the principles of justice and equality of opportunity.

Respect for Autonomy:

When machine learning models make biased decisions based on imbalanced data, they can undermine individuals' autonomy in several ways. First, if the model's predictions are used to allocate resources or opportunities, biased decisions can limit individuals' ability to pursue their chosen life paths. For example, if a model used in college admissions is biased against a particular group, it may unfairly restrict their access to higher education, limiting their autonomy in shaping their educational and career trajectories.

Second, biased models can perpetuate stereotypes and stigmatization, which can constrain individuals' autonomy by shaping how others perceive and interact with them. If a model used in hiring decisions consistently undervalues the merits of a particular group, it can reinforce negative stereotypes and lead to discrimination, even outside the specific context of the algorithmic decision.

To respect individual autonomy, it is essential to develop fairness algorithms that not only mitigate bias but also provide meaningful transparency and explanations for their decisions. By making the decision-making process more interpretable and accountable, individuals can better understand how their personal characteristics and merits are being evaluated, and challenge decisions that they believe are unfair or discriminatory.

Moreover, respecting autonomy also requires involving affected communities in the development and deployment of these algorithms. By engaging in participatory design processes and soliciting feedback from diverse stakeholders, developers can ensure that the models reflect the values and priorities of the communities they serve, rather than imposing externally defined notions of fairness.

In conclusion, the principles of justice, equality of opportunity, and respect for autonomy provide a powerful lens through which to examine the ethical implications of fairness in imbalanced data. By recognizing the ways in which biased algorithms can undermine these fundamental values, we can work towards developing more equitable and inclusive machine learning practices. This requires not only technical innovations in fairness algorithms but also a commitment to engaging with affected communities and grappling with the broader social and political contexts in which these technologies are deployed. Only by centering these philosophical principles can we ensure that machine learning serves as a tool for promoting social justice and human flourishing.

Conclusion

In this paper, we have addressed the limitations of current fairness algorithms when applied to imbalanced datasets, as highlighted in the work "FPFL: Fair and Private Federated Learning" by Padala et al. (2021). By incorporating philosophical principles of justice, equality of opportunity, and respect for autonomy, we have proposed a novel approach that includes modifications such as SMOTE oversampling, custom loss functions, and updated neural network architectures. Our research underscores the importance of interdisciplinary collaboration and the need to consider the ethical dimensions of algorithmic fairness in the development and deployment of machine learning systems.

The impact of our work extends beyond the specific findings of the cited paper, providing a foundation for future research and practical applications that prioritize fairness and ethical considerations in imbalanced data scenarios. By addressing these challenges, we aim to promote the development of more just, equitable, and inclusive AI systems that respect individual rights and foster positive social outcomes.