Towards Robust and Fair Machine Learning

By

ANSHUMAN CHHABRA
DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER SCIENCE

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

Prof. Prasant Mohapatra, Chair

Prof. Xin Liu

Prof. Hao-Chuan Wang

Committee in Charge

2023

To my parents, Anjolie and Navdeep Chhabra.

Contents

List of Figures

List of Tables

Towards Robust and Fair Machine Learning

Abstract

Recent advances in Machine Learning (ML) and Deep Learning (DL) have resulted in the widespread adoption of models across various application pipelines. However, despite these performance improvements, ML/DL models have been shown to be vulnerable to adversarial inputs that can reduce functionality. Concerns over these issues have prompted researchers to study model robustness from multiple perspectives– such as privacy, fairness, security, interpretability, among others. In this thesis, we build upon these ideas of robustness, by investigating adversarial and social robustness for a number of different learning models and problem settings. We first study adversarial robustness of unsupervised clustering models, by proposing novel poisoning and evasion attacks for both deep and classical models. We then study the social robustness of models in the context of fairness, and propose the antidote data problem for fair clustering, as well as the fair video summarization problem. Finally, we investigate two problems at the intersection of adversarial and social robustness. We propose a new robust fair clustering method that can jointly ensure adversarial and social robustness, and data selection approaches that can improve interpretability, and optimize the utility, fairness, and robustness for classification models. Through the concepts and ideas proposed in this thesis we aim to lay the groundwork for analyzing and ensuring robustness of ML/DL models of the future.

Acknowledgments

I would like to express my deepest gratitude to my advisor, Prof. Prasant Mohapatra, who believed in me from the very beginning and gave me the opportunity to pursue multiple research directions independently. His steady guidance and mentorship have instilled tremendous confidence in me, and allowed me to advance my research acumen manifold. I would also like to thank my committee members, Prof. Xin Liu and Prof. Hao-Chuan Wang for their support and feedback, and for helping me augment the work constituting this thesis.

I am indebted and grateful to Prof. Hongfu Liu, Prof. Magdalena Wojcieszak, and Prof. Adish Singla for their invaluable advice, guidance, and encouragement through various collaborations over the years. In this regard I am especially thankful to Prof. Hongfu Liu for mentoring me closely over the last two years.

I also wish to express my appreciation for my peers, colleagues, and mentees for working to solve some of the challenging research problems we have embarked on together, and to all my wonderful friends for the countless adventures we have indulged in over the years. I remain thankful to Karina for her unwavering support, belief and love, which further bolstered my confidence and allowed me to reach greater heights both personally and professionally. I am also incredibly grateful to my grandparents for their ceaseless love throughout this journey.

I am perpetually thankful to my sister, Sonakshi, who continues to encourage me every day and whose sustained confidence in my abilities has motivated me in incalculable ways. Through her actions, she has also taught me the value of patience and empathy, both of which are desirable skills for any academic to possess.

Above all, I dedicate this thesis to my parents for the countless sacrifices they made during my formative years, and for the constant love and support they have provided me with throughout. From a young age, my mother has taught me the importance of taking pride in my work, paying attention to detail, and imbued in me a passion for teaching. Complementarily, my father has taught me the importance of being available and supporting those around you, irrespective of personal convenience or preference. Collectively, it is their belief and encouragement that enabled my younger self to embark on this journey five years ago, and achieve the goals I set for myself. I owe all my successes to them.

CHAPTER 1

Introduction

1.1. Motivation

Recently, machine learning (ML) and deep learning (DL) models have been extremely successful at achieving state-of-the-art performance on a number of learning tasks, leading to significant impacts in a diverse set of fields [1, 2, 3, 4]. However, these models suffer from issues of *robustness*, as was first discovered by Szegedy et al [5] in their work on deep learning based classification networks. Follow-up work [6, 7] demonstrated that ML and DL models are highly susceptible to such *adversarial attacks*, where an informed adversary can perturb inputs minimally to lead to undesirable model outputs, often with regards to some notion of model performance. In this thesis, we explore and extend these concepts of *model robustness* in multiple ways.

In general, *robustness* encompasses many different subproblems related to model functioning on manipulated or corrupted inputs, out-of-distribution generalization, cross-domain generalization, domain shift, performance on unseen test data, and resilience to adversarial attacks [8]. In this thesis, we utilize these definitions but structure them as different *dimensions* of model robustness. For instance, *adversarial robustness* constitutes the study of attacks and defenses against models, whereas *performance robustness* includes consistent performance on distribution-shifted, distorted, or unseen test data. As models become ubiquitously utilized in society, ensuring model robustness from diverse perspectives is of paramount importance. Our work thus also motivates the need for a *social dimension* to robustness, comprising the societal facets of models such as fairness [9] and explainability [10], among others. We delineate the various aspects of robustness covered in this thesis, below.

First, we find that the study of *adversarial robustness* has remained limited mostly to supervised models for classification tasks, with very limited work undertaken for unsupervised learning problems such as data clustering [11, 12]. In such scenarios, adversarial robustness is significantly more

challenging to define due to the absence of class labels corresponding to data samples. Thus, in Chapter 2, we bridge this gap by proposing novel black-box *poisoning* and *evasion* attacks against clustering models. In this manner, our work investigates adversarial robustness for unsupervised classification (or clustering) models for both classical and deep learning based methods. For completeness, we also delineate possible defense strategies against our attacks, but find that better approaches need to be proposed. Finally, we highlight the efficacy of our attacks by attacking `Face++`[1], a production-level MLaaS (ML-as-a-service) photo clustering API and showcasing the subsequent drop in performance.

Second, we posit that model robustness extends beyond analyzing security properties of models under adversarial influence, and models need to be robust *socially* for widespread societal adoption. This definition for *social robustness* encompasses recent work on *fair, accountable, transparent,* and *explainable* ML/DL [9, 13]. However, while the ML community has responded to the need for accommodating these social robustness requirements in models, there are still a number of potential problems that have been overlooked [14, 15, 16, 17]. In Chapter 3, we focus specifically on fairness, and provide novel directions for enhancing fairness and social robustness of models. We discern that most works studying fairness in classification and clustering propose in-processing fair algorithmic variants to models [15]. However, this direction for research is untenable as fairness notions are application-specific and new learning models are being developed rapidly. This results in a large number of possible fairness notion - learning model combinations requiring new fair models for each such combination. Thus, we propose novel approaches for *fair clustering* based on *antidote data* [18], that work with any fairness notion and learning model provided as input at runtime. Furthermore, we observe that the lack of development of fair models for certain ML tasks stems from a lack of appropriate datasets containing information regarding individuals and their sensitive attributes, such as *ethnicity* and *age*. We hence define the *fair video summarization* learning task which incorporates fairness in the traditional vision task of video summarization [19] which is conceptually adjacent to clustering. As there are currently no datasets that facilitate fair video summarization, we propose the *FairVidSum* dataset [20] which includes annotations for sensitive attributes and individuals alongside video frame importance scores. Using *FairVidSum* and newly proposed fairness metrics,

[1] `https://www.faceplusplus.com/photo-album-clustering/`

we benchmark the fairness of various state-of-the-art video summarization models and underscore the need for improved fair video summarization models.

Finally, we study the interplay between the adversarial and social dimensions of robustness in the context of machine learning. As models are deployed in more societal applications, they need to be robust both adversarially and socially. Deficiencies and vulnerabilities relating to either dimension would mitigate the model's impact and diminish its utility in society. Therefore, in Chapter 4, we study two problems at the intersection of adversarial and social robustness. Continuing with our use-case of clustering, we propose a novel poisoning attack that reduces clustering fairness [21]. We show that numerous state-of-the-art fair clustering models tend to be extremely volatile with regards to fairness when our attack is employed. Moreover, we propose a novel *robust fair clustering* model based on consensus clustering and fair graph partitioning, that is resilient to our proposed fairness attack. Then, reverting to the more traditional classification setting, we show how simple data selection approaches based on *influence functions* [22] can improve a classification model's fairness, robustness, and accuracy [23]. We then extend these ideas to non-traditional classification settings, such as active learning [24], and online learning [25]. Our work thus ties adversarial and social robustness together via multiple facets for a number of diverse learning tasks.

It is also important to note that the primary goal of this thesis is to provide general solutions to these proposed problems, wherever possible. It is not possible to study all dimensions of robustness along all possible ML/DL tasks and models, but ideas proposed for a problem defined on one ML task are often transferable across adjacent or complementary tasks. This has been true for concepts proposed in this thesis as well, with Cinà et al [26] expanding our original bi-clustering poisoning attack (Chapter 2) to the general cluster case, and Li et al [27] utilizing antidote data for mitigating (individual) unfairness in classification models based on our work on fair clustering (Chapter 3), among others.

The rest of the thesis proceeds as follows: Chapter 2 discusses adversarial robustness and our work on analyzing it for classical and deep clustering models, Chapter 3 delineates social robustness in the context of ML and our contributions to enhancing the fairness of models, Chapter 4 provides perspective and insights on novel problems relating to both adversarial and social robustness, and Chapter 5 concludes the thesis.

1.2. Related Works

Model Robustness. Existing work on model robustness has investigated problems related to improving *performance robustness* via self-supervised learning [28], pre-training [29], or other alternatives [30, 31]. Other problems along the performance dimension of robustness have also been studied previously– such as tackling distribution shift [32] and corrupted/contaminated data [33, 34]. Adversarial robustness has been the most well-studied dimension of robustness, with multiple works investigating adversarial attacks and defenses, for various learning tasks, models, and threat models [5, 35, 36, 37, 38]. More recently, work on social robustness has also been undertaken, specifically with regards to fairness [39, 40, 41].

Deep Clustering. To leverage clustering algorithms on high-dimensional data, early work on deep clustering [42, 43], aimed to learn a latent low-dimensional *cluster-friendly* representation that could then be clustered on. This task was achieved using AutoEncoders (AEs) [42, 44, 45], Variational AEs [46, 47], or Generative Adversarial Networks (GANs) [48, 49, 50]. More recently, current state-of-the-art deep clustering models employ self-supervised and contrastive learning instead of the earlier AE/GAN based approaches to perform clustering [51, 52, 53].

Video Summarization. Video summarization approaches can be categorized [19] as either *supervised* (frame-level importance scores are used in training) [54, 55, 56, 57, 58] or *unsupervised* (only visual frame information is used during training) [59, 60, 61, 62, 63, 64] with regards to the learning setting, and as *unimodal* (only visual information is used for training) [54, 57, 60, 61] or *multimodal* (other video metadata is also utilized) [65, 66, 67, 68, 69, 70] with regards to the input data type. *Unsupervised unimodal* approaches model the application scenarios for video summarization better as annotated frame importance scores and additional video metadata (such as transcripts) are generally hard to obtain [19]. For video summarization, the state-of-the-art supervised approaches constitute DSNet [71], PGL-SUM [72], among others and the state-of-the-art unsupervised approaches constitute CA-SUM [73], AC-SUM-GAN [74], SUM-GAN-AAE [75], SUM-GAN-SL [63]. All these models are benchmarked on the *TVSum* [76] and *SumMe* [77] datasets.

Adversarial Attacks Against Clustering. Ours is the first work to propose a blackbox adversarial *poisoning* attack against clustering [11]. These seek to poison a small number of samples in the input data, so that when clustering is undertaken on the poisoned dataset, other unperturbed

samples change cluster memberships. This attack setting is better suited to classical clustering algorithms such as k-means [78] which *retrain* on the poisoned input data constituting an attack at training time. For deep clustering, the model does not train again once deployed, so we have to generate adversarial samples that the model misclusters at *test time*. For this reason, we also propose the first *evasion* adversarial attack specific to deep clustering models [12]. There have also been whitebox attacks proposed for traditional clustering, but these can only be employed for the specific traditional clustering algorithm considered [79, 80, 81]. For single-linkage hierarchical clustering, [79] first proposed the *poisoning* and *obfuscation* attack settings, and provided algorithms that aimed to reduce clustering performance. In [80] the authors extended this work to complete-linkage hierarchical clustering and [81] proposed a white-box poisoning attack for DBSCAN [82] clustering.

Fairness in Machine Learning and Summarization. While video summarization has not yet been studied from the purview of fairness, fair models have been developed for various ML tasks and problem settings [9, 14]. These include supervised learning [83, 84], unsupervised learning [15, 16, 85], recommendation systems [18, 86], active learning [87, 88], outlier detection [89, 90], among others. Fairness has also been studied for data summarization such as for k-center based summarization [91, 92, 93] and text summarization [94, 95]. However, these approaches are not general– they are highly specific to the learning algorithm being used (for e.g. k-center [96]).

Fairness in Clustering. ML algorithms can be made *fair* in three stages of the learning pipeline [9, 97]– *before-training* (pre-processing the dataset), *during-training* (changing the ML algorithm), or *after-training* (post-processing the learnt model). Fair clustering aims to conduct unsupervised cluster analysis without encoding any bias into the instance assignments. Most research on fair clustering focuses on the *during-training* phase [85, 98, 99, 100, 101, 102, 103, 104] and proposes fair clustering algorithms. In their paper, [105] study the *after-training* phase for improving fairness post-clustering. Another categorization for fairness can be made between group-level and individual-level fairness notions [14]. The former encapsulates notions that seek to quantify how the model's predictions affect protected or minority groups of people (for example, based on *gender*) unfairly, while the latter comprises notions that seek to ensure that similar individuals are treated similarly by the model. In this thesis, we only consider group-level fairness.

Robustness and Fairness. The robustness of fairness is the study of how algorithmic fairness could be violated or preserved under adversarial attacks or random perturbations. [106] and [107] propose poisoning attack frameworks aiming to violate the predictive parity among subgroups in classification. [108] and [109] also work on classification and study fairness under perturbations on protected attributes. Other work has also studied the reduction in fairness (defined alternately as class-wise performance) when adversarial training is undertaken [39], and proposed novel approaches for fair and robust classification [40, 41].

1.3. Contributions

Through the individual works that comprise this thesis, we make the following contributions:

- We propose the first black-box poisoning attack against classical clustering models which results in non-perturbed samples being misclustered by the model [11]. We theoretically and experimentally validate the extent of our poisoning attack on k-means [78] and Ward's hierarchical clustering [110] models.
- We propose the first black-box evasion attack specific to deep clustering models [12]. We show that our attack can significantly reduce the performance of state-of-the-art models while requiring a minimal number of queries. We also undertake transferability analysis that demonstrates that the attack can be successful even if the attacker does not have exact knowledge of the target model.
- We propose an alternative approach to group-level fair clustering, where we augment the original dataset with data points (antidote data) such that when we use vanilla clustering on this new combined dataset, fairness is improved [15]. This is the first work that utilizes data augmentation and antidote points for improving fairness in clustering. In contrast, existing works on fair clustering modify the clustering algorithm specific to a notion of group-level fairness.
- We introduce *FairVidSum*– the first benchmark video summarization dataset containing individual and group-level fairness information [20]. *FairVidSum* comprises of videos comprising diverse settings such as panel discussions, podcasts, and interviews. We also propose the *SumBal* fairness metric to evaluate model fairness, which is derived from

the Balance [14] metric proposed to measure fairness in unsupervised learning. Using the FairVidSum dataset and the SumBal metric, we conduct extensive experiments to benchmark numerous state-of-the-art supervised and unsupervised models for unfairness. We find that since models do not optimize for fairness, they can be highly unfair, prompting the need for newer methods that can balance both accuracy and fairness.

- We propose a novel black-box adversarial attack against clustering models where the attacker can perturb a small percentage of protected group memberships and yet is able to degrade the fairness performance of state-of-the-art fair clustering models significantly [21]. To achieve truly robust fair clustering, we propose the Consensus Fair Clustering (CFC) model which is highly resilient to the proposed fairness attack. To the best of our knowledge, CFC is the first defense approach for fairness attacks, which makes it an important contribution to the unsupervised ML community.
- We propose the use of tree-based influence estimation models to better interpret what features and samples improve (or degrade) a classification model's performance with respect to utility, fairness, and robustness [23]. Based on that, we also propose a simple training data trimming strategy to improve fairness, robustness, and accuracy.

The research undertaken as part of this dissertation has resulted in the following publications [11, 12, 20, 21, 23, 111].

CHAPTER 2

Analyzing the Adversarial Robustness of Models

2.1. Introduction

Clustering models are utilized in many data-driven applications to group similar samples together, and dissimilar samples separately. They constitute a powerful class of unsupervised ML models which can be employed in many cases where labels for data samples are either hard (or impossible) to obtain. Note that there are a multitude of different approaches to accomplishing the aforementioned clustering task, and an important differentiation can be made between *traditional* and *deep* clustering approaches.

Traditional or classical clustering generally aims to minimize a clustering objective function defined using a given distance metric [112]. These include approaches such as k-means [78], DBSCAN [82], spectral methods [113], hierarchical algorithms [110], among others. Most classical models are considerably fast[1] on moderately-sized tabular datasets and provide satisfactory clustering solutions. On the other hand, such models generally fail to perform satisfactorily on high-dimensional data (i.e., high resolution image datasets), or incur huge computational costs that make the problem intractable [114]. To ameliorate these challenges on high-dimensional data, deep clustering models were proposed to decompose the high-dimensional input to a *cluster-friendly* low-dimensional representation using deep neural networks. Clustering was then undertaken on this latent space representation [42, 43]. Since then, deep clustering models have become considerably advanced, with state-of-the-art (SOTA) models outperforming traditional clustering models by significant margins on a number of high-dimensional image datasets [51, 53].

Despite these successes, clustering models have not been sufficiently analyzed from an *adversarial robustness* perspective. Furthermore, no work investigates *generalized black-box* attacks, where the adversary has zero knowledge of the exact clustering model being used. This is the most realistic

[1] k-means has linear time complexity with regards to both the samples in the dataset and the number of clusters.

setting under which a malicious adversary could aim to disrupt the working of these models. While there is a multitude of work in this domain for supervised learning models, deep clustering approaches have not received the same attention from the community. Thus, to bridge this gap we propose two black-box attacks against clustering models in this chapter. First, since classical models are computationally efficient and retrain on data input, we propose a *training-time poisoning* attack [80] that works by perturbing a single sample close to the decision boundary, which leads to the misclustering of multiple unperturbed samples, named *spill-over adversarial samples*. Next, since deep clustering models are deployed *frozen* for inference post training, we propose a *test-time evasion* attack [7] that utilizes Generative Adversarial Networks (GANs) [115] to generate adversarial images that are misclustered by SOTA deep clustering models despite being minimally perturbed.

2.2. Poisoning Attacks Against Classical Clustering Models

2.2.1. Preliminaries and Notation. Let X be the dataset used for clustering, where $X \in \mathbb{R}^{n \times m}$. The Frobenius norm of a matrix M is denoted by $\|A\|_F = (\sum_{i,j} A_{ij}^2)^{1/2}$, and $\langle \cdot, \cdot \rangle$ represents the standard vector inner product. Let a clustering algorithm be defined as a function $C : \mathbb{R}^{n \times m} \to \{0,1\}^{n \times k}$ that partitions the given dataset $X \in \mathbb{R}^{n \times m}$ into k clusters. Since we are only considering hard clustering problems, we can represent the clustering result as a matrix $Y \in \{0,1\}^{n \times k}$ where each row has all 0 except one 1 indicating the cluster that point belongs to, and thus, $Y = C(X)$. Here n refers to the number of samples in the dataset and m refers to the features. For the clustering output, let k_i and $X_{k_i} \in \mathbb{R}^{n_i \times m}$ denote the i-th cluster and the samples of X that belong to the i-th cluster, respectively. The centroid of the i-th cluster is denoted as c_i. Moreover, we will denote the set of spill-over samples as S, with the number of samples in S denoted as n_S.

2.2.2. Threat Model. We delineate the main features of the threat model that the adversary operates under:

(1) The adversary has no knowledge of the clustering algorithm that has been used and is thus, going to carry out a *black-box* attack.

(2) While the adversary does not have access to the algorithm, we assume that they have access to the datasets, and a noisy version of the true metric used for clustering. This assumption is weak as the metric can be learnt by observing the clustering outputs by the defender.

For details see [116] and the references therein. We therefore first present our algorithm assuming exact knowledge of metric. We then show later that, if the noise is small enough, under certain conditions, a spill-over adversarial sample in clustering using the noisy metric will also spill-over in clustering using the true metric.

(3) Once the attacker has the clusters, they can then use the proposed adversarial attack algorithm to perturb just one judiciously chosen input sample, called the *target sample*. The algorithm perturbs this input data sample by iteratively crafting a precise additive noise to generate the spill-over adversarial samples. We allow the perturbation to be different for each feature of the sample. We assume that each perturbation is within a pre-specified threshold which is determined by adversary's motivation of not getting detected as an outlier, and/or the limited attack budget of the adversary.

2.2.3. Adversary's Objective. The goal of the adversary is to maximize the number of points which are misclustered into the target cluster. Note that, perturbation of the target sample essentially perturbs the decision boundary which creates spill-over adversarial samples but the target sample may not necessarily be misclustered. The adversarial attack algorithm proposed operates on only two clusters at a time (refer to [26] for their work on generalizing our approach to the multi-cluster case). This is similar to a targeted adversarial attack on supervised learning models [7]. Note that our attack has a number of motivating cases in the real world. Similar to [81], where the authors present an attack algorithm against DBSCAN clustering [117], consider the AnDarwin tool [118] that clusters Android apps into plagiarized and non-plagiarized clusters. If an adversary perturbs one plagiarized app such that it along with some other unperturbed apps, gets misclustered, that could lead to a loss of confidence in the tool as the defender is unaware of the reason for this decreased performance.

