
Deep Fair Discriminative Clustering

— Supplementary Material —

Hongjing Zhang
 Department of Computer Science
 University of California - Davis
 hjzzhang@ucdavis.edu

Ian Davidson
 Department of Computer Science
 University of California - Davis
 davidson@cs.ucdavis.edu

A Theorems and Proofs

Lemma A.1. *The optimal balance can be achieved only when all the clusters have the same balance. Formally, $\forall i, j \in \{1, 2, \dots, K\}$: $\text{balance}(C_i) = \text{balance}(C_j)$.*

Proof. The proof is by contradiction, we assume the optimal balance can be achieved when not all clusters have the same balance. Let cluster C_i have the largest balance as b_{max} and cluster C_j has the smallest balance as b_{min} . Denote the number of T different groups' instances in C_i as $\{C_{i1}, C_{i2}, \dots, C_{iT}\}$, similarly the composition for C_j as $\{C_{j1}, C_{j2}, \dots, C_{jT}\}$. Assume the balance for C_i is achieved by $b_{max} = \frac{C_{i\alpha}}{C_{i\beta}}$, the balance for C_j is achieved by $b_{min} = \frac{C_{j\gamma}}{C_{j\theta}}$. Now we discuss two possible cases:

Case 1: If $\alpha = \gamma$ and $\beta = \theta$. Based on our definition we have $\frac{C_{i\alpha}}{C_{i\beta}} > \frac{C_{j\alpha}}{C_{j\beta}}$. Let $r = \frac{1}{2}(\frac{C_{i\alpha}}{C_{i\beta}} + \frac{C_{j\alpha}}{C_{j\beta}})$, we can move ϵ instances which belong to group α from C_i to C_j to achieve the higher balance r . This can be done by setting $\frac{C_{i\alpha}-\epsilon}{C_{i\beta}} = \frac{C_{j\alpha}+\epsilon}{C_{j\beta}} = r$.

Case 2: If $\alpha \neq \gamma$ or $\beta \neq \theta$. Based on the balance definition we have $\frac{C_{i\gamma}}{C_{i\theta}} \geq \frac{C_{i\alpha}}{C_{i\beta}}$. Based on our definition we have $\frac{C_{i\alpha}}{C_{i\beta}} > \frac{C_{j\gamma}}{C_{j\theta}}$. Thus we have $\frac{C_{i\gamma}}{C_{i\theta}} \geq \frac{C_{i\alpha}}{C_{i\beta}} > \frac{C_{j\gamma}}{C_{j\theta}}$. Similar as in case 1 we can move ϵ' instances which belong to group γ from C_i to C_j to achieve higher balance $r' = \frac{1}{2}(\frac{C_{i\gamma}}{C_{i\theta}} + \frac{C_{j\gamma}}{C_{j\theta}})$.

In both cases, we can swap some instances between clusters to increase the final balance. This contradicts our assumption and completes our proof. $\square$

Theorem A.2. *To achieve optimal balance value for multi-state protected variables, we must satisfy the condition: $P(x \in G_t | x \in C_k) = \rho_t$ which is precisely the optimal fairness value for equation¹ 2.*

Proof. Given the condition $P(x \in G_t | x \in C_k) = \rho_t$, let G_{min} be the smallest protected group and G_{max} be the largest protected group, the largest balance we can achieve is $|G_{min}|/|G_{max}|$. The proof by contradiction assumes there exists a solution where all the clusters have the same balance (based on lemma A.1) and have balance $\alpha > |G_{min}|/|G_{max}|$. For each cluster we sort each protected group based on their size in an increasing order. We use $C_{i1}^*, C_{i2}^* \dots C_{iT}^*$ to denote the size of the sorted groups in cluster i . Obviously we have $\alpha = \frac{C_{i1}^*}{C_{iT}^*}$ for any cluster i . Now we sum up the smallest group among all the K clusters as S : $S = \sum_{i=1}^K C_{i1}^*$. Similarly we can calculate the sum of the largest group among all the clusters as: $\sum_{i=1}^K C_{iT}^* = \frac{1}{\alpha} \sum_{i=1}^K C_{i1}^* = \frac{S}{\alpha}$. Consider the allocation of the smallest group G_{min} we have the following inequality: $|G_{min}| \geq \sum_{i=1}^K C_{i1}^* = S$. Similarly we