2.2.4. Proposed Attack. The proposed algorithm is shown as Algorithm 1. The inputs for the algorithm are the dataset $X \in \mathbb{R}^{n \times m}$, the clustering algorithm C, the clustering result on the original data $Y \in \{0,1\}^{n \times 2}$, the data points that populate each of the two clusters, $X_{k_1} \in \mathbb{R}^{n_1 \times m}$ and $X_{k_2} \in \mathbb{R}^{n_2 \times m}$, and the noise threshold $\Delta \in \mathbb{R}^m$ where k_i $(i = \{1,2\})$ denotes the clusters. Δ is the noise threshold for each of the m features, i.e., the j^{th} feature of the optimal perturbation will

Algorithm 1 Proposed Black-box Poisoning Attack

Input: X, C, Δ, $Y = C(X)$, k_1, k_2, n_1, n_2, X_{k_1},X_{k_2}
Output: Optimal additive perturbation $\epsilon^* \in \mathbb{R}^m$

1: **set** $c_2 \leftarrow \frac{1}{n_2} \sum_{x_j \in X_{k_2}} x_j$
2: **set** $x_t \leftarrow \text{argmin}_{x \in X_{k_1}} |x - c_2|$
3: **function** $f(\epsilon)$
4: **set** $x_t' \leftarrow x_t + \epsilon$
5: **obtain** X' **from** X **by replacing** x_t **with** x_t'
6: **obtain** $Y' = C(X')$
7: **set** $\delta := -\|YY^T - Y'Y'^T\|_F$
8: **return** δ
9: **end function**
10: **minimize** $f(\epsilon^*)$ **subject to** $\epsilon_j^* \in [-\Delta_j, \Delta_j]$ **where** $j = 1, 2, ..., m$
11: **return** ϵ^*

lie in the range $[-\Delta_j, \Delta_j]$ where $j = 1, .., m$. This definition for Δ can lead to the case where points of k_2 spill-over into k_1, but since the formulation is equivalent, we consider only the case of spill-over from k_1 to k_2. Δ ensures that the adversary does not perturb the target sample too much to get detected by the defender as an outlier. Δ can also be interpreted as the limited attack budget of the adversary. We elaborate on how to choose Δ at the end of this section.

Algorithm 1 proceeds as follows: In Line 1, we find the centroid of whichever cluster we want the spill-over points to be a part of, after the attack. From here on, without loss of generality, we assume the spill-over points belong to cluster k_1 originally, and therefore cluster centroid c_2 for k_2 is calculated. Next, in Line 2, we select the target point x_t in k_1 which is closest in Euclidean distance to c_2. This point is a good target for the adversarial attack as it is the nearest point of k_1 to the decision boundary between both clusters. Lines 3-10 define the function $f(\epsilon) \in \mathbb{R}$ which we optimize over to find the ϵ that will lead to spill-over.

In Line 4 of the algorithm, we perturb the target point and obtain x_t', and get X' by replacing x_t with x_t' in X. We then find the noisy clustering result $Y' = C(X')$. Line 5 presents the metric δ used to measure how much the clustering result has *changed* from the original clustering to after the attack [79]:

$$\delta := -\|YY^T - Y'Y'^T\|_F \tag{2.1}$$

The ij^{th} element of the YY^T matrix represents whether sample i, and j belong to the same cluster. Note that, if there is no change in cluster membership, $\delta = 0$. $|\delta|$ increases with the number of points that spill over from k_1 to k_2.

Line 11 is essentially the formulation of the minimization problem. We have to find the optimal perturbation $\epsilon_j^* \in [-\Delta_j, \Delta_j]$ which minimizes f, such that $f(\epsilon^*) \leq f(\epsilon)$ for any $\epsilon \in [-\Delta_j, \Delta_j]$, $j = 1, 2, \cdots, m$. It is also important to understand a few aspects about the function f, before we get to the choice of an optimization approach. As the function is not continuous, we cannot use gradient based methods to solve the minimization problem. Instead, we require derivative-free black-box optimization approaches to minimize f while ensuring that the noise threshold constraints on the optimal perturbation ϵ^* are met. There are many possibilities for such an optimization procedure, e.g., genetic algorithms [119], and simulated annealing [120]. We opt for a cubic radial basis function (RBF) based surface response methodology [121] for the optimization. The optimization approach utilizes a modified iterative version of the CORS algorithm [122], and uses the Latin hypercube approach [123] for the initial uniform sampling of search points required for the CORS algorithm. The CORS optimization procedure has achieved competitive results on a number of different test functions [122, 124]. For our attack algorithm, this optimization algorithm achieved much better results on multiple datasets as compared to methods like genetic algorithm, and simulated annealing. We found that this optimization was much less sensitive to parameter choices.

If the adversary does not have an attack budget, then Δ should only be chosen such that the adversarial sample does not get construed as an outlier. Mahalanobis Depth (MD) is one such measure for *outlyingness*:

Definition 2.2.1 (Mahalanobis Depth). Mahalanobis Depth of a point x, $\mathcal{D}(x)$, with respect to a set $X \subseteq \mathbb{R}^m$ is defined as

$$\mathcal{D}(x) = \left(1 + (x - \bar{x})^\top \hat{\Sigma_x}^{-1} (x - \bar{x})\right)^{-1} \tag{2.2}$$

where $\bar{x}$, and $\hat{\Sigma_x}$ are the sample mean and covariance.

The smaller the value of $\mathcal{D}$, the larger the *outlyingness*. Using $\mathcal{D}$ to detect the *outlyingness* of a point for a dataset with clusters may pose problems, e.g., for two well separated clusters the points around the line joining the cluster means have very small depth. But a point between two clusters

which is sufficiently far from both the clusters will clearly be interpreted as an outlier. So we propose a modified measure of depth similar in flavor to [125]:

DEFINITION 2.2.2 (Mahalanobis Depth for Clusters (MDC)). Let there be J clusters. Say $\mathcal{D}(x)$ with respect to only cluster i is given by t_i. Then Mahalanobis Depth for clusters of x is defined as $\mathcal{D}_C(x) = \sum_{i=1}^{J} t_i$.

For high dimensional data computing MD is difficult as the computed covariance matrix can be singular. So for high dimensional data we propose a new depth-based measure, Coordinate-wise Min-Mahalanobis-Depth (COMD), to measure outlyingness:

DEFINITION 2.2.3 (Coordinate-wise Min-Mahalanobis-Depth). Consider $x = [x_1, x_2, \cdots, x_m] \in X \subseteq \mathbb{R}^{n \times m}$. Let X_i denote the i^{th} column of X. Let $\mathcal{D}_{C,i}$ denote the MDC depth of x_i w.r.t the points $X_i \subseteq \mathbb{R}^{n \times 1}$. Then Coordinate-wise Min-Mahalanobis-Depth is defined as $\mathcal{D}_M(x) = \min\{\mathcal{D}_{C,i}\}_{i=1}^{m}$.

Intuitively COMD measures the maximum *outlyingness* along all the coordinates. It is a conservative measure of outlyingness, and hence ensuring small value of COMD is sufficient for the adversary to avoid being detected as an outlier. After observing the clusters, the attacker forms equi-$\mathcal{D}_C(x)$ contours, or equi-$\mathcal{D}_M(x)$ spaces for $m \geq 2$, over the data. Δ is chosen such that the perturbed point is at least above 0.1 quantile of the COMD values of the dataset.

2.2.5. Theoretical Justification. In this section we theoretically study the effect of perturbation, and using a noisy metric. The following theorem shows that perturbing one sample can distort the decision boundary in such a way that another uncorrupted point can spill-over.

THEOREM 2.2.1. Say k-means clustering is used to cluster a linearly separable dataset. A judiciously chosen datapoint can be perturbed by additive noise, ensuring that it does not become an outlier, in such a way that there may exist another point which changes cluster membership, i.e., one or more spill-over point(s) may exist.

PROOF. Let a point x is such that $\langle x - c_1, c_2 - c_1 \rangle \geq 0$. Intuitively we select x belonging to k_1 such that $x - c_1$ forms an acute angle with $c_2 - c_1$. It is easy to see that such a point always exists. Now, an adversary perturbs x to c_2, hence ensuring that the perturbed point is not an outlier. We

will show by contradiction that, under certain conditions, there exists at least one other point which will spill over to k_2.

Let us assume that k_1 remain unchanged after the perturbation except that x moves to c_2. Because of this perturbation the mean and composition of the k_2 does not change. Let the mean of the k_1 be c_1' after the perturbation. We have,

$c_1 - c_1' = c_1 - \frac{n_1 c_1 - x}{n_1 - 1} = \frac{x - c_1}{n_1 - 1}$. Say there is a point y such that $\langle x - c_1, y - c_1 \rangle \geq 0$, and $\|y - c_2\|^2 = \|y - c_1\|^2 + \alpha$ with $\alpha \geq 0$. Consequently, $\langle y - c_1, c_1 - c_1' \rangle = \langle y - c_1, \frac{x - c_1}{n_1 - 1} \rangle \geq 0$. Now we have,

$$\begin{aligned} &\|y - c_1'\|^2 \\ =&\|y - c_1 + c_1 - c_1'\|^2 \\ =&\|y - c_2\|^2 + \|c_1 - c_1'\|^2 + 2\langle y - c_1, c_1 - c_1' \rangle - \alpha \end{aligned}$$

The second and third term in the above expression is non-negative. If $\alpha \leq \|c_1 - c_1'\|^2 + 2\langle y - c_1, c_1 - c_1' \rangle$, then the point y is closer to c_2 and should be in k_2. This contradicts our earlier assumption and concludes the proof. □

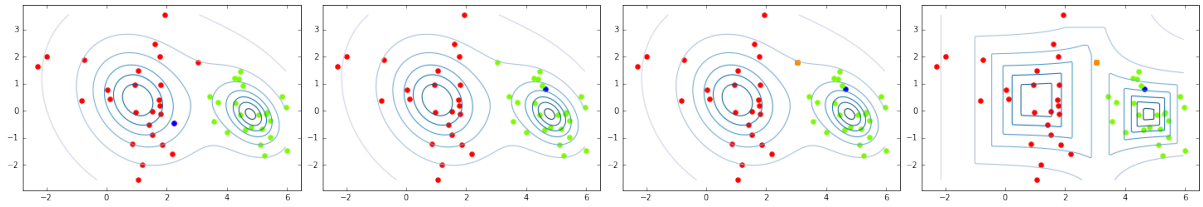
As discussed above, we assume that the adversary has access to a noisy version of the true metric. In the following theorem we show that a spill-over adversarial sample under noisy metric will spill-over under clustering with true metric under certain conditions.

THEOREM 2.2.2. Let there be a point y which spilled over from k_1 to k_2 of the dataset X due to a attack following Algorithm 1 with distance metric $d' \colon X \times X \to \mathbb{R}^+$. If the true distance metric $d \colon X \times X \to \mathbb{R}^+$ used for clustering by the defender satisfies

$$\max\{0, d(u, v) - \zeta\} \leq d'(u, v) \leq d(u, v) + \zeta \tag{2.3}$$

$\forall (u, v) \in X$, and $\zeta \geq 0$, then, under certain conditions, y will spill-over in the clustering using metric d as well.

PROOF. Let the cluster centers for $d\,(d')$ be $c_1\,(c_1')$ and $c_2\,(c_2')$, and the clusters be $k_1\,(k_1')$, and $k_2\,(k_2')$. We assume that the attacker can query to find out which point of k_1' is closest to c_2. We use



(a) (b) (c) (d)

FIGURE 2.1. Results on toy data after using Algorithm 1 for spill-over attack.

the attack used to prove Theorem 1, i.e., the attacker perturbs a point x from k'_1, to the center of k'_2. Consequently, c'_1 becomes $\bar{c_1}'$, c'_2 remains the same, and another point y spills-over from k'_1 to k'_2.

After the attack, in the actual clustering using d, let the centers be represented by $\bar{c_1}$ and $\bar{c_2}$. We will show that y will spill-over in the actual clustering using d as well, under certain conditions. Now, we have before the attack:

$$d'(y, c'_2) > d'(y, c'_1) \tag{2.4}$$

Using the triangle inequality twice on (2.4) we can write:

$$d'(y, c'_2) > d'(y, \bar{c_1}') - d'(\bar{c_1}', c'_1)$$

$$d'(y, \bar{c_2}) + d'(\bar{c_2}, c'_2) > d'(y, \bar{c_1}) - d'(\bar{c_1}', \bar{c_1}) - d'(\bar{c_1}', c'_1)$$

Let $\gamma = d'(\bar{c_1}', c'_1) + d'(\bar{c_1}', \bar{c_1}) + d'(\bar{c_2}, c'_2) \geq 0$, and using (2.3) we can write:

$$d'(y, \bar{c_2}) - d'(y, \bar{c_1}) > -\gamma$$

$$d(y, \bar{c_2}) - d(y, \bar{c_1}) > -\gamma - 2\zeta \tag{2.5}$$

If $d(y, \bar{c_2}) - d(y, \bar{c_1}) < 0$ then point y has spilled-over in the actual clustering too, and we can see that the lower bound in (2.5) is negative as both ζ and γ are non-negative. Therefore, this ensures that $d(y, \bar{c_2}) - d(y, \bar{c_1})$ is negative for a range of values of y. □

2.2.6. Results. We now present results for our attack for synthetic toy data and then for a number of real-world datasets and traditional clustering models.

2.2.6.1. *Toy Example.* We present the working of our attack algorithm consistent with the assumptions in Theorem 2.2.1. We create 2-dimensional Gaussian clusters with standard deviations of 1.45 and 0.75, and cluster centroids at $(1, 0)$ and $(5, 0)$, respectively. Using Algorithm 1 we find the target point x_t to perturb which is originally in k_1. The clusters k_1 and k_2 generated along with x_t are shown in Figure 2.1a. The first cluster k_1, the second cluster k_2, and the target sample x_t are shown in red, green, and blue respectively.

It is important to note that the assumption taken in Theorem 2.2.1 regarding the target sample also holds true as $\langle x_t - c_1, c_2 - c_1 \rangle = 5.0511$, where c_1 and c_2 denote the cluster centroids of k_1 and k_2. Next, using the optimization procedure outlined in Algorithm 1, we perturb x_t in such a way so as to lead to spill-over. Figure 2.1b shows that the perturbed x_t has changed cluster membership, and there is one spill-over adversarial sample. Figure 2.1 also shows the equi-$\mathcal{D}_C$ contours with depth decreasing by 0.1, away from the cluster centers. The contours in Figure 2.1 show that the adversarial sample is not an outlier. Figure 2.1a-2.1c show the equi-$\mathcal{D}_C$ contours, and Figure 2.1d shows the equi-$\mathcal{D}_M$ contours. Note that $\mathcal{D}_C$ correctly prescribes more *outlyingness* for points that lie between two clusters, which do not belong to either cluster, as compared to points which belong to one of the clusters. The spill-over adversarial sample has been highlighted in orange in Figure 2.1c. We find that the spill-over adversarial sample y satisfies the condition stated in Theorem 2.2.1, i.e., $\langle x_t - c_1, y - c_1 \rangle = 1.2872 \geq 0$.

2.2.6.2. *UCI Digits.* The UCI Digits dataset [126] consists of 8×8 images of handwritten digits from 0 to 9. In these images each pixel is an integer between 0, and 16. Each image can be represented by feature vectors of length 64. We test Ward's clustering for this dataset since it clusters the digits well. We use these images as inputs to the clustering algorithm. We apply Ward's clustering on two clustering problems: For clustering Digits `1` and `4` images, and clustering Digits `8` and `9` images. For each case, the cluster information, parameter details, total misclustered samples, $\mathcal{D}_M(x_t')$, and the quantile of $\mathcal{D}_M(x_t')$ with respect to $\mathcal{D}_M(X)$, are listed in Table 2.1. Algorithm 1 starts with the target sample from k_1, and generates spill-over adversarial samples that switch cluster membership from k_1 to k_2. For the Digits `1` and `4` clusters the spill-over adversarial images are shown in Figure 2.2, and for Digits `8` and `9`, they are shown in Figure 2.3. For both the clustering problems, the $\mathcal{D}_M(X)$ quantile above which $\mathcal{D}_M(x_t')$ lies, indicates that it cannot be an outlier.

TABLE 2.1. UCI Handwritten Digits Dataset Results (Ward's Clustering).

Digit clusters	k_1	k_2	n_1	n_2	# Misclustered samples	$\mathcal{D}_M(x_t')$	$\mathcal{D}_M(X)$ quantile
Digit 1 & 4	Digit 4	Digit 1	181	182	24	0.061	0.10
Digit 8 & 9	Digit 9	Digit 8	164	190	21	0.114	0.285

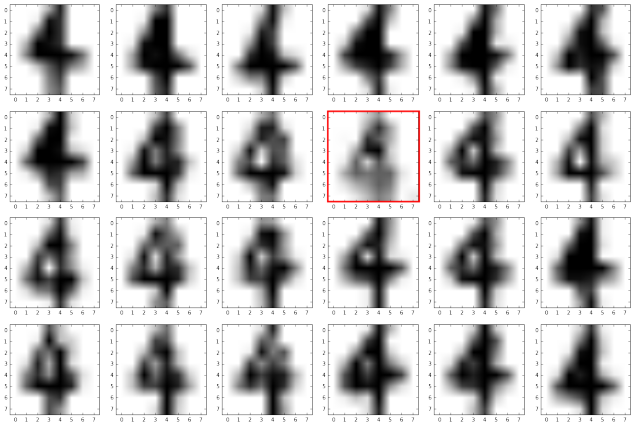


FIGURE 2.2. Misclustered images that switched clusters from the Digit 4 to the Digit 1 cluster (adversarially perturbed sample shown with red border).

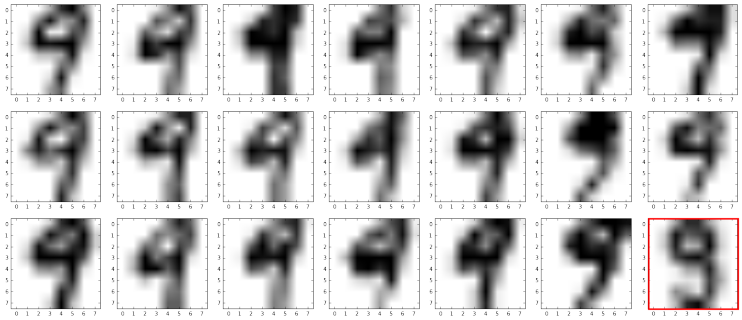


FIGURE 2.3. Misclustered images that switched clusters from the Digit 9 to the Digit 8 cluster (adversarially perturbed sample shown with red border).

2.2.6.3. *MNIST.* To show the performance of Algorithm 1 we use the MNIST dataset [127] which is an image dataset. We utilize small subsets of the original digit images, and use 200 images for each digit. The digit images here are 28×28 grayscale images of digits from 0 to 9. For inputs to the clustering, we flatten each image sample and get a feature vector of length $m = 784$. We apply Ward's clustering on two clustering problems: For clustering Digits 1 and 4 images, and clustering Digits 2 and 3 images. For each of these, the cluster information, parameter details, total misclustered samples, the perturbed target sample depth $\mathcal{D}_M(x_t')$, and the quantile of $\mathcal{D}_M(x_t')$ with

TABLE 2.2. MNIST Dataset Results (Ward's Clustering).

Digit clusters	k_1	k_2	n_1	n_2	# Misclustered samples	$\mathcal{D}_M(x_t')$	$\mathcal{D}_M(X)$ quantile
Digit 1 & 4	Digit 4	Digit 1	192	208	11	0.067	0.49
Digit 2 & 3	Digit 3	Digit 2	176	224	2	0.13	0.828

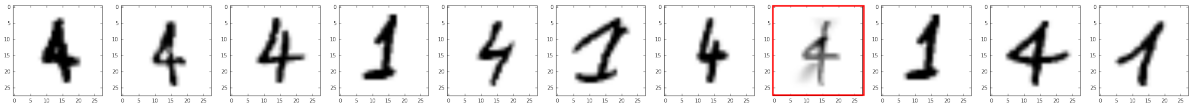


FIGURE 2.4. Misclustered MNIST images that switched clusters from the Digit 4 to the Digit 1 cluster (adversarially perturbed sample shown with red border).

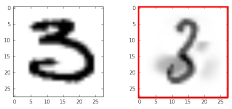


FIGURE 2.5. Misclustered MNIST images that switched clusters from the Digit 3 to the Digit 2 cluster (adversarially perturbed sample shown with red border).

respect to $\mathcal{D}_M(X)$, are listed in Table 2.2. The attack algorithm starts with the target sample from the k_1 cluster and then generates spill-over adversarial samples that switch cluster membership from k_1 to k_2. For the Digits 1 and 4 clusters the spill-over adversarial images are shown in Figure 2.4, and for Digits 2 and 3 clusters they are shown in Figure 2.5. Here too, the $\mathcal{D}_M(X)$ quantile range that $\mathcal{D}_M(x_t')$ lies in ensures that the perturbed adversarial sample cannot be detected as an outlier.

2.2.6.4. *UCI Wheat Seeds Dataset.* The UCI Wheat Seeds dataset [128] contains measurements of geometric properties of three different varieties of wheat kernels: Kama, Rosa, and Canadian, with 70 samples for each seed variety. Each sample has the following 7 features: Area of the kernel A, perimeter of the kernel P, compactness $C = 4\pi A/P^2$, kernel length, kernel width, asymmetry coefficient, and length of kernel groove. We use k-means clustering for clustering Kama and Rosa wheat kernels. Cluster sizes for Rosa is $n_1 = 79$, and for Kama is $n_2 = 61$. The noise threshold Δ is selected using the outlier depth methodology described in the previous sections. Algorithm 1 starts with the target sample in the Rosa cluster and generates 2 adversarial spill-over adversarial samples which have switched cluster labels from the Rosa cluster (or cluster k_1) to the Kama (or cluster k_2) cluster including the target sample. The original clustering with the target sample selected is

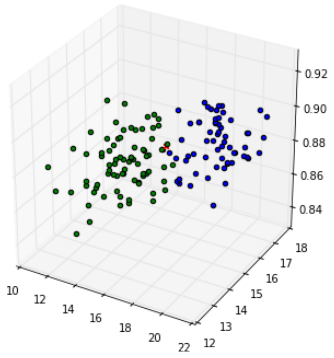


FIGURE 2.6. Kama and Rosa wheat kernel clusters (target sample to be adversarially perturbed in red) visualized using the *area*, *perimeter*, and *compactness* features.

shown in Figure 2.6, and is plotted in 3D using the area, perimeter, and compactness features. Here $\mathcal{D}_M(x'_t) = 0.33$ which is the 0.28 quantile of $\mathcal{D}_M(X)$ which ensures that it will not be an outlier.

2.2.6.5. *MoCap Hand Postures Dataset.* The MoCap Hand Postures dataset [129] consists of 5 types of hand postures/gestures from 12 users recorded in a motion capture environment using 11 unlabeled markers attached to a glove. We only use a small subset of the data with 200 samples for each cluster. For clustering, the possible features are each of the 11 markers' X, Y, Z coordinates. However, we only use the first 3 markers' recorded X, Y, Z coordinates because due to resolution and occlusion, missing values are common in the other markers' data. Thus, we have a total of 9 features: X_i, Y_i, Z_i for each i^{th} marker, where $i = 1, .., 3$. We use k-means clustering for clustering the Point1 posture and the Grab posture. Cluster size for Grab posture is $n_1 = 209$, and for Point1 posture is $n_2 = 191$. Algorithm 1 starts with the target sample in the Grab posture cluster, and generates 5 adversarial spill-over adversarial samples which have switched cluster labels from the Grab cluster (k_1) to the Point1 (k_2) cluster including the target sample. The original clustering with the target sample selected is shown in Figure 2.7, and is plotted in 3D using the Z_1, Z_2, Z_3 marker coordinates features. Here $\mathcal{D}_M(x'_t) = 0.325$ is the 0.27 quantile of $\mathcal{D}_M(X)$. This indicates that x'_t is not an outlier.

2.3. Evasion Attacks Against Deep Clustering Models

2.3.1. Preliminaries and Notation. Note that for a matrix A of size $u \times v$, we can index into row i as $A_i, \forall i \in [u]$, and index into a single value at row i and column j as $A_{i,j}, \forall i \in [u], j \in [v]$.

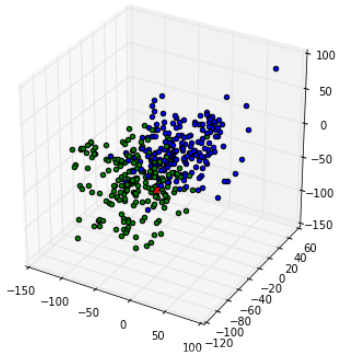


FIGURE 2.7. Point1 and Grab pose clusters (target sample to be adversarially perturbed in red) visualized using the Z_1, Z_2, Z_3 marker position features.

Moreover, ||.|| denotes the Euclidean (2-norm) norm of a vector. Since we are proposing a black-box attack we will be defining deep clustering models in a more generalized manner that abstracts their inner functioning. We defer the reader to [114] for more details on all the models analyzed. A deep clustering model is denoted as $\mathcal{C}$ and operates on samples of the given dataset X consisting of n samples and maps them to one of k clusters. As deep clustering models utilize deep neural networks internally, they generate a set of softmax probabilities indicating cluster memberships, denoted as $M \in [0,1]^{n \times k}$. From this, the cluster labels can be obtained by taking the maximum of the cluster probabilities. For a dataset sample $X_i \in X$, the cluster label can be obtained as $l = \text{argmax}_{j \in [k]}\{M_{i,j}\}$. We denote the vector of cluster labels computed in this manner as $L \in \mathbb{N}^{n \times 1}$.

Note that unlike supervised learning problems, the ground truth labels $Y \in \mathbb{N}^{n \times 1}$ are not utilized for training the model in deep clustering. However, these ground truth labels are used for evaluating the performance of the model. That is, the output cluster labels L are evaluated in comparison with the ground truth Y. Performance metrics such as Normalized Mutual Information (NMI) [130], Adjusted Rand Index (ARI) [131], and Unsupervised Accuracy (ACC) [132] are commonly used in deep clustering literature for this purpose. These metrics are defined analytically below:

ACC: This is the unsupervised equivalent of the traditional classification accuracy. Let there be a mapping function ϕ that computes all possible mappings between ground truth labels and possible cluster assignments[2] for some m samples. Then we can define ACC as:

[2]Such a mapping function can be computed optimally using the Hungarian assignment algorithm [133].

$$\text{ACC} = \max_{\phi} \frac{\sum_{i=1}^{m} \mathbb{1}\{Y_i = \phi(L_i)\}}{m}$$

NMI: Normalized Mutual Information is essentially a normalized version of the widely used mutual information metric. Let I denote the mutual information metric [134], and E denote entropy [134]. Then we can define NMI as:

$$\text{NMI} = \frac{I(Y, L)}{(1/2) * [E(Y) + E(L)]}$$

ARI: The Adjusted Rand Index is based on the Rand Index, commonly used in statistical literature [135]. Let R denote the "unadjusted" Rand Index, then we can write ARI as:

$$\text{ARI} = \frac{R - \mathbb{E}[R]}{\max(R) - \mathbb{E}[R]}$$

2.3.2. Threat Model. We now define the threat model for the adversary:

(1) The attacker has knowledge of the dataset X.

(2) The attacker carries out a black-box attack and does not know which deep clustering model $\mathcal{C}$ is being used, but can query it to observe M (softmax cluster memberships), and hence, L[3]. In adversarial attacks on (un)supervised models, this is often what constitutes a black-box attack [137, 138, 139, 140].

(3) The goal of the attack is to provide minimally perturbed images (there is a noise threshold that cannot be exceeded) as input to the deep clustering model. Upon doing so, the model should *miscluster* these samples and the performance measured via the evaluation metrics (ACC, NMI, ARI, etc) should significantly reduce post the attack.

2.3.3. Adversary's Objective. The goal of the attacker is to input adversarial images to the model and lead to a performance decrease, measured using metrics such as NMI, ACC, and ARI. We cannot directly optimize post-attack M or L as the adversary does not possess ground truth labels. Moreover, cluster labels generated by the model may not be the same as the ground truth labels, thus requiring some notion of a mapping function, further complicating the problem. Instead we will indirectly achieve this goal through another objective function.

[3]If only L can be observed it is a *decision-based* attack; if M can be observed it is a *score-based* attack [136].

Let an input image be $X_i \in X$ and through a single query the cluster probabilities M_i can be obtained before the attack. To attack, the adversary will introduce a carefully crafted perturbation/noise δ specific to this sample. The attacker queries the deep clustering model and obtains the set of cluster probabilities for this sample. Abusing notation slightly, these are denoted as $\mathcal{C}(X_i + \delta)$.

Assuming the original unperturbed cluster labels/probabilities accurately depict the cluster representing the sample's ground truth label, the attacker can simply generate the adversarial noise via the following optimization problem:

$$\max_{\delta} \quad ||M_i - \mathcal{C}(X_i + \delta)|| \quad \text{s.t.} \quad ||\delta|| \leq \epsilon \tag{2.6}$$

Here the constraint with ϵ is simply to ensure that the adversarial sample does not have unbounded noise and remains *realistic* to a human observer. The objective function above ensures that the cluster probabilities post the attack $\mathcal{C}(X_i + \delta)$ are as distinct as possible to the cluster probabilities M_i obtained before the attack. Then, assuming that prior to the attack the cluster labels were the correct representative of the ground truth label Y_i[4], the deep clustering model will miscluster the sample after the attack and performance (NMI, ACC, ARI) should reduce. If the adversary introduces many such adversarial samples, the performance can thus be significantly affected[5].

Our attack can be motivated via the following example: Recently, deep clustering has been successfully applied to the problem of Human Activity Recognition (HAR) where wearable sensors are used to record the user's data [141] and their current "activity" is predicted using the clustering model. While most research on HAR comprises of supervised approaches, often it is impossible to collect and annotate data for this task, prompting the use of clustering. Moreover, the target audience/users for HAR are generally elderly people [142] as it can be used to prevent unprecedented health risks (such as falls, etc.). Given that deep clustering is being utilized for this task, an informed adversary can use this to cause direct harm to the users if they wish to do so. If they can gain control (i.e., query the system) of the target's wearable device (through social engineering, security

[4]This is true for most samples in the dataset as clustering performance of the model prior to the attack is assumed to be superlative, otherwise there is no reason to attack.

[5]Note that this is an *untargeted* attack. One could also consider *targeted* attacks by replacing M_i in the objective with a k length vector where the target cluster entry is 1, and all other entries are 0.

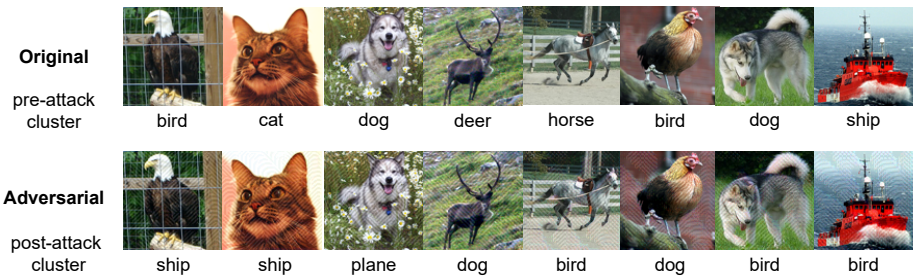


FIGURE 2.8. Adversarial samples generated by our attack (first 4 image pairs from the left correspond to SPICE and the others to RUC).

flaws, etc.), they can generate adversarial samples for it using our attack. In this manner, even if an individual needs immediate help, the attacker can use adversarial samples (which will likely be misclustered) to make the cause (activity) seem benign. The contrary case is also problematic– the adversary can generate adversarial samples for multiple devices that lead to "false alarms" solely for anarchistic purposes. Defending against such an attack would also be significantly challenging.

2.3.4. Proposed Attack. We utilize a simple GAN based architecture for generating the adversarial perturbation δ and solving the attack problem delineated in the previous section. There are many different variations to using GANs for generating adversarial samples, such as AdvGAN [143], AdvGAN++ [144], WPAdvGAN [145], CycleAdvGAN [46], among many others. For our attack, we employ a vanilla GAN architecture consisting of deep neural networks for both the Generator and Discriminator, similar to AdvGAN.

We utilize the Generator model $\mathcal{G}$ to generate the adversarial perturbation δ for a given input image $X_i \in X$, i.e., $\mathcal{G}(X_i) \rightarrow \delta$. The Discriminator model $\mathcal{D}$ plays a similar role as for the original GAN model [115] as it aims to ensure that the perturbed image is similar to the distribution of input images.

We then rewrite our attack optimization problem in the context of the GAN architecture. The loss for the attack objective can be written directly as in the optimization problem:

$$\mathcal{L}_{\text{attack}} := \mathbb{E}_{X_i} ||M_i - \mathcal{C}(X_i + \mathcal{G}(X_i))||$$

And we can simply reformulate the constraint on the norm of the adversarial noise as:

$$\mathcal{L}_{\text{constraint}} := \mathbb{E}_{X_i} \min\{\epsilon - ||\mathcal{G}(X_i)||, 0\}$$

We also write the vanilla minimax GAN loss [115] as:

$$\mathcal{L} := \mathbb{E}_{X_i}[\log(\mathcal{D}(X_i)) + \log(1 - \mathcal{D}(X_i + \mathcal{G}(X_i))]$$

To train the Generator $\mathcal{G}$ and Discriminator $\mathcal{D}$, we then optimize these combined losses by solving the following saddle-point problem (where α_a, α_c are hyperparameters to control tradeoff):

$$\max_{\mathcal{D}} \min_{\mathcal{G}} \{\mathcal{L} - \alpha_a \mathcal{L}_{\text{attack}} - \alpha_c \mathcal{L}_{\text{constraint}}\}$$

Upon obtaining the trained Generator $\mathcal{G}$, we can generate the adversarial perturbation as $\delta = \mathcal{G}(X_i)$ for any image $X_i \in X$ provided as input. We then provide these adversarial images $X_i + \delta$ as input to the pre-trained deep clustering model $\mathcal{C}$ to obtain cluster membership confidence scores as $M_i' = \mathcal{C}(X_i + \delta)$ after the attack. From M_i we know the original cluster label as $L_i = \text{argmax}_{j \in [k]} M_{i,j}$ and similarly, we can obtain the cluster label of the adversarial image as $L_i' = \text{argmax}_{j \in [k]} M_{i,j}'$.

If the optimization problem was solved successfully and the distance between M_i and $\mathcal{C}(X_i + \delta)$ is sufficiently large enough, we can have $L_i \neq L_i'$. Moreover, assuming that a mapping function ϕ exists that maps the output cluster labels to the ground truth labeling, we know that: $\phi(L_i) = Y_i$. Since $L_i \neq L_i'$, we can conclude that $\phi(L_i') \neq Y_i$ leading to misclustering and a drop in performance.

Remark. Note that since our attack objective is indirectly formulated, it is possible that M_i and M_i' are not sufficiently far apart even after the attack, leading to $L_i = L_i'$. However, as we find and our experiments show, this does not happen frequently in practice as our black-box attack is highly successful at generating adversarial samples that disrupt the performance of all deep clustering models considered. We have considered the following open-source deep clustering models in experiments: CC [52], SPICE [51], SCAN [146], MiCE [147], NNM [148], RUC [53]. Figure 2.8 shows a few adversarial samples generated via our attack for the SPICE and RUC models on the STL-10 dataset.

2.3.5. Possible Defenses. We also employ some possible defense approaches to better gauge the extent of our proposed attack. For this reason, we opt for two natural defenses that can be utilized to mitigate an attack against deep clustering models:

2.3.5.1. *In-processing Defense: Robust Deep Clustering.* Adversarially retrained models (where the learning process incorporates a joint adversarial loss along with the original loss) have been shown to considerably mitigate adversarial attacks, and thus constitute a natural in-processing defense approach. In the context of deep clustering, RUC and ALRDC utilize adversarial learning to improve model robustness. However, we only consider RUC in this work for the following reasons: 1) RUC is an add-on module, and can be applied to any deep clustering model, 2) in contrast, the ALRDC approach specifically works only to make the latent space robust to perturbations (which most deep clustering models considered in this work do not possess), restricting its use, and 3) unlike RUC, from our early experiments with ALRDC we found that it did not work well with high-dimensional real-world image datasets such as CIFAR-10/CIFAR-100/STL-10, which we have considered in this work[6].

In our experimental results, we show that using our GAN based attack and RUC as the deep clustering model $\mathcal{C}$, we can disrupt its performance significantly as well. Once the generator has learnt how to generate adversarial noise specific to RUC, it can easily degrade RUC's clustering ability. These results clearly indicate that there is a deficiency in current robust deep clustering strategies, and the problem requires different solutions[7].

2.3.5.2. *Pre-processing Defense: Deep Learning Based Anomaly Detection.* With advancements in deep learning, the long-standing field of anomaly/outlier detection has also seen major advancements. Further, due to the unsupervised nature of the deep clustering task (no labels), anomaly detection is a suitable pre-processing approach for detecting adversarial samples. In particular, most deep learning based anomaly detection models are trained on specific datasets and can detect out-of-distribution samples with extremely high precision. For our experiments, we use a recently proposed self-supervised deep anomaly detection approach SSD [150] which achieves state-of-the-art performance on benchmarks when labeled training data is not present [151].

[6]In the ALRDC paper only MNIST [127] and Fashion-MNIST [149] were considered in their experiments.

[7]One simple but compute-intensive solution could be to jointly minimize the deep clustering loss on adversarial inputs by using our trained adversarial noise generators, similar to traditional adversarial retraining.

While a detailed description of SSD is beyond the scope of our work, the authors employ contrastive self-supervised representation learning combined with a Mahalanobis distance based threshold detector in the feature space to detect anomalies. In our experiments later we show that even SSD is unable to detect a large majority of our adversarial samples. We further show that this is likely due to the distribution of our adversarial samples in space, as they mimic the original samples fairly well due to the norm constraint on the generator's output. We use Principal Component Analysis (PCA) [152] to analyze the adversarial and benign samples and find that their principal components tend to be very similar. In this regard, anomaly detection approaches are also unable to detect most of our generated adversarial samples (less than 1% in the best case; $\approx 14\%$ on average).

2.3.6. Results. We now present results for the GAN based attack, describe the general effects of the attack, and conduct a query complexity as well as a transferability analysis. Subsequently, we also provide results for the attack when the aforementioned defense approaches are employed.

For our experiments, we utilize the following real-world image datasets: CIFAR-10 [153], CIFAR-100 [154], and STL-10 [155]. These are commonly utilized in a majority of deep clustering literature. For the models, we consider a number of state-of-the-art deep clustering models: SPICE [51], SCAN [146], NNM [148], MiCE [147], and CC [52]. We also consider RUC [53], but discuss its results as part of the defense experiments, later. Notably, SPICE and RUC (add-on to SCAN) are SOTA #1 and #2 for performance on the aforementioned datasets. For additional experiment details refer to [12].

As part of our experiments, we generate adversarial images (adversarial set) using our attack for all the images in the original test set. We show the performance metrics (ACC, NMI, ARI), pre-attack (original images) and post-attack (adversarial images) for all the aforementioned models and datasets in Table 2.3. Quite evidently, it can be observed that performance for all models is significantly reduced post the attack. Note that since the GAN network generates fixed noise for the same input, there is no variance in the obtained results. For hyperparameter ($\alpha_a, \alpha_c, \epsilon$, etc.) values for all experiments please refer to the appendix in [12].

2.3.6.1. *Effects of the Attack.* For all the deep clustering models, we find a general trend emerges in how the models' post-attack clusters differ from the clusters generated on the original test set. In particular, performance on the benign images is generally very good– this can be observed in

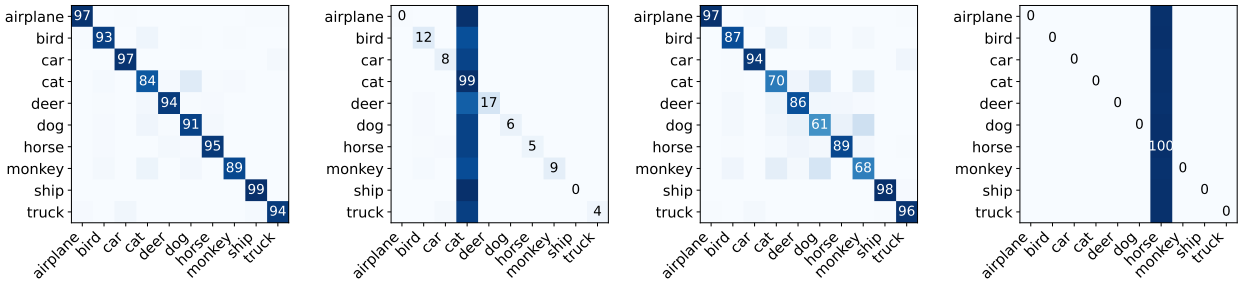
TABLE 2.3. Pre-attack and post-attack performance for deep clustering models.

Model		CIFAR-10			CIFAR-100			STL-10		
		NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI	ACC
CC	Pre-attack	0.70	0.64	0.79	0.43	0.27	0.43	0.76	0.73	0.85
	Post-attack	**0.01**	**0.00**	**0.10**	**0.03**	**0.00**	**0.07**	**0.03**	**0.00**	**0.12**
MiCE	Pre-attack	0.73	0.69	0.83	0.45	0.29	0.44	0.56	0.46	0.62
	Post-attack	**0.20**	**0.12**	**0.44**	**0.15**	**0.04**	**0.20**	**0.30**	**0.20**	**0.43**
NNM	Pre-attack	0.75	0.71	0.84	0.48	0.32	0.48	0.70	0.65	0.81
	Post-attack	**0.30**	**0.09**	**0.41**	**0.18**	**0.01**	**0.15**	**0.22**	**0.07**	**0.25**
SCAN	Pre-attack	0.71	0.66	0.82	0.49	0.33	0.51	0.67	0.62	0.79
	Post-attack	**0.27**	**0.06**	**0.48**	**0.10**	**0.01**	**0.14**	**0.16**	**0.02**	**0.22**
SPICE	Pre-attack	0.85	0.84	0.92	0.57	0.39	0.54	0.87	0.87	0.94
	Post-attack	**0.23**	**0.10**	**0.36**	**0.12**	**0.02**	**0.15**	**0.14**	**0.00**	**0.16**
RUC	Pre-attack	0.83	0.81	0.90	0.55	0.39	0.53	0.78	0.74	0.87
	Post-attack	**0.26**	**0.08**	**0.33**	**0.23**	**0.05**	**0.26**	**0.25**	**0.07**	**0.30**

TABLE 2.4. Query complexity of the attack.

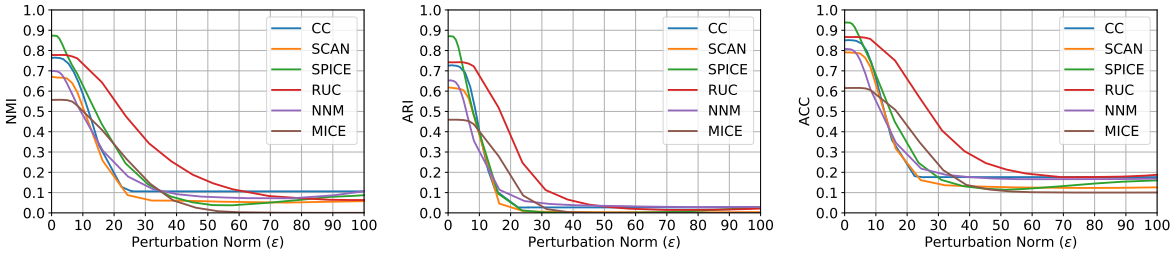
Model	CIFAR-10	CIFAR-100	STL-10
CC	5148	5382	2150
MiCE	2820	3142	2244
NNM	2320	11360	4660
SCAN	3100	2200	3080
RUC	3500	4000	3380
SPICE	2320	7520	2304

the confusion matrix for the SPICE model and STL-10 dataset in Figure 2.9a. Note that most of the clusters are correctly defined and there are a few miclusterings. However, for the adversarial images, we notice that there is a complete *clustering breakdown*– most images tend to get lumped in a small number of select clusters, and the remaining few images create small-sized clusters. This can also be seen in the confusion matrix in Figure 2.9b for SPICE on adversarial STL-10 images. Most of the adversarial images are clustered as part of the "cat" cluster, despite being clustered accurately before the attack. This trend is prevalent for the attack on all the models, and tends to be more drastic for less performant deep clustering models, such as CC. This can be seen in the confusion matrices shown in Figures 2.9c and 2.9d. For the confusion matrices for the other models please refer to the appendix in [12]. A major takeaway from these results is that while deep clustering models tend to work very well on "clean" data, simple adversarial samples exist that can completely degrade performance. It is thus imperative that deep clustering model designers evaluate their models against adversarial samples.