¹ All the equations referenced in supplementary material are in the main paper.

have $|G_{max}| \leq \sum_{i=1}^K C_{iT}^* = \frac{S}{\alpha}$. By combining previous two inequalities we have $\alpha|G_{max}| \leq S \leq |G_{min}|$ which means $|G_{min}|/|G_{max}| \geq \alpha$. This contradicts with our initial assumption that there exists a α which is larger than $|G_{min}|/|G_{max}|$. Thus we complete the proof. $\square$

Proof Total Unimodularity of Constraint Matrix We know that if a constraint matrix of an ILP is totally unimodular (TU) then we can solve the problem using an LP (linear program) solver and the solution will still be integral [4]. Using an LP solver will largely reduce the running time and [5] has shown that the running time for LP is polynomial in the input size.

In the above proposed constraints, there are NK unique regular variables (N instances and K categories). To construct the constraint matrix C which encodes constraint 7, constraint 8 and constraint 9, we will use $2NK$ regular variables. Matrix C has $T + 1$ rows (the first T rows correspond to the fairness constraints in equation 7 and last row corresponds to constraints in equation 9) and $N + K$ columns. Note the first K columns of the last row are set to 0 and the last n columns of first T rows are set to 0. In matrix C , each entry of C is from $\{-1, 0, 1\}$. Moreover, each column only has one non-zero element. This is because: (1) for constraints set in equation 7, each instance only belongs to one protected group, (2) for constraints set in equation 9, there is only one row vector with K elements as 1 to ensure the valid assignment.

Lemma A.3. TU Identity [4]. *Let C be a matrix such that all its entries are from $\{0, 1, -1\}$. Then C is totally unimodular, i.e., each square submatrix of C has determinant 0, 1, or -1 if every subset of rows of C can be split into two parts A and B so that the sum of the rows in A minus the sum of the rows in B produces a vector all of whose entries are from $\{0, 1, -1\}$.*

Theorem A.4. *The matrix C formed by the coefficients of the constraints used to encode our proposed constraints from equation 7, 8 and equation 9 is totally unimodular.*

Proof. We consider any subset F of rows in C . We will show that F can be partitioned into two sets A and B to satisfy the condition in lemma A.3. Our partitioning scheme is as follows: the first row of F is put into A and the remaining rows are put into B . Let row vectors SA and SB denote the sums of the rows within A and B respectively. It is clear that elements in SA and SB are from $\{1, 0, -1\}$ because each column only has one non-zero element. Now we show all the elements in $SA - SB$ are from $\{1, 0, -1\}$. Firstly there will be only one non-zero element in SA , and we denote the column which has a non-zero element as r . If column r belongs to the fairness constraints (from column 1 to column N in C), then the column r within SB will be 0 since one instance only belongs to one protected group. If the column r belongs to the valid assignment constraints (from column $N + 1$ to $N + K$ in C) then column r within SB will be 0 since those elements are filled with 0. Thus column r of $SA - SB$ is still from $\{1, 0, -1\}$. The non-zero elements in other columns will change their sign but still from $\{1, 0, -1\}$. This completes the proof. $\square$

B Limitations

Although we empirically demonstrate the promising results of our proposed approach, our work does not have a theoretical guarantee in terms of clustering performance. We think that future work on theoretical performance guarantees in deep fair clustering is an interesting and challenging direction.

C Broader Impact

Our work will be open-sourced, allowing researchers to conduct deep clustering tasks on both tabular and image datasets with group-level fairness rules. One advantage of our work is that users can define their own group-level fairness as we shown in the flexible fairness rules experiments. Another advantage is that the learned model can be used for out-of-sample predictions of new instances which may not have sensitive attributes available. However, our flexible design will also provide the opportunities for malicious users to pass the "unfair constraints" into our ILP fairness objective which lead to artificially unfair results. We recommend the users to output both the fairness requirements within the ILP objective and the clustering results to ensure the correct use of our algorithm.