(a) Pre-attack (SPICE) (b) Post-attack (SPICE) (c) Pre-attack (CC) (d) Post-attack (CC)

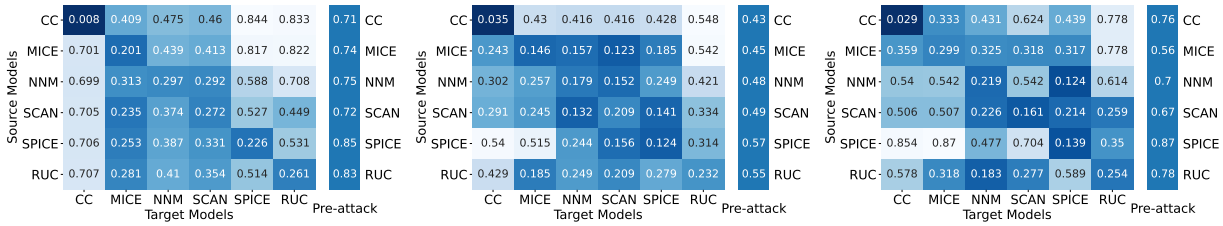
FIGURE 2.9. Confusion matrices showcasing the effect of the attack for the SPICE/CC models on STL-10.



(a) NMI (b) ARI (c) ACC

FIGURE 2.10. Performance versus adversarial perturbation norm (STL-10).

2.3.6.2. *Query Complexity and Perturbation Analysis.* We also measure *query complexity* of our approach to analyze the cost associated with carrying out our attack in the real world. In this black-box attack scenario, the query complexity is defined as the number of times the deep clustering model is queried by the adversary. Thus, we can measure the number of times the Generator $\mathcal{G}$ queries the clustering model $\mathcal{C}$ with an input batch, before the loss of the GAN network converges. We present these results in Table 2.4 where the batch size is 256. It can be seen that the query complexity of our attack is quite minimal for all models and datasets. In particular, the values obtained for our method are comparable to the query complexity rates obtained by existing black-box attacks against supervised learning models [137, 139]. Moreover, a smaller query complexity is doubly beneficial in our case, because once the generator has been trained, we can use it to generate the optimal perturbation for any images that belong to that input distribution without requiring any additional queries to the clustering model.



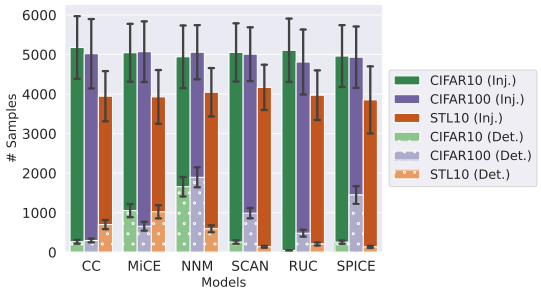
(a) CIFAR-10 (b) CIFAR-100 (c) STL-10

FIGURE 2.11. Transferability results showcasing post-attack (and pre-attack) NMI for different source/target models.

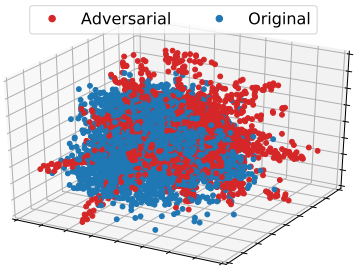
We also analyze the effect of varying the noise penalty (via ϵ) on the extent to which the attack degrades the performance of the deep clustering model. We depict the NMI, ARI, ACC for all the models on the STL-10 dataset as a function of the norm of the generated perturbation in Figure 2.10. It can be observed that as the norm threshold is increased the attack becomes more successful and the performance of the models is worsened, eventually plateauing close to 0.

2.3.6.3. *Transferability Analysis.* We undertake a transferability analysis to see whether adversarial samples generated for one model transfer to other deep clustering models. We present these results in Figure 2.11 for all our datasets as transferability matrices. In these, we show the post-attack NMI for source and target models, along with the pre-attack NMI for each of the deep clustering models. It can be seen that the overall transferability of most adversarial samples is high. Note that SPICE (SOTA #1) is in general a much better deep clustering model than CC in terms of performance. Interestingly however, we can see that SPICE's adversarial samples do not transfer well to CC and vice versa. It would be interesting to investigate the reasons for this occurrence in future theoretical or empirical work.

2.3.6.4. *Utilizing Robust Deep Clustering for Defense.* As mentioned before, we utilize the RUC add-on module as a first defense against our attack. As in the original paper, we utilize RUC with SCAN as the deep clustering model and then carry out our GAN based attack against it. Similar to previous experiments on the models, we measure the performance on a benign test set and then on a corresponding adversarial set generated using our GAN attack. We find that our attack is also successful against RUC, thus proving that RUC is not entirely robust to adversarial noise. This is shown in Table 2.3. It can be seen that RUC is not a satisfactory defense approach for our attack. However, note that the robustness of RUC is somewhat observable in our perturbation norm versus



(a) SSD is unable to detect a large majority of our attack samples (Injected → Inj. and Detected → Det.).



(b) Visualizing the PCA components of original samples (blue) and their adversarial counterparts (red) computed for the SPICE model.

FIGURE 2.12. Results on using anomaly detection as a possible defense.

performance analysis shown in Fig 2.10. In general, it takes higher norm thresholds to reduce the performance of RUC, although it is achievable without adding too much noise to the images. For the transferability analysis shown in Fig 2.11, we find RUC behaves similar to SPICE, as RUC's adversarial samples are highly transferable across all models and datasets except for CC on the CIFAR-10 and CIFAR-100 datasets.

2.3.6.5. *Utilizing Deep Learning Based Anomaly Detection for Defense.* We now present results for SSD, the deep learning based anomaly detection approach, as a pre-processing defense to our attack. In our experiments, we start with a test set of 8000 images for all of our datasets. For each benign image X_i, we uniformly randomly either add adversarial noise to it or let it remain benign. We do this over 100 trials and average our results to observe how many adversarial samples were detected by the anomaly detection approach. We present these results in Fig 2.12a. We find that on average the method detects 11.69%, 19.41%, and 11.71%, of adversarial samples for the CIFAR-10, CIFAR-100, and STL-10 datasets, respectively. However, this is negligible compared to the large number of adversarial samples being injected and remaining undetected. To analyze the reasons for this occurrence, we compute the first 3 principal components of the adversarial set and the benign set using PCA and present these in Fig 2.12b for SPICE. Visually, the principal components of a large majority of adversarial samples are superimposed and interspersed amongst those of the benign samples, showcasing the potency of our attack.

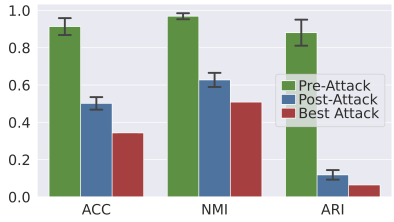


FIGURE 2.13. NMI/ACC/ARI before and after the attack on `Face++` API.

2.4. Attacking `Face++`: A Production-Level MLaaS Clustering Service

To showcase the disruptive capability of our attacks, we attack `Face++`, which provides an extremely well-performing face clustering REST API service. At a high-level, the API performs a basic face clustering task where it takes in as input a face dataset and seeks to cluster images belonging to the same person together. We do not know the clustering algorithm/model being utilized in the backend. Note that there is one significant difference here compared to the threat model for open-source deep clustering models previously considered– we do not have access to cluster memberships, but just the final set of labels. To overcome this problem, we attack `Face++` by training a *surrogate* open-source deep clustering model on the dataset and then generate adversarial samples for this model using our GAN based attack pipeline. We then use these adversarial samples as input to the `Face++` API to conduct an evasion attack. Hence, our attack constitutes a *transferability* attack via a *surrogate* model.

The `Face++` API service functions as follows: we first create a new *face album* using the `createAlbum` endpoint which gives us a `facealbum_token` as a response. Using this token, we add the images for our dataset using the `addimage` endpoint, and then call the `groupFace` endpoint to perform the clustering task. Finally, we obtain cluster labels for each image using the `getAlbumDetail` endpoint, which returns `group_ids` as the integer cluster labels.

We now present our results for attacking `Face++`, a production-level face clustering API service. Since the service only works with face images, we use the Extended Yale Face B [156] dataset for these experiments. There are 28 persons in this dataset, i.e., $k = 28$ and 500 images for each person. However, due to rate limits and the latency associated with uploading images using REST APIs, we test the service by sending only 10 images per person, for a total of 280 images. Then, as only cluster labels are outputted by the API, we resort to using a *surrogate model* for the attack via

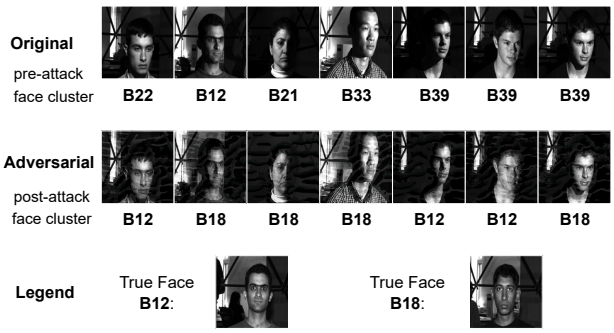


FIGURE 2.14. Adversarial samples for the `Face++` attack.

transferability. We train a CC model as the surrogate on the entire Extended Yale Face B dataset and then train our GAN network to generate the adversarial noise for a given image from the dataset. Thus, we generate adversarial counterparts to the original 280 benign images and carry out the attack. Subsequently, we observe the NMI/ACC/ARI pre-attack and post-attack.

Since the results obtained can be affected by which 10 representative images were picked for each person in the test set, we randomize this selection process and take the average over 10 runs. As can be seen in Figure 2.13, before the attack, the `Face++` clustering API boasts stellar performance on the test set, with average NMI ≈ 0.97. However, for the set of adversarial images, performance of the API service degrades significantly, with the NMI dropping down to ≈ 0.62 on average. This shows that our generated adversarial examples can disrupt the working of a high-performance real-world service, even when we cannot query the model for memberships. We also show some of the adversarial samples generated from original images, and how they have been misclustered in Figure 2.14. Before the attack, most of the faces are clustered correctly, but after the attack they only majorly belong to the `B12` or `B18` face clusters. Here too, we are observing the *clustering breakdown* effect previously noted for the attack against open-source deep clustering models. For more details regarding implementation and results, please refer to [12].

CHAPTER 3

Fairness and Social Robustness in the Context of Machine Learning

In this chapter, we will explore fairness in ML by delving into two problems: fair clustering using antidote data, and fair video summarization.

3.1. Fair Clustering Using Antidote Data

3.1.1. Introduction. With the increasing application of machine learning (ML) algorithms in modern society, the design of fair variants to traditional ML algorithms is an important concern. Vanilla ML algorithms do not account for the biases present in training data against certain *minority protected groups*, and hence, might reinforce them. Furthermore, clustering has been widely used to find meaningful structures, explanatory underlying processes, generative features, and groupings inherent in a set of examples. It plays a significant role in most modern data science applications, such as in medicine [157], vision [158], language modeling [159], financial decisions [160], and various societal resource allocation problems. Thus, ensuring fairness with respect to protected groups is an important issue for clustering algorithms.

Currently, many different *group-level* notions for fairness in clustering exist, such as *balance* [85], *proportionality* [161], *social fairness* [104], among others. Traditionally, to make clustering outputs fair with respect to a specific notion of fairness, fair variants to clustering algorithms need to be proposed. Given that many different clustering algorithms exist, each fair variant proposed requires individual analysis, and possesses different theoretical guarantees. Moreover, if fairness notions or clustering algorithms are changed in a deployed real-world system, the corresponding fair algorithms would also have to be reimplemented. Therefore, instead of coming up with new fair algorithms for each fairness definition and each clustering algorithm, we propose an alternate approach to ensuring fairness for clustering. Inspired by recent research on *adversarial attacks* and *data poisoning*, we aim to augment the dataset with *antidote* data points such that when we use vanilla clustering on this new combined dataset, fairness constraints are met. Thus, instead of

changing the clustering algorithm to ensure fairness, we *find* an augmented dataset for which the specified fairness constraints are met when vanilla clustering is undertaken on it. Our approach is therefore applicable in very general case scenarios where group-level fairness on the original dataset can be achieved for any arbitrary choice of center-based clustering algorithm and fairness definition. Note that we aim to make clustering fair in the *pre-clustering* stage as opposed to the *in-clustering* stage, unlike most research on fair clustering.

Data augmentation to improve fairness was first proposed by [18] for recommendation systems. The authors coined the term *antidote* data for the data points added to the original dataset. However, since recommendation systems and clustering algorithms differ widely, their problem formulation and techniques do not translate to clustering. The antidote data problem for clustering is then as follows: *given a dataset U, can we compute (antidote) data V such that when we cluster on $U \cup V$ we obtain a fair clustering output for a chosen fairness notion and clustering algorithm?*

We answer this question in the affirmative by proposing a general bi-level formulation of the antidote data problem for clustering. We also cannot reuse existing approaches for adversarial attacks on clustering algorithms as our bi-level formulation requires the antidote data addition to lead to very specific clustering outcomes that improve fairness irrespective of where points lie in clusters.

3.1.2. Problem Statement. The original dataset is denoted as $U \in \mathbb{R}^{n \times d}$. This is the dataset we wish to augment with some antidote data points such that certain fairness constraints are met when we cluster on the augmented dataset. Furthermore for a matrix M, let M_i and M^i denote the i-th row and i-th column respectively. To start, we first define the clustering problem on U. A center-based clustering objective, $\mathcal{C}$, takes in a dataset as input (such as U) and outputs a set of k centers $\mu \in \mathbb{R}^{k \times d}$, where $k \leq n$. That is, a clustering objective induces a k-partition set of the data, where each sample in the dataset is uniquely mapped to a center $\mu_i \in \mu$ where $\mu \in \mathbb{R}^d$. For example, the k-means clustering objective on U can be defined as $\mathcal{C}_{\text{k-means}}(U) \coloneqq \mu = \text{argmin}_{\mu' \in \mathbb{R}^{k \times d}} \sum_{x \in U} \min_{i \in [k]} ||x - \mu'_i||^2$.

We denote the group-level fairness notion as $\mathcal{F} : (\mu, U) \to \mathbb{R}$. That is, the fairness notion takes as input the set of centers from a clustering algorithm and the original dataset, and outputs a fairness cost. The goal of improving fairness is to then minimize $\mathcal{F}$. It is important to note that fairness will

be evaluated only on the original real dataset U. Moreover, as we will see, all group-level fairness notions can be defined this way.

The General Problem. We now state the antidote data problem for improving fairness. We aim to add a set of data points V to U, such that when we cluster on $U \cup V$ and obtain centers μ, $\mathcal{F}(\mu, U)$ is less than some given value α. The cost of adding points can be defined as the size of set V, and hence, we aim to add as few points as possible. The general bi-level optimization problem is as follows:

$$\text{(P1)} \qquad \begin{aligned} \min_{V,\,\mu} \quad & |V| \\ \text{s.t.} \quad & \mathcal{F}(\mu, U) \leq \alpha \\ & \mu = \mathcal{C}(U \cup V) \end{aligned}$$

Relaxation P1.R. We also consider a relaxed formulation of problem P1. This relaxation allows us to propose algorithms that in turn also solve problem P1 indirectly. The idea is to fix the size of the antidote dataset $|V| \leq \overline{V}_s$ for a given $\overline{V}_s \in \mathbb{R}$, and optimize the fixed-set V so that we only minimize $\mathcal{F}$ in the upper-level problem. Since minimizing the fairness cost is now the upper-level objective, we can also omit writing it as a constraint using α:

$$\text{(P1.R)} \qquad \begin{aligned} \min_{V,\,\mu} \quad & \mathcal{F}(\mu, U) \\ \text{s.t.} \quad & \mu = \mathcal{C}(U \cup V) \\ & |V| \leq \overline{V}_s \end{aligned}$$

We now define the group-level fairness costs we use. Consider some $g \in \mathbb{Z}^+$ number of protected groups that comprise U. Each protected group has an index $j \in [g]$ and contains a certain number of points of U. For simplicity of notation we also assume that a mapping function $\psi(U, j)$ exists which takes in as input U and an integer j, where $1 \leq j \leq g$, and gives us the set of points of U which belong to the protected group j. Now we can define the *social fairness* cost of Ghadiri et al [104]. This was originally proposed for k-means clustering, but it fits well with any center-based clustering objective where Euclidean distance is used as the clustering distance metric.

DEFINITION 3.1.1. (***Social Fairness*** [104]). *Let* $\Delta(\mu, U) = \sum_{x \in U} \min_{\mu_i \in \mu} ||x - \mu_i||^2$ *where* U *is the original dataset and* μ *are cluster centers. Then the social fairness cost is defined as:*

$$\mathcal{F}_{\text{social}}(\mu, U) = \max_{j \in [g]} \left\{ \frac{\Delta(\mu, \psi(U, j))}{|\psi(U, j)|} \right\}$$

Next we define the *balance* metric [85, 98]. Traditionally, *balance* is a fairness metric that is not a cost, and is maximized. To fit within our framework, we frame it as a cost by multiplying it with -1, and name it the *balance cost*. Again, for simplicity of notation, we assume a mapping function $\phi(U, \mu, i)$ exists which takes in as input U, μ, and a cluster label $i \in [k]$ and gives us the points in U which belong to cluster i. Note that obtaining cluster labels is trivial as for each $x \in U$ the corresponding label can be obtained as $i = \text{argmin}_{i' \in [k]} ||x - \mu_{i'}||$.

DEFINITION 3.1.2. (***Balance Cost*** [98]). *Let* U *be the original dataset and* $\mu \in \mathbb{R}^{k \times d}$ *be the set of cluster centers. Define the following ratio* $R(i, j) = \frac{|\psi(U,j)|/|U|}{|\psi(U,j) \cap \phi(U,\mu,i)|/|\phi(U,\mu,i)|}$ *which signifies the ratio between the proportion of points of group* j *in* U *and proportion of group* j *points in cluster* i*. The balance cost* $\mathcal{F}_{balance} \in [-1, 0]$ *is then defined:*

$$\mathcal{F}_{\text{balance}}(\mu, U) = - \min_{i \in [k], j \in [g]} \left\{ \min \left\{ R(i, j), \frac{1}{R(i, j)} \right\} \right\}$$

3.1.3. Proposed Approaches. We consider problem P1 under 2 different settings and provide algorithms and analysis for each: (1) Convex $\mathcal{C}$ and Convex $\mathcal{F}$, and (2) General $\mathcal{C}$ and General $\mathcal{F}$. While setting (1) comprises more of a toy problem as clustering objectives used in practice are rarely convex, solving problem P1 for setting (2) is quite challenging. For the first setting with convex functions, we can reduce the bi-level problem to a single-level optimization, allowing us to utilize off-the-shelf solvers to obtain V. For the general setting, the antidote data problem is significantly harder and we resort to using zeroth-order optimizers as part of our proposed solution to finding a feasible V.

3.1.3.1. *Warmup: Convex* $\mathcal{C}$ *and Convex* $\mathcal{F}$*.* For this setting, we assume that both $\mathcal{C}$ and $\mathcal{F}$ are convex functions. Assuming convexity allows us to effectively reduce the bi-level problem to a single-level form, which can then be provided to off-the-shelf convex/non-convex solvers for optimization. In particular, we exploit the convexity of the functions by replacing the lower-level

problem with its Karush-Kuhn-Tucker (KKT) optimality conditions as constraints for the upper-level problem. Since the lower-level clustering problem is convex, the KKT conditions are necessary and sufficient to ensure optimality [162].

As optimizing bi-level problems is in general NP-Hard [163], and problem P1 contains an NP-Hard cardinality minimization problem [164] as the upper-level objective, we use the relaxed form P1.R to indirectly solve P1. This involves fixing $|V|$ as an input hyperparameter and optimizing V so as to minimize $\mathcal{F}$, without considering α. We then use the convexity of the lower-level problem to obtain a single-level reduction from this bi-level problem by replacing the lower-level problem with its KKT constraints. When we minimize this reduced single-level problem, we effectively minimize P1.R.

We describe our approach as Algorithm 2. We aim to solve problem P1.R using our algorithm, and in each iteration try to find a suitable V to optimize using the reduced single-level problem (obtained via KKT conditions). In each iteration of the algorithm, we start by fixing the size of V to some V_s, and obtain $\mathcal{F}$ after optimizing V. If this fairness cost is less than α, we can exit, otherwise we increase the size of V (denoted as V_s) by $\xi \in \mathbb{Z}^+$ for the next iteration and continue. Algorithm 2 can also exit if the constraint is not met, if a certain number of iterations are exceeded, or if $|V|$ grows to an unacceptable value. We omit these details from Algorithm 2 for simplicity, but they can be easily implemented.

Not many widely used convex formulations for clustering algorithms exist except for sum-of-norms (SON) clustering [165, 166], which is strongly convex. SON clustering has been shown to be a convex relaxation to both k-means clustering [165] and hierarchical agglomerative clustering [166]. Below, we analyze SON clustering in the context of Algorithm 2. For the fairness notion, we utilize $\mathcal{F}_{\text{social}}$ which is clearly convex and well-defined for SON clustering. We first define the SON clustering objective. It is important to note that we modify the notation– since the objective is convex, the number of clusters are not discretely defined, but obtained via a regularization parameter λ. Centers are represented as a $\mathbb{R}^{n\times d}$ matrix as there is no explicitly defined k, but note there will only be some unique $k \leq n$ centers decided by the parameterization of λ. The objective is as follows: $\mathcal{C}_{\text{SON}}(U) \coloneqq \mu = \text{argmin}_{\mu' \in \mathbb{R}^{n\times d}} \frac{1}{2} \sum_{j=1}^{n} ||U_j - \mu'_j||^2 + \lambda \sum_{i<j} ||\mu'_i - \mu'_j||$.

Let $V_s^{(t)}$ denote the size V_s of V in iteration t of Algorithm 2 (line 2). The number of centers we have will be $\mu \in \mathbb{R}^{m\times d}$ where $m = n + V_s^{(t)}$ for $U \cup V$. To derive the KKT conditions we first

reformulate the objective. Consider an ordering of all (μ_i, μ_j) pairs where all $i < j$. We can let each of the m centers μ_i be a node in a graph G. The created ordering essentially enumerates the list of edges E for the graph G. We denote this ordering as O where we will have $|E| = |O| = m(m-1)/2$. We also denote the node-arc-incidence matrix [167] for (G, E) as $I \in \mathbb{R}^{m \times |O|}$. We can then rewrite the SON objective, define the dual problem to the reformulation, and derive the KKT conditions. We provide more detail on these next.

Consider the original strongly convex SON clustering objective:

$$\min_{\mu' \in \mathbb{R}^{n \times d}} \frac{1}{2} \sum_{j=1}^{n} ||U_j - \mu'_j||^2 + \lambda \sum_{i<j} ||\mu'_i - \mu'_j|| \tag{3.1}$$

we create the ordering O, the graph G and define its node-arc-incidence matrix I [167] and then reformulate the above objective:

$$\min_{\mu \in \mathbb{R}^{n \times d},\, \eta \in \mathbb{R}^{d \times |O|}} \quad \frac{1}{2}||\mu - U||^2 + \lambda \sum_{i \in O} ||\eta^i|| \quad \text{s.t.} \quad \mu^T I - \eta = 0 \tag{3.2}$$

It can be verified that the above objectives are equivalent. We can even define the dual formulation for the above primal problem (where $\langle , \rangle$ denotes the matrix Frobenius inner-product):

$$\begin{aligned} \max_{\theta \in \mathbb{R}^{n \times d},\, \zeta \in \mathbb{R}^{d \times |O|}} \quad & \langle U^T, \theta \rangle - \frac{1}{2}||\theta||^2 \\ \text{s.t.} \quad & I\zeta^T - \theta = 0 \\ & ||\zeta_i|| \leq \lambda, \forall i \in O \end{aligned} \tag{3.3}$$

Now, we discuss the KKT conditions. Since the SON objective is strongly convex, we can use the reformulated primal and dual problems to arrive at the KKT conditions:

$$\begin{aligned} & \theta + \mu - U = 0 \\ & \eta - \mathcal{P}(\eta + \zeta) = 0 \\ & \mu^T I - \eta = 0 \\ & I\zeta^T - \theta = 0 \end{aligned} \tag{3.4}$$

Here $\mathcal{P}(.)$ refers to the proximal operator of the Euclidean norm, therefore $\mathcal{P}(\eta+\zeta) = \max\{0, 1 - \frac{1}{||\eta+\zeta||}\}(\eta+\zeta)$. Since we now have the KKT conditions we can undertake the single-level reduction for problem P1.R.

For this we first have to substitute U for $U \cup V$. So now we can use the KKT conditions by replacing U with $U \cup V$ and n with m. The other variables will also then be: $\mu \in \mathbb{R}^{m\times d}, \eta \in \mathbb{R}^{d\times|O|}, \theta \in \mathbb{R}^{m\times d}, \zeta \in \mathbb{R}^{d\times|O|}$. The original problem P1.R for $\mathcal{C}_{\text{SON}}$ and $\mathcal{F}_{\text{social}}$ is:

$$\begin{aligned} \min_{V,\mu} \quad & \mathcal{F}_{\text{social}}(\mu, U) \\ \text{s.t.} \quad & \mu = \mathcal{C}_{\text{SON}}(U \cup V) \end{aligned} \tag{3.5}$$

Replacing equations 3.4 as constraints for the upper-level objective and removing the lower-level objective gives us the single-level optimization problem:

$$\begin{aligned} \min_{V,\mu,\eta,\theta,\zeta} \quad & \mathcal{F}_{\text{social}}(\mu, U) \\ \text{s.t.} \quad & \theta + \mu - (U \cup V) = 0 \\ & \eta - \max\{0, 1 - \frac{1}{||\eta+\zeta||}\}(\eta+\zeta) = 0 \\ & \mu^T I - \eta = 0 \\ & I\zeta^T - \theta = 0 \end{aligned}$$

Here, $\mu \in \mathbb{R}^{m\times d}, \eta \in \mathbb{R}^{d\times|O|}$ are the primal variables, and $\theta \in \mathbb{R}^{m\times d}$, $\zeta \in \mathbb{R}^{d\times|O|}$ are the dual variables. We also observe that replacing KKT conditions as constraints can introduce non-convexity. All the constraints and objectives are convex, except for one: $\eta - \max\{0, 1 - (1/||\eta+\zeta||)\}(\eta+\zeta) = 0$. To approximate this, we can replace it with an affine constraint as $\eta - \gamma(\eta+\zeta) = 0$ where $0 \leq \gamma \leq 1$. Then a convex solver such as CVX [168] can be used to solve the above problem. Finally, assuming it takes time T_{KKT} to solve the single-level problem, and a feasible antidote dataset V^* exists, Algorithm 2 has a running time of $\mathcal{O}(T_{\text{KKT}}|V^*|/\xi)$.

3.1.3.2. *General $\mathcal{C}$ and General $\mathcal{F}$.* In this setting, we make no assumptions about the clustering objective $\mathcal{C}$ and the fairness cost $\mathcal{F}$. In such a minimal assumption setting where group-level fairness notions as well as center-based clustering objectives can vary widely, it is not trivial to propose

Algorithm 2 Convex $\mathcal{C}$ and $\mathcal{F}$

Input: $U, \mathcal{C}, \mathcal{F}, V_s, \xi$
Output: V
while true **do**
  initialize V arbitrarily with $|V| = V_s$
  reduce problem P1.R by replacing $\mathcal{C}(U \cup V)$ with its KKT conditions as constraints
  solve this single-level problem for optimal V
  if $\mathcal{F}(\mu, U) \leq \alpha$ **return** V **else** $V_s \leftarrow V_s + \xi$
end while

Algorithm 3 General $\mathcal{C}$ and $\mathcal{F}$

Input: $U, \mathcal{C}, \mathcal{F}, \mathcal{A}, V_s, n', \xi$
Output: V
while true **do**
  define $\mu \leftarrow \mathcal{C}(U \cup V)$ and $f(V) \leftarrow \mathcal{F}(\mu, U)$
  initialize V arbitrarily with $|V| = V_s$
  optimize V using $\texttt{SRE}(n', f(V), \mathcal{A})$
  obtain optimized V and $\mathcal{F}(\mu, U)$ from SRE & $\mathcal{A}$
  if $\mathcal{F}(\mu, U) \leq \alpha$ **return** V **else** $V_s \leftarrow V_s + \xi$
end while

algorithms with strong theoretical guarantees. Furthermore, some of the most popular and widely utilized clustering algorithms such as k-means, hierarchical clustering, DBSCAN, etc. possess highly non-convex objectives and are generally optimized via heuristic algorithms (such as Lloyd's algorithm [78] for k-means). In terms of fairness notions for clustering, *balance* is generally the most widely used metric in proposing fair algorithms. As evident in Definition 3.1.2, it is both non-convex and non-differentiable.

Furthermore, general bi-level optimization is NP-Hard; even for the simpler case when the upper-level and lower-level problems are linear, a polynomial time algorithm that finds the global optima of the bi-level problem might not exist [163]. Since we are dealing with possibly many non-convex upper-level and lower-level problems in this setting, finding a global optima for P1 is not a trivial task. We then resort to finding a locally optimal solution that satisfies our problem constraints. To do this, we relax the NP-Hard upper-level problem which seeks to minimize the size of the antidote dataset V. Similar to the convex setting, we are attempting to solve the relaxed formulation P1.R (indirectly solving P1), where we fix $|V|$ to some given value, and optimize V to minimize $\mathcal{F}$.

To solve P1.R, we can use zeroth-order optimization algorithms (such as `RACOS` [169], `CMAES` [170], `IMGPO` [171]). Let such an algorithm be denoted as $\mathcal{A}$. Most zeroth-order optimization algorithms do not scale well with problem input, and hence, cannot usually be applied to data with number of samples $n \geq 1000$ [172]. However, since our goal is to utilize antidote data on large-scale datasets, the algorithm $\mathcal{A}$ cannot be applied directly to solve P1.R in practice. To circumvent this problem, we propose using the Sequential Random Embedding (`SRE`) approach of [172], which can be used in conjunction with the zeroth-order black-box optimizer $\mathcal{A}$ to solve P1.R. The `SRE` approach scales the problem input by projecting it to a low-dimensional setting where it invokes $\mathcal{A}$ to solve the optimization. `SRE` takes in as input the reduced dimension $n' \ll n$, the objective function f to optimize, and zeroth-order optimization algorithm $\mathcal{A}$. We defer the reader to [172] for more details on `SRE`. In our experiments for this setting, we use `RACOS` [169] as the algorithm $\mathcal{A}$, which is a *Sampling-and-Learning* (SAL) framework.

Using the `SRE` approach, we propose Algorithm 3 for solving P1.R. We begin by defining the nested function f to optimize (line 2) which takes in as input some V and outputs the fairness cost $\mathcal{F}(\mu, U)$ where μ is obtained via $\mathcal{C}(U \cup V)$. The basic idea is to fix $|V|$ to some pre-defined starting value V_s and optimize V using the `SRE` approach as the back-end (line 3-5). Then, if the constraint $\mathcal{F}(\mu, U) \leq \alpha$ is not met, we increase $|V|$ by some small number $\xi \in \mathbb{Z}^+$ and repeat (line 6). Similar to Algorithm 2, we can exit in the while loop after a certain number of iterations or if $|V| \gg |U|$.

3.1.4. Results. We consider four real-world datasets commonly used to evaluate fair clustering algorithms: `adult` [173], `bank` [174], `creditcard` [175], and Labeled Faces in the Wild (`LFW`) [176]. The `adult` dataset has 10000×5 samples, and protected groups signify *race* (*white, black, asian-pac-islander, amer-indian-eskimo, other*). The `bank` dataset has 45211×3 samples, and protected groups signify *marital status* (*married, single, divorced*). The `creditcard` dataset has 30000×23 samples, and the protected groups signify *education* (*higher* and *lower* education). `LFW` has 13232×80 samples, and the protected groups signify *sex* (*male, female*).

Since Algorithm 2 is solving a simple convex optimization problem, we omit its results here. For the results, refer to the appendix in [15]. We compare Algorithm 3 against vanilla clustering and state-of-the-art fair clustering algorithms and let $k = 2$. We also compare Algorithm 3 and other fair clustering approaches in terms of clustering performance, using clustering performance metrics such

TABLE 3.1. Comparing fairness costs of Algorithm 3 with vanilla clustering. (Consider Combination #1 and the `bank` dataset as an example. The fairness cost for the vanilla cluster centers μ^{vanilla} is $\mathcal{F}(\mu^{\text{vanilla}}, U) = -0.3054$ and α is set to this value to improve on this fairness cost. After Algorithm 3 is run, V is obtained, with size $|V| = 0.00011|U|$. Cluster centers μ obtained by clustering on $U \cup V$ result in fairness cost $\mathcal{F}(\mu, U) = -0.3077$. This is lower than $\mathcal{F}(\mu^{\text{vanilla}}, U)$, leading to improved fairness.)

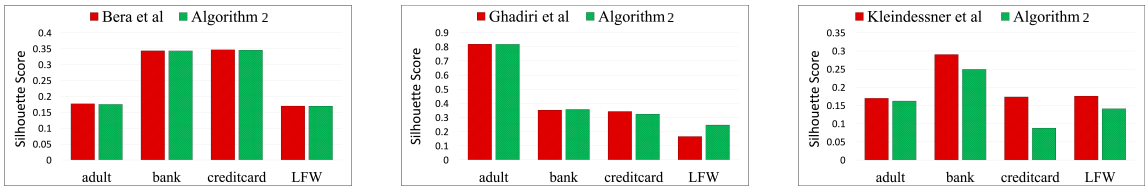
Clustering-Fairness Combination	Dataset	α	$\lvert V\rvert/\lvert U\rvert$	$\mathcal{F}(\mu^{\text{vanilla}}, U)$	$\mathcal{F}(\mu, U)$
Combination #1: $\mathcal{C}_{\text{k-means}}, \mathcal{F}_{\text{balance}}$	`adult`	-0.6119	0.001	-0.6119	**-0.6196**
	`bank`	-0.3054	0.00011	-0.3054	**-0.3077**
	`creditcard`	-0.8696	0.00017	-0.8696	**-0.8715**
	`LFW`	-0.8815	0.00075	-0.8815	**-0.8821**
Combination #2: $\mathcal{C}_{\text{k-means}}, \mathcal{F}_{\text{social}}$	`adult`	5.3678	0.0005	5.3678	**4.2104**
	`bank`	2.3432	0.00022	2.3432	**2.3416**
	`creditcard`	19.740	0.00034	19.740	**19.729**
	`LFW`	1406.3411	0.00076	1406.3411	**1406.1676**
Combination #3: $\mathcal{C}_{\text{spectral}}, \mathcal{F}_{\text{balance}}$	`adult`	-0.6458	0.001	-0.6458	**-0.6911**
	`bank`	-0.4811	0.00022	-0.4811	**-0.5489**
	`creditcard`	-0.8384	0.00034	-0.8384	**-0.8407**
	`LFW`	-0.9279	0.00076	-0.9279	**-0.9389**

TABLE 3.2. Comparing fairness costs of Algorithm 3 with fair clustering algorithms. (Reads similarly to Table 3.1.)

Clustering-Fairness Combination	Dataset	α	$\lvert V\rvert/\lvert U\rvert$	$\mathcal{F}(\mu^{\text{SOTA}}, U)$	$\mathcal{F}(\mu, U)$
Combination #1: $\mathcal{C}_{\text{k-means}}, \mathcal{F}_{\text{balance}}$	`adult`	-0.6059	0.001	-0.6059	**-0.6196**
	`bank`	-0.3065	0.00011	-0.3065	**-0.3077**
	`creditcard`	-0.8696	0.00017	-0.8696	**-0.8715**
	`LFW`	-0.8816	0.00075	-0.8816	**-0.8821**
Combination #2: $\mathcal{C}_{\text{k-means}}, \mathcal{F}_{\text{social}}$	`adult`	4.2636	0.0005	4.2636	**4.2104**
	`bank`	2.3135	0.1549	2.3135	**2.3119**
	`creditcard`	18.998	0.19	**18.998**	**18.998**
	`LFW`	1344.5468	0.3999	1344.5468	**1344.5461**
Combination #3: $\mathcal{C}_{\text{spectral}}, \mathcal{F}_{\text{balance}}$	`adult`	-0.5973	0.001	-0.5973	**-0.6911**
	`bank`	-0.6086	0.5	-0.6086	**-0.6899**
	`creditcard`	-0.8407	0.38	-0.8407	**-0.9990**
	`LFW`	-0.9926	0.4	-0.9926	**-0.9997**

as the Silhouette coefficient [177], Calinski-Harabasz score [178], and the Davies-Bouldin index [179]. We use these metrics to unify comparisons across the different clustering algorithms considered in experiments. For all experiments, we choose α to be the fairness cost of the algorithms being compared against (vanilla clustering, fair algorithms) so as to improve on them. We let $\mathcal{A}$ be the `RACOS` [169] algorithm, $V_s = 10, n' = 100, \xi = 1$.

3.1.4.1. *Comparison with Vanilla and Fair Clustering Approaches.* Since Algorithm 3 can accommodate general $\mathcal{C}$ and $\mathcal{F}$, we experiment on 3 combinations: Combination #1 with $\mathcal{C}_{\text{k-means}}$ and $\mathcal{F}_{\text{balance}}$, Combination #2 with $\mathcal{C}_{\text{k-means}}$ and $\mathcal{F}_{\text{social}}$, and Combination #3 where $\mathcal{C}$ is unnormalized spectral clustering, and $\mathcal{F}$ is $\mathcal{F}_{\text{balance}}$. The results when comparing against vanilla clustering are



(a) Combination #1 (b) Combination #2 (c) Combination #3

FIGURE 3.1. Comparing clustering performance of Algorithm 3 with fair clustering algorithms using Silhouette scores. (Higher scores indicate better clustering performance. As can be observed, fair clusters obtained via Algorithm 3 achieve similar clustering performance to SOTA algorithms, while providing improved fairness.)

shown in Table 3.1. Vanilla cluster centers are denoted as μ^{vanilla} and centers obtained via Algorithm 3 are denoted by μ. As can be seen we add very few antidote data points ($|V|/|U|$) and improve on the fairness cost over vanilla clustering. For each of the combination settings considered, we also compare against an equivalent state-of-the-art fair clustering algorithm. For Combination #1 we consider the algorithm of Bera et al [98], for Combination #2 we consider the Fair-Lloyd algorithm of Ghadiri et al [104], and for Combination #3 we consider the algorithm of Kleindessner et al [103]. Since the approach of [103] cannot handle large datasets, we subsample each dataset to 1000 samples for Combination #3. The results are shown in Table 3.2, and centers obtained from fair clustering algorithms are denoted as μ^{SOTA}. We find that we outperform fair algorithms in terms of lower fairness costs.

3.1.4.2. *Clustering Performance Comparison.* For comparison, we use the widely utilized Silhouette score [177] which lies between $[-1, 1]$, with higher scores indicating better clustering performance. We show the results in Figure 3.1 for each combination setting considered. The fair clusters of Algorithm 3 used here are the same from Table 3.1. We observe that despite outperforming fair algorithms in terms of fairness, we still exhibit competitive clustering performance.

3.2. Fair Video Summarization

3.2.1. Introduction. With the rapid growth of video content on the internet, there is an increasing need to automatically summarize lengthy videos to provide users with a condensed version that contains the most important information. This has led to the machine learning (ML) vision task of automated video summarization, which entails generating a short, representative

summary video (comprised of key-frames) of a longer input video that showcases its main content and events. In recent years, deep learning (DL) based models have achieved the state-of-the-art (SOTA) in video summarization by leveraging powerful feature representations and learning complex relationships between video frames [19]. Furthermore, the video summarization task itself is employed in several downstream practical applications, such as surveillance [180, 181], video retrieval [182, 183], among others. Advancements made in video summarization can then directly impact and improve performance on these downstream video analysis tasks.

The ML/DL community has also recently pivoted to studying model fairness as models can exhibit harmful biases against minority groups and individuals [9]. These issues of unfairness have been evidenced in many high-impact applications as well.[1] Thus, with the growing use of video summarization in numerous applications, it is extremely important to ensure that these automated methods are fair and unbiased, both at the *individual-level* [186] and at the *group-level* [187] (such as with regards to *sensitive attributes* like ethnicity and sex). However, no work has been undertaken in *fair video summarization*, while significant progress has been made in developing fair models for other tasks/fields in ML/DL [9, 14].

The major reason for this lack of development can be directly attributed to the lack of any video summarization datasets containing individuals, and appropriate annotations reflecting their sensitive attributes (such as sex and ethnicity). Current benchmark datasets used to train and evaluate video summarization models are the *TVSum* [76] and *SumMe* [77] datasets. Both datasets consist of user/home videos that do not primarily contain human subjects[2] and lack information regarding any protected groups or sensitive attributes. Thus, to bridge this gap hindering the development of fair video summarization models and benchmark existing models for unfairness, we propose the *FairVidSum*[3] dataset, which consists of 34 videos[4] containing multiple individuals spanning diverse settings such as interviews, podcasts, and panel discussions. Unlike the other datasets, we provide manual annotations for sensitive attributes (fairness) as well as frame importance scores (utility).

[1]Notable examples include Microsoft's Tay chatbot that became racist and homophobic after training on user data online [184], and the COMPAS tool which recommended that black individuals were more likely to reoffend compared to other ethnicities, despite no statistical differences between the individuals themselves [185].
[2]With the exception of a few videos in *TVSum* which we manually annotate for fairness information.
[3]`https://sites.google.com/view/fairvidsum`
[4]Note that this is in line with the other datasets– *SumMe* consists of 25 videos, and *TVSum* consists of 50 videos.

We also propose and analytically define the *fair video summarization problem*, to allow for the development of fair methods at the individual- and group-level. Furthermore, we propose novel metrics to evaluate (un)fairness in current SOTA supervised and unsupervised video summarization models and benchmark them. Finally, for completeness we also propose a novel unsupervised method for fair video summarization named FVS-LP, which is a linear program [188] based baseline that only optimizes for fairness.

3.2.2. The Video Summarization Problem. We first describe the standard video summarization problem and discuss protocols for evaluating utility of trained models as well. In the subsequent section, we introduce the fair video summarization problem, and provide an analytical definition for it, along with evaluation metrics and motivating use-cases. Note that we only consider unimodal video summarization models as these are more commonly used in the context of deep learning [19].