D Baselines Implementations

For the k-means baseline, we choose to use the k-means++ implementation from scikit-learn library. For scalable fair clustering work [1], we use the code from their author’s github repository (https://github.com/talwagner/fair_clustering). For deep embedding clustering [7] we use the implementations from the public github repositories (<https://github.com/vlukiyanov/pt-dec>) and for IMSAT [2] we use the code from their author’s github repository (<https://github.com/weihua916/imsat>). For Wang and Davidson’s deep fair clustering work [6], we have contacted the author to get their code re-run the experiments under our setting. For the DFCV [3] approach, we directly excerpt the experimental results from their original paper.

E Hyper-parameters and Architectures of Our Approach

For the digit datasets like reverse-MNIST and MNIST-USPS, we use a 2-block convolutional architecture with first block consisting of a convolutional layer with 8 (5×5) filters and second block consisting of 4 (5×5) filters. Each block is followed with BatchNorm, ReLU activation and max pooling layers. The second block is finally connected to a fully connected layer with 10 units. For the HAR data we use a stacked fully connected neural network with intermediate layers as $d - 1200 - 1200 - k$, note $d = 561$ is the total number of features in HAR and $k = 6$ is the number of clusters. Moreover, we have listed the hyper-parameters that we used for these three data sets in Table 1. In unsupervised learning, it is not straightforward to determine hyper-parameters by cross-validation. Hence, we fixed the hyper-parameters (weight decay α and augmentation term γ) across all the datasets; we tuned the hyper-parameter β based on the training set’s balance value as mentioned in the main paper to achieve satisfying fair results.

Table 1: Deep-clustering data: hyperparameter summary

	HAR	reverse-MNIST	MNIST-USPS
Learning rate	0.0002	0.0005	0.0005
Batch size	256	256	256
Train epochs	60	80	50
Weight decay α	0.0001	0.0001	0.0001
Parameter β	4.00	6.00	4.00
Parameter γ	1.00	1.00	1.00

Table 2: Traditional tabular data: hyperparameter summary

	Census	Credit	Bank
Learning rate	0.0002	0.0002	0.0005
Batch size	256	256	256
Training epochs	30	50	30
Weight decay α	0	0.0001	0.0001
Parameter β	4.00	5.00	4.00
Parameter γ	1.00	1.00	1.00

For the traditional fair clustering data sets, we use a stacked fully connected neural network with intermediate layers as $d - 50 - 50 - k$, note the total number of features d for Census, Credit and Bank are 5, 3, 14 and $k = 2$ is the number of clusters. Moreover, we have listed the hyperparameters that we used for these three data sets in Table 2. We set the same values for hyper-parameters α and γ . Since these tabular data sets only have limited features we observe that fewer training epochs are needed to train our model.

F Extra Experimental Results

In this section we first report the run time results in Table 3. Then we show clustering visualization results for MNIST-USPS in Figure 1 and 2. Our codes will be available on GitHub for public use after the reviewing process.

Runtime Analysis

We implement our algorithm in Python 3.6 and run our experiments on a PC with a 3.3 GHz Intel Core i9 Processor and 1 GeForce RTX 2080 TI graphic card. We use Gurobi² for solving all the ILP problems. We report the runtime results in Table 3. We can observe that solving the ILP via LP is not time-consuming, the introduction of our fairness objective does add more computations compared with the non-fair version but the overall time cost is still acceptable.

Table 3: We report the average running time (second) over 10 trials.

	ILP solver	Train with fairness objective	Train w/o fairness objective
HAR	87.20	130.81	12.49
reverse-MNIST	389.71	813.26	218.90
MNIST-USPS	201.56	434.18	127.76
Census	32.65	96.13	22.54
Credit	53.82	151.13	34.28
Bank	5.12	12.49	3.57