3.2.2.1. *Unsupervised Video Summarization.* Let a video V consist of n frames $X = \{x_1, x_2, ..., x_n\}$ where $x_i \in \mathbb{R}^d$. These are sampled at some frequency (usually 2 frames per second [19]) from V and hence, n is generally large. Here, d is the dimension of the feature descriptor of the frame (for example, this could represent features extracted per frame using a ResNet [189]). An *unsupervised* video summarization model can then generally be denoted as $\mathcal{M}^{\text{unsup}}$ that takes in as input a summary length requirement $k \ll n$ and the original video frame set X, and outputs a set of key frames constituting the video summary as $S = \{x_j\}_{j=1}^{k} \subseteq X$. That is, $\mathcal{M}^{\text{unsup}}(X, k) = S$, where $S \in \mathbb{R}^{k \times d}$. The summary length budget k is generally set to be 15% of the original video length, that is, $k/n = 0.15$.

3.2.2.2. *Supervised Video Summarization.* While unsupervised variants are better suited for video summarization [19] since they model the application scenarios in a more realistic manner (human-level annotations are hard to obtain), supervised models are employed as well. A supervised model also takes in as input $Y = \{y_i\}_{i=1}^{m}$ where $0 < y_i \leq 1$ is an importance score given by a human annotator for a corresponding frame $x_i \in X$.[5] Annotations are only obtained for a small subset of

[5]Importance score annotations are generally obtained between 1 (least important) and 5 (most important) and then normalized to lie between 0 and 1.

frames m since n can be quite large. Thus, for a supervised model, we can obtain a summary as $\mathcal{M}^{\text{sup}}(X, k, Y) = S$ where $|S| = k$.

3.2.2.3. *Evaluating Models.* Trained video summarization models are evaluated based on the agreement of the generated summary for a video with its ground truth summary obtained using the annotated importance scores provided by a given user. Note that obtaining summaries from the importance scores Y is also an optimization problem since we have a budget k for the length of the summary. Usually, the 0/1 knapsack [190] problem is used to obtain user summaries in this manner [54]. Thus, if we have u users who annotated video V, we will have summaries available denoted as $O_1^V, O_2^V, ..., O_u^V$ corresponding to each user. The given model generates a summary S^V for a particular video V. We can then obtain the precision and recall between each O_i^V and S^V, denoted as p_i^V and r_i^V, respectively. To evaluate models, we then calculate the average pairwise F_β-measure averaged over all user summaries as follows:

$$F_\beta^V = \frac{1}{u} \sum_{i=1}^{u} \frac{(1+\beta^2) \times p_i^V \times r_i^V}{(\beta^2 \times p_i^V) + r_i^V} \tag{3.6}$$

Usually, β is set to 1 [76], so we compute the average pairwise F_1^V-measure for a given video V. These values are then averaged over all videos V in the test set, and overall $\bar{F}_1$-measure is calculated. Further, note that for the supervised setting, videos that are used for model training cannot be used in the evaluation/test set. Hence, cross validation is generally undertaken [19] to create 80% (train) - 20% (test) splits. Although this issue of train-test splits does not arise for unsupervised models, for consistency, we follow the same protocol for evaluation of all models.

3.2.3. The Fair Video Summarization Problem. We now define the fair video summarization problem for a video V. Here, along with X, Y, and k, we are also given (fairness) information regarding g individuals or protected groups as $\mathcal{H} = \{H^1, H^2, ..., H^g\}$ where $H^j \in \{0, 1\}^n$ and $H_i^j = 1$ implies that individual/group j is present in frame i. Conversely, $H_j^i = 0$ implies that individual/group j is absent in frame i. Note that unlike importance scores these are not subjective decisions, so we have discrete labels indicating individual/group presence in frames. Note that this abstraction using H^j is very flexible, and can allow for the development of fair models that optimize

for individual fairness or group fairness. For individual fairness this constitutes the idea that all persons in the video should be represented in approximately the same proportions in the generated summary as they appear in the entire video. For group fairness, this could constitute different groups being represented in the same proportions in the summary frames as their proportions in the overall video frames. For example, for *ethnicity* as the sensitive attribute, this would necessitate proportional representation for each ethnicity in summary frames compared to total video frames. This is the very notion of *disparate impact* [191] and ensures that no protected group or individual[6] be adversely affected as a result of a predictive algorithm.

A fair video summarization model $\mathcal{M}^{\text{fair}}$ then also takes in as input $\mathcal{H}$ and generates summary S for video V as $\mathcal{M}^{\text{fair}}(X, k, \mathcal{H}) = S$. Along with optimal utility performance, the model must ensure that the proportion of appearance of entities represented by $\mathcal{H}$ are as close as possible to their overall proportions in the video V. The supervised fair variant can also be defined similarly. As is evident by our definition, in this work we only consider optimizing one type of $\mathcal{H}$ at a time (that is, *sex* consisting of *male/female* appearances in frames). However, as our dataset has information regarding multiple groups, this can be studied in future work.

3.2.3.1. *Motivating Examples.* Consider a platform such as YouTube [192]. For simplicity, consider a set of news/podcast videos on the platform that have one male and one female host. Summaries for these videos are generated on the platform as the user browses the homepage. Here too, if a standard summarization model is used, there is no guarantee that the outputted video summary will respect the appearance proportions of the male/female hosts in the original video. In fact, even if the original video has 50%-50% appearance proportions for both male/female hosts, the model might skew these proportions heavily in the generated summary. A *fair* video summarization model instead would ensure that both male and female hosts appear in roughly the same amounts as in the original video, leading to fair representation.

Consider another example– a video surveillance application which utilizes video summarization models in the backend, such as in [180, 181] being used by law enforcement with multiple persons appearing in the video. If a standard video summarization model is used, the generated summary footage might have certain individuals appearing for large segments of the summary and might not

[6]For brevity, at times we use the term protected groups to also refer to the set of individuals, but this will be clear from context.

reflect their actual proportion of appearance in the overall video. As a result, this might lead to a falsified description of the original footage. On the other hand, if a fair video summarization model is used, the individuals would appear in the summary in the same proportions as in the original video footage, and result in a more *fair* overview. The same arguments can be made with regards to people from different ethnicities or gender appearing in the summary footage and preventing discrimination and bias at the group-level.

3.2.3.2. *Evaluating Unfairness.* Now that we have described the fair summarization problem, it is important to propose metrics for evaluating the discrepancies in fairness. Our basic goal is to measure whether or not each entity constituting $\mathcal{H}$ follows the same proportions in the summary as they do in the original video. To do so, we propose the SumBal metric, which is a modified version of the Balance fairness metric generally employed in fair unsupervised learning tasks [14]:

$$\begin{aligned} \text{SumBal}(S, X, \mathcal{H}) &= \min_{H^g \in \mathcal{H}} \min \left\{ R(S, X, H^g), \frac{1}{R(S, X, H^g)} \right\} \\ \text{where } R(S, X, H^g) &= \sum_{i=1}^{n} \frac{H_i^g}{n} \Big/ \sum_{x_j \in S} \frac{H_j^g}{k} \end{aligned} \tag{3.7}$$

Here, $R(S, X, H^g)$ is the ratio of the proportion of appearances of group/individual g in the overall video to the generated summary S. We take the minimum between $R(S, X, H^g)$ and $1/R(S, X, H^g)$ to account for both under-representation and over-representation cases. Finally, SumBal returns the minimum over all groups/individuals and hence SumBal $\in [0, 1]$. We can take a simple example where we have a video with two individuals, A and B. Person A appears in 20% of the video frames and Person B appears in 50% of the frames, with 30% frames having no individuals. Now, assume that we generate a summary using a model which has the following proportions– Person A appears in 40% of the summary frames (over-representation) and Person B appears in 30% of the summary frames (under-representation). Then the SumBal term for Person A would be calculated as: $\min\{0.2/0.4, 0.4/0.2\} = 0.5$ and for Person B would be calculated as $\min\{0.5/0.3, 0.3/0.5\} = 0.6$. Since we take the minimum over all groups/individuals to calculate SumBal for the video, we get $\min\{0.5, 0.6\} = 0.5$ and the violating individual (with lowest fairness) is Person A.

3.2.4. The *FairVidSum* Dataset.

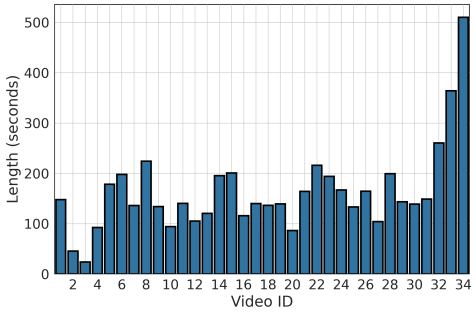


FIGURE 3.2. Video length distribution.

3.2.4.1. *Collecting Videos.* As mentioned before, our goal is to select videos that feature multiple individuals in diverse settings that allow us to annotate and account for fairness information. Similar to *TVSum* [76], we collect videos from YouTube [192]. We use the search terms "panel discussions", "podcasts", "interviews", "debates", "news", "discussions", and combinations of these keywords. Moreover, we restrict our videos to ones with a Creative Commons license, that lie between 1-4 minutes, and those that contain more than a single shot. Using this strategy we obtain 22 videos. Moreover, while the *SumMe* dataset [77] has no videos that meet this criteria, *TVSum* has a set of few videos (such as in the "documentaries" category) that we can use. In this manner, we also add another 12 videos from *TVSum* to *FairVidSum* and annotate them for fairness information. Thus, *FairVidSum* currently has a total of 34 videos, in line with current video summarization datasets. For more details refer to appendix in [20].

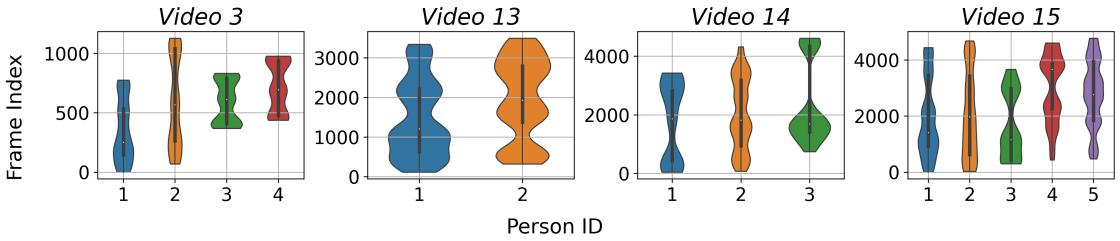
3.2.4.2. *Annotating Videos with Importance Scores.* We follow much of the same procedure as used in *TVSum* [76]. We employ 10 annotators who consist of individuals from diverse fields in either graduate or post-graduate study. Annotators are first required to watch videos on mute in a single setting to ensure that the annotation scores are only based on visual information [76]. Similarly, to alleviate chronological bias [76], frames are shuffled randomly. Next, to obtain scores annotators are shown uniformly sampled frames at 2 frames per second. Each annotator annotates every video and is required to label the provided frames with a score between 1 (least important) to 5 (most important) to be included in the summary. This task excludes the 12 *TVSum* additions as those already possess annotation scores. In this manner, we obtain 15400 responses total over all videos. Note that the number of annotators employed for this purpose is also satisfactory, as our annotation consistency analysis will later show.

3.2.4.3. *Annotating Videos with Fairness Information.* Other than these subjective annotations for importance scores, part of our dataset requires objective annotations for individuals and their sensitive attribute information. To do so we employ 4 annotators who collectively annotate all 34 videos with this information. Note that compared to annotation scores which are generally obtained for a subset of frames, fairness information needs to be collected for the entire video to calculate unfairness (such as using the SumBal metric). Thus, here, we annotate over 168120 frames total with information regarding different individuals appearing in frames and their sensitive attributes with respect to sex and ethnicity. For *sex* we annotate as *Male/Female* and for *ethnicity* we annotate for *White, Black, Middle Eastern, Asian,* and *Hispanic*.

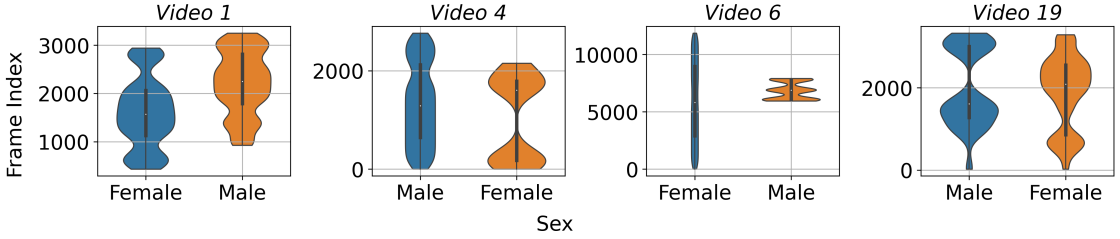
3.2.4.4. *Distribution of Individuals and Sensitive Attributes Across Videos.* We aim to analyze the distribution of individuals appearing across videos. For this purpose, we randomly sample 8 videos out of 34, and plot the distributions of individuals as well as the distributions of *sex* and *ethnicity* protected groups in those videos as a function of their video frames using violin plots. These are visualized in Figure 3.3. It is evident that both the number as well as frame-level distribution of individuals and protected groups varies widely across videos. This also demonstrates one of the challenges associated with developing fair summarization models, as they need to be able to account for fairness in many diverse application settings.

Next, we provide remaining violin plots for all videos in *FairVidSum* to visualize the distribution of unique individuals (Figure 3.4), *sex* sensitive attribute (Figure 3.5), and *ethnicity* sensitive attribute (Figure 3.6). These distributions highlight the importance and difficulty of summarizing videos while ensuring fairness. The individual plots, in particular, showcase the most challenging scenarios, as they contain videos (such as Vid. #24, #25, #30, #33) with numerous individuals appearing in very limited frames. Consequently, any missing individuals would result in an unfair summary and a SumBal score of zero. The plots emphasize the need to capture and represent proportionality and fairness in the video summaries.

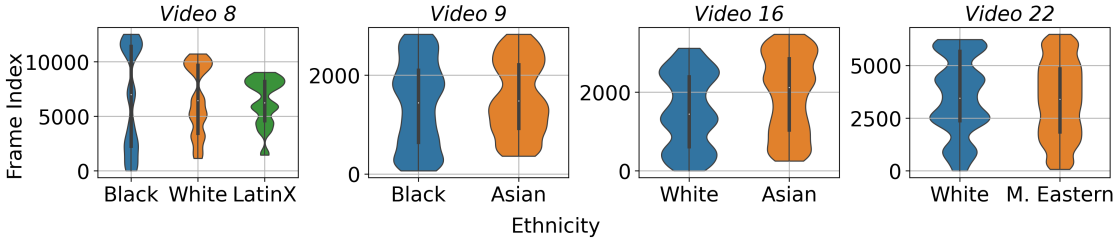
We also analyze group-level information for each video as a function of the annotation trend. Here, we can visualize the mean importance score for a video as a function of the frame indices, while also denoting sensitive attribute information for the frames. We demonstrate this for *Video 19* and *sex* as the protected group in Figure 3.7a and for *Video 16* with *ethnicity* as the protected group in



(a) Distribution of unique entities/individuals appearing across subset of videos.



(b) Distribution of *sex* sensitive attribute across subset of videos.



(c) Distribution of *ethnicity* sensitive attributes across subset of videos.

FIGURE 3.3. Violin plots showcasing the distribution of individuals and protected groups / sensitive attributes across randomly sampled videos.

Figure 3.7b. It can be seen that group-level information varies widely, and there is little correlation between importance scores and group-level information that would allow existing models to be fair.

3.2.4.5. *Annotator Consistency.* We now cover another aspect of our dataset– the annotations, and their consistency. Annotator consistency with respect to video summarization is usually measured using the Cronbach's alpha (CA) [193]. A higher CA value indicates more consistency among annotations. For *FairVidSum*, the CA value is 0.995. This is much higher than both *SumMe* (CA=0.74) and *TVSum* (CA=0.81).

Annotator consistency can also be observed qualitatively for a given video. We can visualize this as a heatmap with rows as individual annotators, columns as respective video frames, and each cell thus representing the annotator's importance score for that frame. We show this in Figure 3.8 for

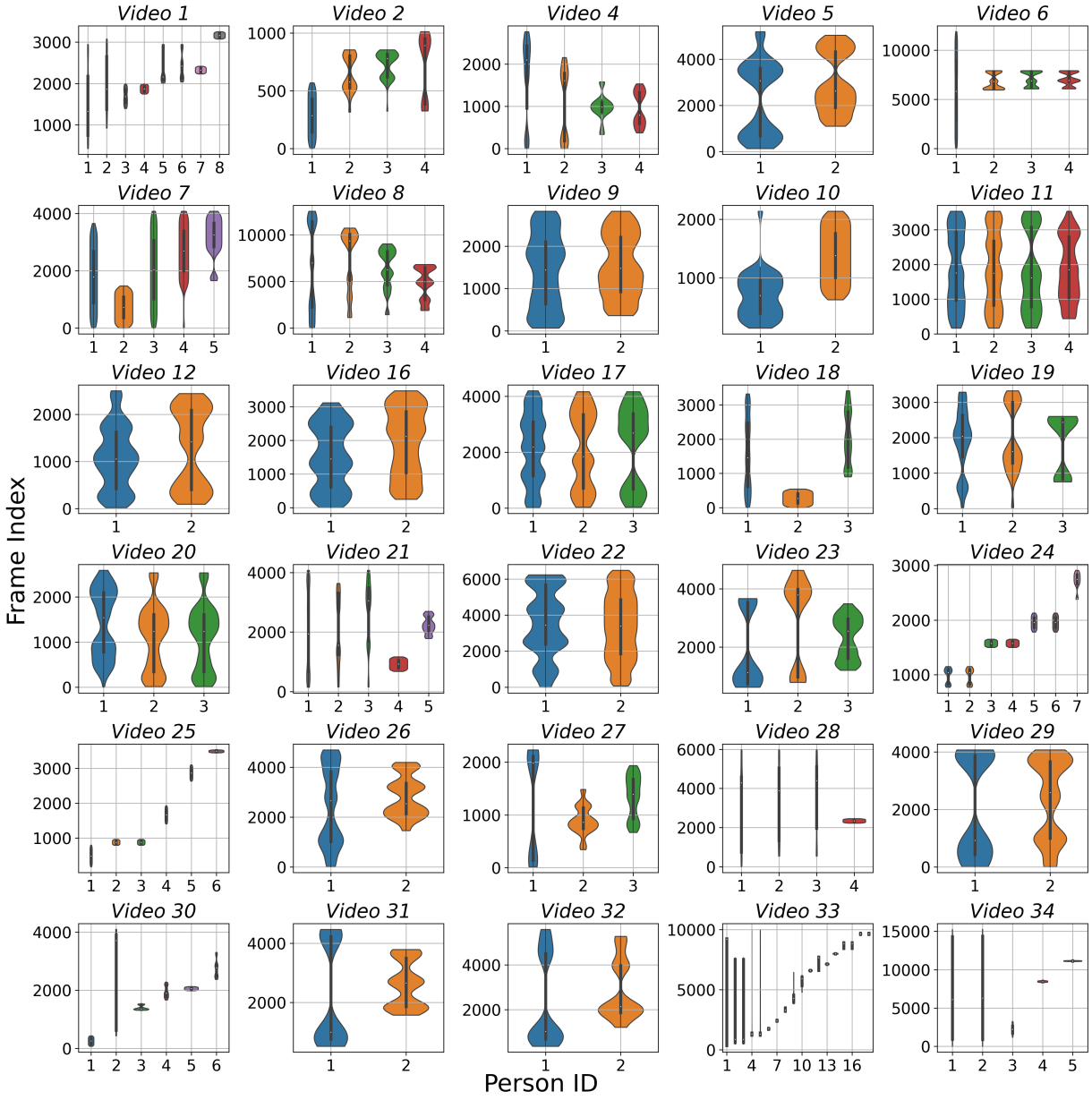


FIGURE 3.4. Violin plots showcasing the distribution of unique *entities/individuals* appearing across all videos. *Videos #3, #13, #14, #15* are discussed previously.

Video 19. It can be seen that for most frames, annotators agree on similar importance scores, also indicating why the CA value is so high for *FairVidSum*.

3.2.5. The FVS-LP Fair Video Summarization Baseline. We now present our proposed method for fair video summarization– the Fair Video Summarization Linear Program (FVS-LP)

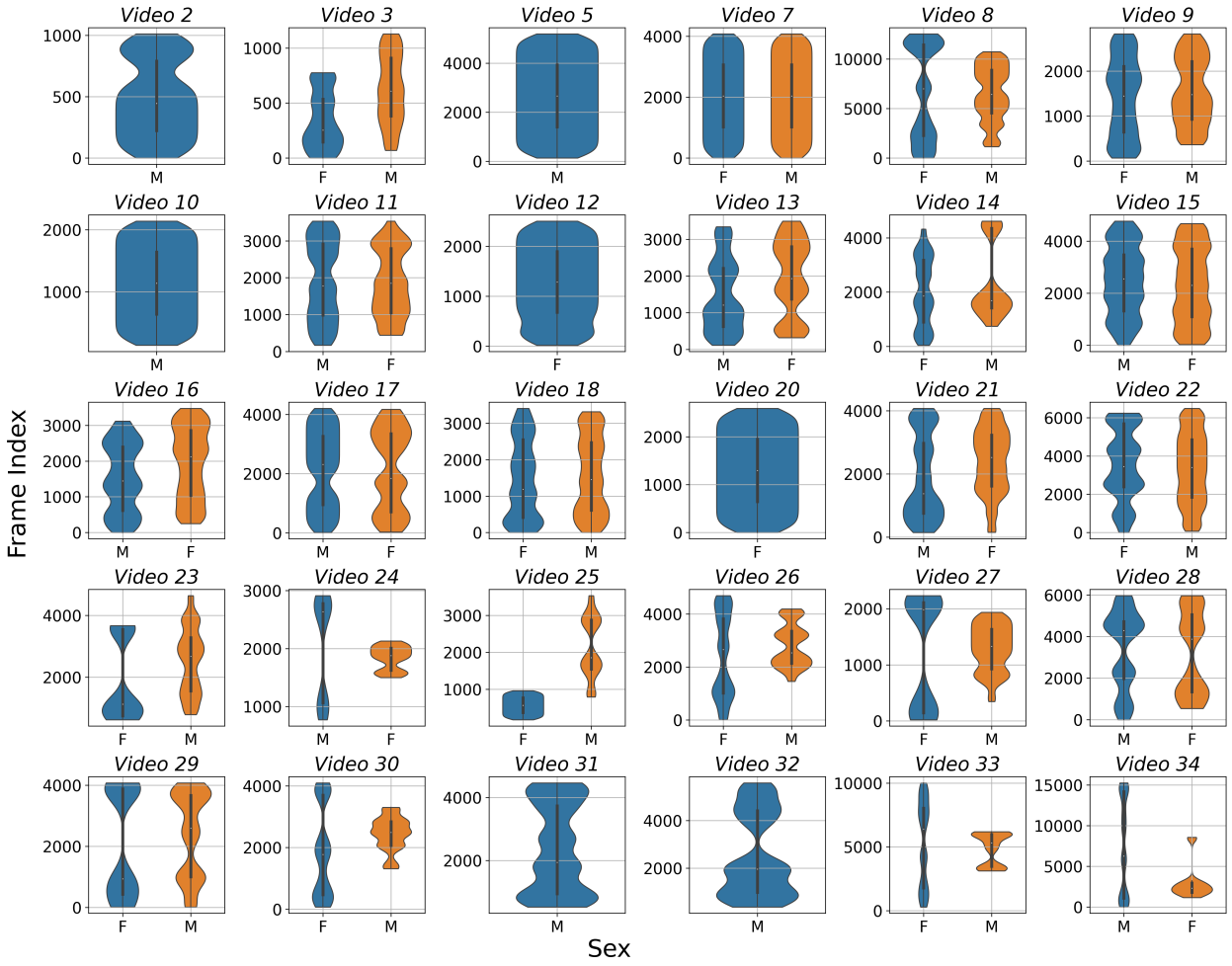


FIGURE 3.5. Violin plots showcasing the distribution *sex* sensitive attribute across all videos. Here M denotes *Male*, and F denotes *Female*. *Videos #1, #4, #6, #19* are discussed previously.

baseline which is a simple linear program [188] approach that only optimizes for fairness and selects frames such that the group proportions in the selected summary are as close as possible to the group proportions of the overall video. Let $\mathbf{0}_m$ and $\mathbf{1}_m$ denote an m length vector of all zeros and all ones, respectively. We have a given video V and its set of frames X, along with the set of group memberships $\mathcal{H}$. First, we transform $\mathcal{H}$ to matrix form for formulating the LP. Let $\mathcal{G} \in \{0,1\}^{n \times g}$ be derived from $\mathcal{H}$ such that each row vector $\mathcal{G}_i \in \{0,1\}^g$, $i \in [n]$ represents a frame and each of its entries are either 0 for absence or 1 for presence of a group in the frame. Let $0 \leq \mathbf{x} \leq 1$ be the

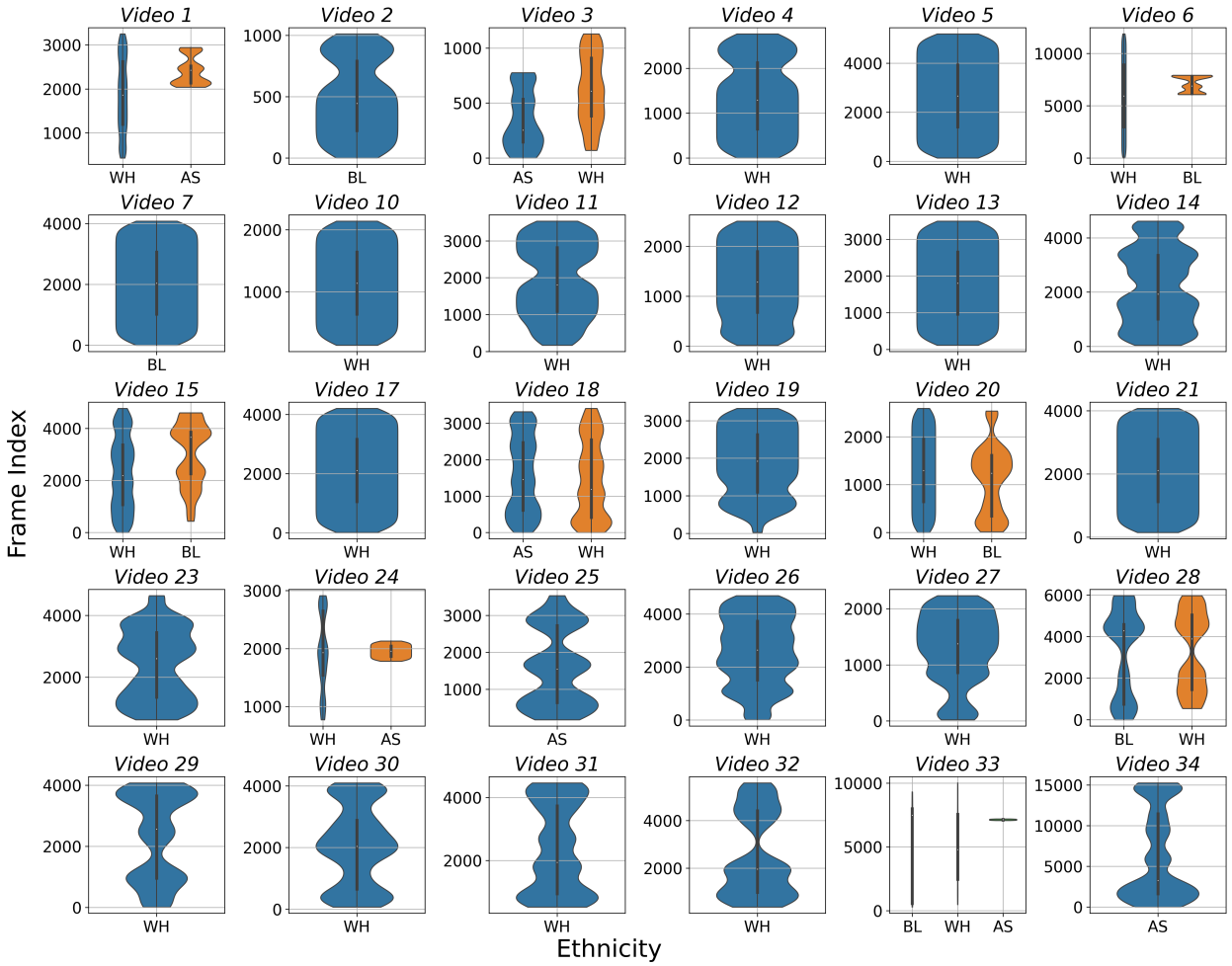
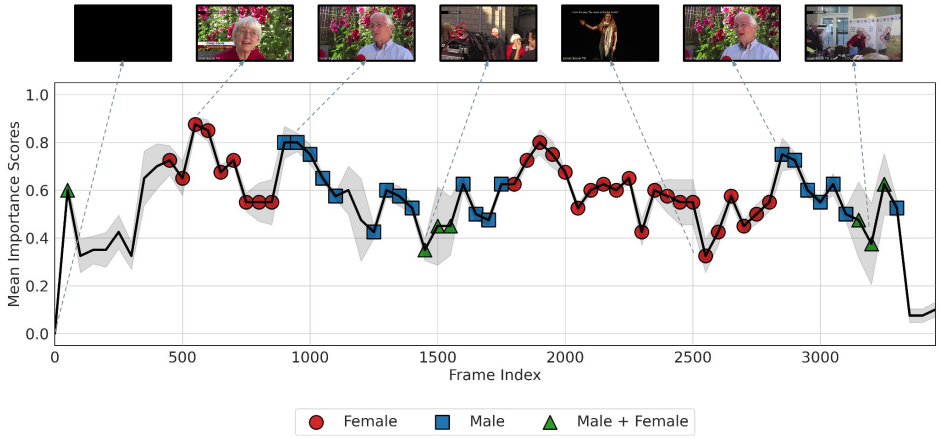


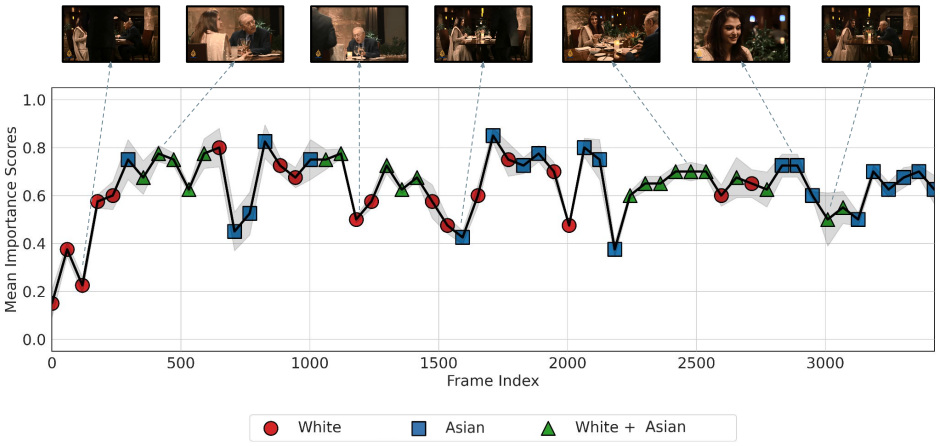
FIGURE 3.6. Violin plots showcasing the distribution *ethnicity* sensitive attribute across all videos. Here WH denotes *White*, BL denotes *Black*, and AS denotes *Asian*. *Videos #8, #9, #16, #22* are discussed previously.

optimization variable where each entry of $\mathbf{x} \in \mathbb{R}^n$ indicates if a frame is selected in the summary, then the LP can be written as Equation 3.8:

$$\begin{aligned} \text{minimize} \quad & \mathbf{0}_n^\top \mathbf{x} \\ \text{subject to} \quad & \mathcal{G}^\top \mathbf{x} = k \cdot \frac{1}{n} \sum_{i=1}^{n} \mathcal{G}_i \\ & \mathbf{1}_n^\top \mathbf{x} = k \\ & 0 \leq \mathbf{x} \leq 1. \end{aligned} \tag{3.8}$$



(a)



(b)

FIGURE 3.7. Mean importance (annotated) scores for (a) *Video 19* with protected group labels for *sex* and (b) for *Video 16* with protected group labels for *ethnicity*.

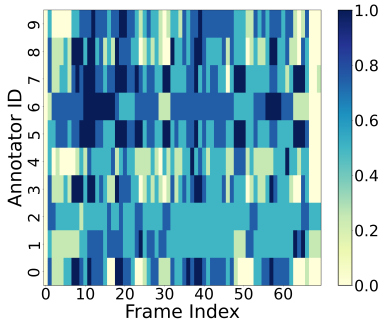


FIGURE 3.8. Annotator consistency matrix for *Video 19*.

Note that since we are only optimizing for fairness, we do not care about utility and our optimization objective can simply be a vector of all zeros. Now, as is evident, the first constraint simply ensures that the sum of the selected samples' group memberships is equal to k times the group proportions for the overall video. The second constraint ensures that the number of selected samples must be exactly k. After solving the above LP for $\mathbf{x}$, we can obtain the indices of summary frames selected from the set of frames X by rounding the solution, as $I = \{i : \text{round}(\mathbf{x}_i) = 1\}$. Then, we can get the summary S of video V as $S = \{X_i \ : \ \forall i \in I\}$.

3.2.6. Results. We now present results for benchmarking SOTA supervised and unsupervised models on *FairVidSum*. We utilize the following unsupervised models: CA-SUM [73], AC-SUM-GAN [74], SUM-GAN-AAE [75], SUM-GAN-SL [63], SUM-IND [62] and the following supervised models: DSNet [71], VASNet [194], PGL-SUM [72]. Moreover, we also provide baseline results for a randomly generated summary (Random) and a summary generated using the knapsack algorithm on the average human annotated importance scores (Human). Finally, we also present results for FVS-LP while optimizing for each protected group type (*individual*, *sex*, and *ethnicity*). For each model/baseline, we provide the group members that achieve the minimum fairness values as well.

3.2.6.1. *Training and Evaluation.* We follow the standard evaluation procedure in existing video summarization literature, which involves randomly splitting the entire dataset into multiple parts or splits, typically 5, each split subjected to an 80:20 train/test partitioning [71, 72, 73, 74, 75, 194, 195]. The models are trained on the training set of a given split and subsequently evaluated on the corresponding test set within the same split. The video distribution for all 5 train/test splits is as follows:

- Split #1 Test set: Vid. #9, Vid. #11, Vid. #19, Vid. #25, Vid. #26, Vid. #34
- Split #2 Test set: Vid. #2, Vid. #8, Vid. #20, Vid. #24, Vid. #29, Vid. #32
- Split #3 Test set: Vid. #8, Vid. #18, Vid. #20, Vid. #24, Vid. #28, Vid. #33
- Split #4 Test set: Vid. #4, Vid. #8, Vid. #17, Vid. #18, Vid. #29, Vid. #30
- Split #5 Test set: Vid. #3, Vid. #15, Vid. #16, Vid. #20, Vid. #25, Vid. #28

The F_1-measure evaluates the similarity between a model predicted summary and a user-defined summary by assessing their overlap. The F_1 scores are calculated for each individual video and then

TABLE 3.3. Comparison of SOTA video summarization approaches on *FairVidSum*. The utility and fairness averages are calculated across all five splits. The violating groups that achieve the minimum fairness SumBal scores are also presented. Results on FVS-LP (our fairness baseline) along with Random and Human baselines are also provided. Blue/red indicates highest/lowest performance.

Model	Type	Average	SumBal (*Sex*)			SumBal (*Ethnicity*)			SumBal (*Individual*)		
		F_1 Measure	Average	Min	Violating	Average	Min	Violating	Average	Min	Violating
Random	-	14.92	0.9497	0.8814	*Male* (Vid. #30)	0.9468	0.6670	*Asian* (Vid. #33)	0.8747	0.6670	*Person 13* (Vid. #33)
Human	-	68.91	0.4605	0.0000	*Female* (Vid. #30)	0.5503	0.0000	*Asian* (Vid. #25)	0.2773	0.0000	*Person 2* (Vid. #18)
CA-SUM [73]	Unsupervised	62.78	0.5201	0.0000	*Female* (Vid. #30)	0.5468	0.0000	*Asian* (Vid. #24)	**0.2441**	**0.0000**	*Person 4* (Vid. #8)
AC-SUM-GAN [74]	Unsupervised	64.33	0.5176	0.0000	*Female* (Vid. #30)	0.5455	0.0000	*Asian* (Vid. #24)	0.2616	0.0000	*Person 4* (Vid. #8)
SUM-GAN-AAE [75]	Unsupervised	63.81	0.5222	0.1302	*Female* (Vid. #26)	0.5665	0.0000	*Asian* (Vid. #24)	0.2739	0.0000	*Person 4* (Vid. #8)
SUM-GAN-SL [63]	Unsupervised	**64.92**	0.5254	0.0000	*Female* (Vid. #30)	0.5661	0.0000	*Asian* (Vid. #24)	0.2550	0.0000	*Person 4* (Vid. #8)
SUM-IND [62]	Unsupervised	50.57	0.5677	0.0000	*Female* (Vid. #24)	0.5889	0.0000	*Asian* (Vid. #24)	0.2541	0.0000	*Person 4* (Vid. #8)
DSNet [71]	Supervised	63.69	0.5358	0.0000	*Female* (Vid. #30)	0.5478	0.0000	*Asian* (Vid. #24)	0.2706	0.0000	*Person 1* (Vid. #25)
VASNet [194]	Supervised	64.11	**0.4622**	**0.0000**	*Female* (Vid. #25)	0.5391	0.0000	*Asian* (Vid. #24)	0.2515	0.0000	*Person 4* (Vid. #8)
PGL-SUM [72]	Supervised	63.75	0.4804	0.1042	*Female* (Vid. #34)	**0.5374**	**0.0000**	*Asian* (Vid. #24)	0.2575	0.0000	*Person 4* (Vid. #8)
FVS-LP (*Sex*)	Unsupervised	15.69	**0.9987**	**0.9960**	*Female* (Vid. #4)	0.7411	0.0000	*Hispanic* (Vid.#8)	0.3062	0.0000	*Person 3* (Vid. #8)
FVS-LP (*Ethnicity*)	Unsupervised	**13.46**	0.6642	0.0000	*Female* (Vid. #25)	**0.9980**	**0.9822**	*Asian* (Vid. #33)	0.2727	0.0000	*Person 6* (Vid. #25)
FVS-LP (*Individual*)	Unsupervised	14.13	0.9556	0.6289	*Male* (Vid. #28)	0.9471	0.6559	*White* (Vid. #19)	**0.9932**	**0.9704**	*Person 6* (Vid. #25)

averaged over the entirety of a given split which are then averaged across all 5 splits. SumBal is also evaluated per video, and the same procedure is followed to obtain averages.

3.2.6.2. *Details on Model Training.* We downsample videos to 1/2 frames per second as our video frames are often repetitive. Following prior work, we utilize GoogleNet [196] (trained on ImageNet) to extract frame features from the *pool5* layer, which outputs a dimensionality of 1024. To ensure a fair comparison, we employ the same splits for training/testing across all models. When training the various models, we adhere to their original procedures, and generally employ default settings and hyperparameters. Any alterations or adjustments made to the default training parameters are as follows:

- AC-SUM-GAN: regularization_factor = 5.0, clip = 1.0, action_state_size = 8
- CA-SUM: block_size = 60, init_gain = 1.0, n_epochs = 200, clip = 1.0, lr = 1e-4, l2_req = 1e-6, reg_factor = 5.0
- PGL-SUM: clip = 1.0, lr = 1e-4, l2_req = 1e-4
- SUM-GAN-AAE: clip = 1.0, hidden_size = 512, regularization_factor = 5.0, lr = 1e-5
- SUM-GAN-SL: clip = 1.0, hidden_size = 512, regularization_factor = 5.0

3.2.6.3. *FairVidSum Benchmarking Results.* Since we have 5 evaluation splits, we present average results over all splits in Table 3.3. We also present results for the 5 individual evaluation splits as

TABLE 3.4. Comparison of SOTA video summarization model on *FairVidSum* for evaluation Split #1.

Model	Type	Average	SumBal (*Sex*)			SumBal (*Ethnicity*)			SumBal (*Individual*)		
		F_1 Measure	Average	Min	Violating	Average	Min	Violating	Average	Min	Violating
Random	-	14.74	0.9473	0.8818	*Male* (Vid. #26)	0.97	0.9415	*White* (Vid. #26)	0.8639	0.7477	*Person 2* (Vid. #19)
Human	-	67.08	0.4800	0.0000	*Female* (Vid. #25)	0.6717	0.0000	*Asian* (Vid. #25)	0.3080	0.0000	*Person 1* (Vid. #25)
CA-SUM [73]	Unsupervised	62.11	0.5104	0.1044	*Female* (Vid. #34)	0.6746	0.4292	*Asian* (Vid. #9)	0.3042	0.0000	*Person 1* (Vid. #25)
AC-SUM-GAN [74]	Unsupervised	63.99	0.4481	0.0000	*Female* (Vid. #25)	0.6467	0.0000	*Asian* (Vid. #25)	0.3261	0.0000	*Person 1* (Vid. #25)
SUM-GAN-AAE [75]	Unsupervised	63.44	0.4887	0.1302	*Female* (Vid. #26)	0.6921	0.4567	*Asian* (Vid. #9)	0.2952	0.0000	*Person 1* (Vid. #25)
SUM-GAN-SL [63]	Unsupervised	64.77	0.5298	0.2156	*Male* (Vid. #26)	0.7056	0.4567	*Asian* (Vid. #9)	0.2792	0.0000	*Person 1* (Vid. #25)
SUM-IND [62]	Unsupervised	49.47	0.5415	0.3868	*Male* (Vid. #26)	0.6641	0.4503	*Asian* (Vid. #9)	0.3019	0.0000	*Person 1* (Vid. #25)
DSNet [71]	Supervised	63.24	0.4432	0.1049	*Female* (Vid. #31)	**0.6095**	0.2742	*Asian* (Vid. #25)	0.3040	0.0000	*Person 1* (Vid. #25)
VASNet [194]	Supervised	**66.14**	0.3386	**0.0000**	*Female* (Vid. #25)	0.5866	**0.0000**	*Asian* (Vid. #25)	0.2800	0.0000	*Person 1* (Vid. #25)
PGL-SUM [72]	Supervised	65.18	**0.4392**	0.1042	*Female* (Vid. #34)	0.5869	0.2631	*Asian* (Vid. #34)	**0.2701**	**0.0000**	*Person 1* (Vid. #25)
FVS-LP (*Sex*)	Unsupervised	17.03	**0.9990**	**0.9975**	*Female* (Vid. #25)	0.8357	0.6559	*White* (Vid. #19)	0.4417	0.0000	*Person 3* (Vid. #19)
FVS-LP (*Ethnicity*)	Unsupervised	**12.13**	0.3967	0.0000	*Female* (Vid. #25)	**0.9993**	**0.9983**	*Asian* (Vid. #25)	0.3186	0.0000	*Person 3* (Vid. #19)
FVS-LP (*Individual*)	Unsupervised	16.81	0.9562	0.8250	*Male* (Vid. #34)	0.8999	0.6559	*White* (Vid. #19)	**0.9930**	**0.9704**	*Person 6* (Vid. #25)

TABLE 3.5. Comparison of SOTA video summarization models on *FairVidSum* for evaluation Split #2.