Clustering Results Visualization

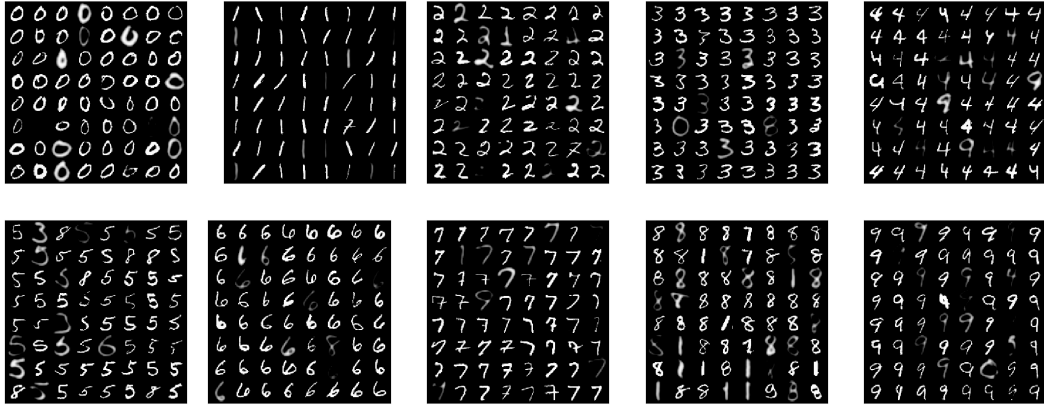


Figure 1: Sampled instances of MNIST-USPS discovered using our fair clustering approach.

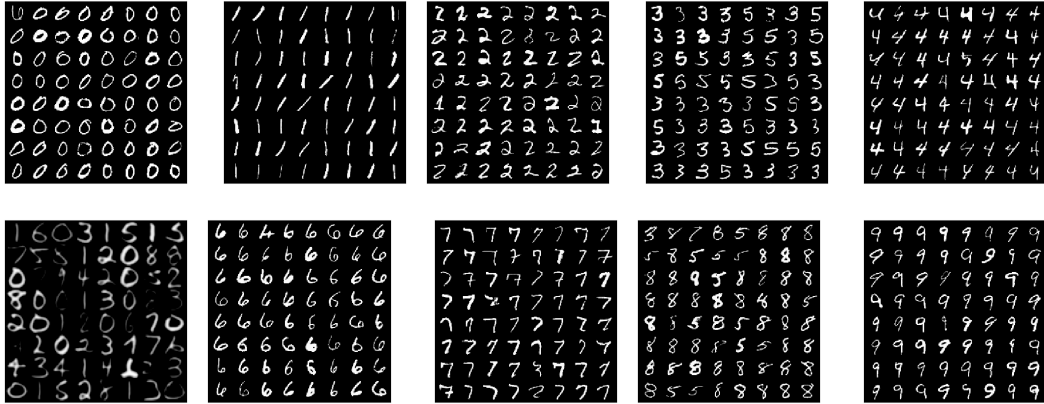


Figure 2: Sampled instances of MNIST-USPS discovered using our approach w/o fairness objective.

²<https://www.gurobi.com/>

Figure 1 and 2 shows randomly selected instances of the MNIST-USPS that clustered via our proposed fair clustering approach and non-fair version. We observe that in the non-fair version all the instances from USPS are grouped to one cluster and our approach successfully separated them into different clusters to achieve fairness goals.

References

- [1] A. Backurs, P. Indyk, K. Onak, B. Schieber, A. Vakilian, and T. Wagner. Scalable fair clustering. In *International Conference on Machine Learning*, pages 405–413, 2019.
- [2] W. Hu, T. Miyato, S. Tokui, E. Matsumoto, and M. Sugiyama. Learning discrete representations via information maximizing self-augmented training. In *International Conference on Machine Learning*, pages 1558–1567, 2017.
- [3] P. Li, H. Zhao, and H. Liu. Deep fair clustering for visual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9070–9079, 2020.
- [4] A. Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, 1998.
- [5] P. M. Vaidya. Speeding-up linear programming using fast matrix multiplication. In *30th annual symposium on foundations of computer science*, pages 332–337. IEEE, 1989.
- [6] B. Wang and I. Davidson. Towards fair deep clustering with multi-state protected variables. *arXiv preprint arXiv:1901.10053*, 2019.
- [7] J. Xie, R. Girshick, and A. Farhadi. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487, 2016.