Model	Type	Average	SumBal (*Sex*)			SumBal (*Ethnicity*)			SumBal (*Individual*)		
		F_1 Measure	Average	Min	Violating	Average	Min	Violating	Average	Min	Violating
Random	-	14.95	0.9837	0.9555	*Female* (Vid. #29)	0.9696	0.9199	*Asian* (Vid. #24)	0.8866	0.7417	*Person 3* (Vid. #24)
Human	-	65.00	0.4551	0.0000	*Female* (Vid. #24)	0.4768	0.0000	*Asian* (Vid. #24)	0.2909	0.0000	*Person 1* (Vid. #24)
CA-SUM [73]	Unsupervised	60.77	0.5449	0.0000	*Female* (Vid. #24)	0.4971	0.0000	*Asian* (Vid. #24)	0.1819	0.0000	*Person 4* (Vid. #8)
AC-SUM-GAN [74]	Unsupervised	62.41	0.5362	0.0000	*Female* (Vid. #24)	0.4573	0.0000	*Asian* (Vid. #24)	0.2634	0.0000	*Person 3* (Vid. #24)
SUM-GAN-AAE [75]	Unsupervised	**61.49**	0.5213	0.0000	*Female* (Vid. #24)	0.4658	0.0000	*Asian* (Vid. #24)	0.1835	0.0000	*Person 4* (Vid. #8)
SUM-GAN-SL [63]	Unsupervised	62.02	0.4958	0.0000	*Female* (Vid. #24)	0.4529	0.0000	*Asian* (Vid. #24)	**0.1714**	**0.0000**	*Person 4* (Vid. #8)
SUM-IND [62]	Unsupervised	49.80	0.6805	0.0000	*Female* (Vid. #24)	0.6940	0.0000	*Asian* (Vid. #24)	0.3570	0.0000	*Person 1* (Vid. #2)
DSNet [71]	Supervised	62.45	0.5661	0.0000	*Female* (Vid. #24)	0.5032	0.0000	*Asian* (Vid. #24)	0.2888	0.0000	*Person 4* (Vid. #8)
VASNet [194]	Supervised	60.53	0.4796	0.0000	*Female* (Vid. #24)	0.4676	0.0000	*Asian* (Vid. #24)	0.1963	0.0000	*Person 4* (Vid. #8)
PGL-SUM [72]	Supervised	61.04	**0.4627**	**0.0000**	*Female* (Vid. #24)	**0.4300**	**0.0000**	*Asian* (Vid. #24)	0.1881	0.0000	*Person 4* (Vid. #8)
FVS-LP (*Sex*)	Unsupervised	15.30	**0.9985**	**0.9968**	*Male* (Vid. #24)	0.7033	0.0000	*Hispanic* (Vid. #8)	0.4183	0.0000	*Person 1* (Vid. #2)
FVS-LP (*Ethnicity*)	Unsupervised	14.70	0.8599	0.4721	*Female* (Vid. #29)	**0.9981**	**0.9934**	*Asian* (Vid. #24)	0.3755	0.0000	*Person 1* (Vid. #2)
FVS-LP (*Individual*)	Unsupervised	**12.41**	0.9714	0.8350	*Male* (Vid. #2)	0.9648	0.8350	*Black* (Vid. #2)	**0.9963**	**0.9905**	*Person 2* (Vid. #2)

Tables 3.4 - 3.8 It can be observed that all the results exhibit similar trends. As can be seen in Tables 3.3 to 3.8 both unsupervised and supervised SOTA models tend to achieve high utility performance computed in terms of the F_1-measure (> 60) averaged over all videos and all splits.[7] However, these models have low fairness performance, with minimum SumBal scores for all three group types: *sex*, *ethnicity*, and *individuals* most often tend to be 0 and generally < 0.5. Interestingly, the group members that achieve the lowest fairness values across all splits and videos tend to consistently be *Female* for *sex* as the protected group, *Asian* for *ethnicity* as the protected group, and *Person 4* for *individual* fairness. We also present average SumBal scores which are higher at times, but have

[7]This utility performance is in line with SOTA results for *TVSum* and *SumMe*; refer to [19, 76] for details.

TABLE 3.6. Comparison of SOTA video summarization models on *FairVidSum* for evaluation Split #3.

Model	Type	Average	SumBal (*Sex*)			SumBal (*Ethnicity*)			SumBal (*Individual*)		
		F_1 Measure	Average	Min	Violating	Average	Min	Violating	Average	Min	Violating
Random	-	14.96	0.9477	0.8758	*Female* (Vid. #24)	0.8942	0.6670	*Asian* (Vid. #33)	0.8614	0.6670	*Person 13* (Vid. #33)
Human	-	67.61	0.4784	0.0000	*Female* (Vid. #24)	0.4518	0.0000	*Asian* (Vid. #24)	0.2097	0.0000	*Person 2* (Vid. #18)
CA-SUM [73]	Unsupervised	59.76	0.5547	0.0000	*Female* (Vid. #24)	**0.3859**	**0.0000**	*Asian* (Vid. #24)	0.1901	0.0000	*Person 4* (Vid. #8)
AC-SUM-GAN [74]	Unsupervised	61.68	0.6236	0.0000	*Female* (Vid. #24)	0.4123	0.0000	*Asian* (Vid. #24)	0.1999	0.0000	*Person 4* (Vid. #8)
SUM-GAN-AAE [75]	Unsupervised	61.63	0.5859	0.0000	*Female* (Vid. #24)	0.4291	0.0000	*Asian* (Vid. #24)	0.2779	0.0000	*Person 1* (Vid. #24)
SUM-GAN-SL [63]	Unsupervised	**62.77**	0.6211	0.0000	*Female* (Vid. #24)	0.4122	0.0000	*Asian* (Vid. #24)	0.2767	0.0000	*Person 3* (Vid. #24)
SUM-IND [62]	Unsupervised	44.79	**0.5061**	**0.0000**	*Female* (Vid. #24)	0.4214	0.0000	*Asian* (Vid. #24)	**0.1481**	**0.0000**	*Person 4* (Vid. #8)
DSNet [71]	Supervised	60.16	0.5448	0.0000	*Female* (Vid. #26)	0.4364	0.0000	*Asian* (Vid. #24)	0.2304	0.0000	*Person 4* (Vid. #8)
VASNet [194]	Supervised	60.93	0.5646	0.0000	*Female* (Vid. #24)	0.4551	0.0000	*Asian* (Vid. #24)	0.2858	0.0000	*Person 1* (Vid. #24)
PGL-SUM [72]	Supervised	59.15	0.5117	0.0000	*Female* (Vid. #24)	0.4086	0.0000	*Asian* (Vid. #24)	0.1965	0.0000	*Person 4* (Vid. #8)
FVS-LP (*Sex*)	Unsupervised	17.03	**0.9990**	**0.9968**	*Male* (Vid. #24)	0.5242	0.0000	*Hispanic* (Vid. #8)	0.0939	0.0000	*Person 3* (Vid. #8)
FVS-LP (*Ethnicity*)	Unsupervised	16.06	0.7394	0.0000	*Male* (Vid. #33)	**0.9951**	**0.9822**	*Asian* (Vid. #33)	0.1387	0.0000	*Person 4* (Vid. #8)
FVS-LP (*Individual*)	Unsupervised	**16.03**	0.9371	0.6289	*Male* (Vid. #28)	0.9373	0.6499	*White* (Vid. #28)	**0.9923**	**0.9731**	*Person 14* (Vid. #33)

TABLE 3.7. Comparison of SOTA video summarization models on *FairVidSum* for evaluation Split #4.

Model	Type	Average	SumBal (*Sex*)			SumBal (*Ethnicity*)			SumBal (*Individual*)		
		F_1 Measure	Average	Min	Violating	Average	Min	Violating	Average	Min	Violating
Random	-	16.08	0.9324	0.8814	*Male* (Vid. #30)	0.9760	0.9043	*White* (Vid. #18)	0.8886	0.7253	*Person 3* (Vid. #30)
Human	-	76.51	0.3735	0.0000	*Female* (Vid. #30)	0.5895	0.0752	*White* (Vid. #30)	0.2522	0.0000	*Person 2* (Vid. #18)
CA-SUM [73]	Unsupervised	71.89	0.4336	0.0000	*Female* (Vid. #30)	**0.6222**	**0.0750**	*White* (Vid. #30)	0.2501	0.0000	*Person 4* (Vid. #8)
AC-SUM-GAN [74]	Unsupervised	74.02	**0.4258**	**0.0000**	*Female* (Vid. #30)	0.6222	0.2741	*White* (Vid. #30)	**0.2026**	**0.0000**	*Person 4* (Vid. #8)
SUM-GAN-AAE [75]	Unsupervised	72.49	0.5096	0.0000	*Female* (Vid. #30)	0.6994	0.4450	*White* (Vid. #29)	0.2823	0.0000	*Person 4* (Vid. #8)
SUM-GAN-SL [63]	Unsupervised	**73.42**	0.4458	0.0000	*Female* (Vid. #30)	0.6988	0.4450	*White* (Vid. #29)	0.2615	0.0000	*Person 4* (Vid. #8)
SUM-IND [62]	Unsupervised	60.96	0.5692	0.0000	*Male* (Vid. #29)	0.6363	0.2419	*White* (Vid. #29)	0.2004	0.0000	*Person 4* (Vid. #8)
DSNet [71]	Supervised	71.53	0.5743	0.0000	*Female* (Vid. #30)	0.6661	0.4450	*White* (Vid. #29)	0.2412	0.0000	*Person 3* (Vid. #8)
VASNet [194]	Supervised	69.95	0.4444	0.0000	*Female* (Vid. #30)	0.6470	0.2719	*White* (Vid. #30)	0.2055	0.0000	*Person 4* (Vid. #8)
PGL-SUM [72]	Supervised	70.03	0.5058	0.0000	*Female* (Vid. #30)	0.7343	0.4450	*White* (Vid. #29)	0.3027	0.0000	*Person 4* (Vid. #8)
FVS-LP (*Sex*)	Unsupervised	16.69	**0.9981**	**0.9960**	*Female* (Vid. #4)	0.8231	0.0000	*Hispanic* (Vid. #8)	0.3169	0.0000	*Person 3* (Vid. #4)
FVS-LP (*Ethnicity*)	Unsupervised	**14.49**	0.5619	0.0000	*Female* (Vid. #4)	**0.9994**	**0.9987**	*White* (Vid. #18)	0.2257	0.0000	*Person 2* (Vid. #4)
FVS-LP (*Individual*)	Unsupervised	15.90	0.9762	0.8695	*Male* (Vid. #17)	0.9926	0.9621	*White* (Vid. #29)	**0.9949**	**0.9820**	*Person 5* (Vid. #30)

very large variance showcasing that models are not inherently optimizing for fairness. The human annotated summary also fares similarly to the SOTA models, as it is only annotated for performance. Moreover, the randomly generated summary has very low utility performance scores– typically with F_1-measure values less than 15 which follows the fact that summary frames are picked completely at random. However, the random summary has high fairness scores. We hypothesize that this is the case because by picking frames uniformly at random, the probability that each group member is picked according to their proportions is uniform in expectation. As a result, random frame selection leads to improved fairness.

TABLE 3.8. Comparison of SOTA video summarization models on *FairVidSum* for evaluation Split #5.

Model	Type	Average	SumBal (*Sex*)			SumBal (*Ethnicity*)			SumBal (*Individual*)		
		F_1 Measure	Average	Min	Violating	Average	Min	Violating	Average	Min	Violating
Random	-	13.96	0.9375	0.8692	*Male* (Vid. #3)	0.9242	0.8692	*White* (Vid. #3)	0.8732	0.7020	*Person 2* (Vid. #25)
Human	-	68.34	0.5157	0.0000	*Female* (Vid. #25)	0.5615	0.0000	*Asian* (Vid. #25)	0.3257	0.0000	*Person 1* (Vid. #25)
CA-SUM [73]	Unsupervised	59.34	0.5567	0.0384	*Female* (Vid. #3)	0.5542	0.0384	*Asian* (Vid. #3)	0.2944	0.0000	*Person 1* (Vid. #25)
AC-SUM-GAN [74]	Unsupervised	59.56	0.5545	0.0390	*Female* (Vid. #3)	0.5890	0.0390	*Asian* (Vid. #3)	0.3159	0.0000	*Person 1* (Vid. #25)
SUM-GAN-AAE [75]	Unsupervised	60.01	0.5053	0.0401	*Female* (Vid. #3)	0.5460	0.0401	*Asian* (Vid. #3)	0.3304	0.0000	*Person 1* (Vid. #25)
SUM-GAN-SL [63]	Unsupervised	61.59	0.5346	0.0384	*Female* (Vid. #3)	0.5613	0.0384	*Asian* (Vid. #3)	0.2863	0.0000	*Person 1* (Vid. #25)
SUM-IND [62]	Unsupervised	47.81	0.5411	0.0390	*Female* (Vid. #3)	0.5287	0.0390	*Asian* (Vid. #3)	**0.2629**	**0.0000**	*Person 1* (Vid. #25)
DSNet [71]	Supervised	61.06	0.5506	0.0401	*Female* (Vid. #3)	**0.5238**	0.0411	*Asian* (Vid. #3)	0.2883	0.0000	*Person 1* (Vid. #25)
VASNet [194]	Supervised	62.99	0.4836	0.0000	*Female* (Vid. #25)	0.5393	0.0000	*Asian* (Vid. #25)	0.2897	0.0000	*Person 1* (Vid. #25)
PGL-SUM [72]	Supervised	**63.37**	**0.4828**	**0.0000**	*Female* (Vid. #25)	0.5274	**0.0000**	*Asian* (Vid. #25)	0.3304	0.0000	*Person 1* (Vid. #25)
FVS-LP (*Sex*)	Unsupervised	12.42	**0.9988**	**0.9975**	*Female* (Vid. #25)	0.8192	0.5633	*Black* (Vid. #20)	0.2604	0.0000	*Person 3* (Vid. #3)
FVS-LP (*Ethnicity*)	Unsupervised	9.928	0.7632	0.0000	*Female* (Vid. #25)	**0.9983**	**0.9974**	*Black* (Vid. #28)	0.3052	0.0000	*Person 3* (Vid. #3)
FVS-LP (*Individual*)	Unsupervised	**9.512**	0.9367	0.6289	*Male* (Vid. #28)	0.9406	0.6499	*White* (Vid. #28)	**0.9894**	**0.9704**	*Person 6* (Vid. #25)

Furthermore, we provide results for three versions of FVS-LP, each instantiated to optimize one type of protected group/sensitive attribute. For each of these, FVS-LP achieves the highest fairness performance across all models and baselines for the group it is optimizing for. However, it does not lead to good utility performance, which is to be expected as it is only directly optimizing for fairness. This implies that while there is a gap in fairness that can be optimized for, optimizing for both fairness and performance is a non-trivial task. For future work, methods that jointly optimize both fairness and utility can thus be proposed. Note that the trends between the average performance and Split #1 are very similar, and this is also the case for the other evaluation splits. Generally, we observe that supervised models tend to exhibit lower average SumBal values. This trend might be a direct consequence of these models' learning, which strive to closely align with human or ground truth summaries that, as previously mentioned, are solely optimized for utility. This observation further underscores the importance of incorporating a fairness evaluation and learning criterion in the model design and training process. Another crucial insight from our benchmarking analysis is the distinct difficulty in upholding *individual* fairness. This is clearly evident by the consistently lowest average SumBal values (compared with *sex* and *ethnicity*) and predominant minimum values of zero. A SumBal value of zero essentially indicates that a group or individual, though present in the original video, has been completely excluded from the generated summary.

CHAPTER 4

On the Interplay Between Adversarial and Social Robustness

In this chapter, we discuss problems at the intersection of adversarial and social robustness. This direction has been relatively less explored, with the community often focusing on only one dimension of robustness in individual works. In particular, for real-world applications, there is a dire need to optimize for both adversarial and social robustness jointly. It is also important to propose general approaches that can be used to improve upon adversarial and social robustness, while preserving utility. Thus, to bridge this gap, we 1) investigate the robustness of fair clustering models under adversarial influence, and 2) propose a generalized data selection framework for improving the fairness, accuracy, and adversarial robustness of classification models.

4.1. Robust Fair Clustering

4.1.1. Introduction. ML models are ubiquitously utilized in many applications, including high-stakes domains such as loan disbursement [197], recidivism prediction [198, 199], hiring and recruitment [200, 201], among others. For this reason, it is of paramount importance to ensure that decisions derived from such predictive models are unbiased and fair for all individuals treated [9]. In particular, this is the main motivation behind *group-level fair* learning approaches [108, 202, 203], where the goal is to generate predictions that do not *disparately impact* individuals from minority protected groups (such as ethnicity, sex, etc.). It is also worthwhile to note that this problem is technically challenging because there exists an inherent fairness-performance tradeoff [204], and thus fairness needs to be improved while ensuring approximate preservation of model predictive performance. This line of research is even more pertinent for data clustering, where error rates cannot be directly assessed using class labels to measure disparate impact. Thus, many approaches have been recently proposed to make clustering models group-level fair [15, 85, 91, 100]. In a nutshell, these approaches seek to improve fairness of clustering outputs with respect to some fairness metrics,

which ensure that each cluster contains approximately the same proportion of samples from each protected group as they appear in the dataset.

While many fair clustering approaches have been proposed, it is of the utmost importance to ensure that these models provide fair outputs even in the presence of an adversary seeking to degrade fairness utility. Although there are some pioneering attempts on fairness attacks against supervised learning models [106, 107], unfortunately, none of these works propose defense approaches. Moreover, in the unsupervised scenario, fair clustering algorithms have not yet been explored from an adversarial attack perspective, resulting in unsupervised fair clustering models being potentially vulnerable to fairness attacks. This leads us to our fundamental research question:

Are fair clustering algorithms vulnerable to adversarial attacks that seek to decrease fairness utility, and if such attacks exist, can we develop an adversarially robust fair clustering model?

We answer both these questions in the affirmative by proposing a novel *black-box* adversarial attack against fair clustering models. Here the attacker can only perturb a small percentage of protected group memberships and yet is able to degrade the fairness performance of SOTA fair clustering models significantly. Through extensive experiments using our attack approach, we find that existing fair clustering algorithms are not robust to adversarial influence, and are extremely volatile with regards to fairness utility. We conduct this analysis on a number of real-world datasets, and for a variety of clustering performance and fairness utility metrics. Finally, to achieve truly robust fair clustering, we propose the *Consensus Fair Clustering* (CFC) model which is highly resilient to the proposed fairness attack. To the best of our knowledge, CFC is the first defense approach for fairness attacks, which makes it an important standalone contribution to the unsupervised ML community.

4.1.2. Fairness Attack. In this section, we study the attack problem on fair clustering. Specifically, we propose a novel attack that aims to reduce the fairness utility of fair clustering algorithms, as opposed to traditional adversarial attacks that seek to decrease clustering performance [26]. To our best knowledge, although there have been a few pioneering attempts investigating fairness attacks [106, 107], all of them consider the supervised setting. Our proposed attack exposes a novel problem prevalent with fair clustering approaches that has not been given considerable attention yet– as the protected group memberships are input to the fair clustering optimization

problem, they can be used to disrupt the fairness utility. We study attacks under the *black-box* setting, where the attacker has no knowledge of the fair clustering algorithm being used.

4.1.2.1. *Preliminaries and Notation.* Given a tabular dataset $X{=}\{x_i\}{\in}\mathbb{R}^{n\times d}$ with n samples and d features, each sample x_i is associated with a protected group membership $g(x_i){\in}[L]$, where L is the total number of protected groups, and we denote group memberships for the entire dataset as $G{=}\{g(x_i)\}_{i=1}^{n}{\in}\,\mathbb{N}^n$. We also have $H = \{H_1, H_2, ..., H_L\}$ and H_l is the set of samples that belong to l-th protected group. A clustering algorithm $\mathcal{C}(X, K)$ takes as input the dataset X and a parameter K, and outputs labeling where each sample belongs to one of K clusters [205]. That is, each point is clustered in one of the sets $\{C_1, C_2, ..., C_K\}$ with $\cup_{k=1}^{K} C_k = X$. Based on the above, a *group-level fair* clustering algorithm $\mathcal{F}(X, K, G)$ [85] can be defined similarly to $\mathcal{C}$, where $\mathcal{F}$ takes as input the protected group membership G along with X and K, and outputs fair labeling that is expected to be more *fair* than the clustering obtained via the original unfair/vanilla clustering algorithm with respect to a given fairness utility function ϕ. That is, $\phi(\mathcal{F}(X, K, G), G) \leq \phi(\mathcal{C}(X, K), G)$. Note that ϕ can be defined to be any fairness utility metric, such as Balance and Entropy [9, 14].

4.1.2.2. *Threat Model.* Take the customer segmentation [206, 207] problem as an example and assume that the sensitive attribute considered is *age* with 3 protected groups: {*youth, adult, senior*}. Now, we can motivate our threat model as follows: the adversary can control a small portion of individuals' protected group memberships (either through social engineering, exploiting a security flaw in the system, etc.); by changing their protected group memberships, the adversary aims to disrupt the fairness utility of the fair algorithm on other uncontrolled groups. That is, there would be an overwhelming majority of some protected group samples over others in clusters. This would adversely affect the youth and senior groups, as they are more vulnerable and less capable of enforcing self-prevention. The attacker could carry out this attack for profit or anarchistic reasons.

Our adversary has partial knowledge of the dataset X but not the fair clustering algorithm $\mathcal{F}$. However, they can query $\mathcal{F}$ and observe cluster outputs. This assumption has been used in previous adversarial attack research against clustering [11, 26, 79]). They can access and switch/change the protected group memberships for a small subset of samples in G, denoted as $G_A{\subseteq}G$. Our goal of the fairness attack is to change the protected group memberships of samples in G_A such that the fairness utility value decreases for the remaining samples in $G_D{=}G\backslash G_A$. As clustering algorithms [208]

and their fair variants [103] are trained on the input data to generate labeling, this attack is a *training-time* attack. Our attack can also be motivated by considering that fair clustering outputs change with any changes made to protected group memberships G or the input dataset X. We can formally define the fairness attack as follows:

DEFINITION 4.1.1 (Fairness Attack). *Given a fair clustering algorithm $\mathcal{F}$ that can be queried for cluster outputs, dataset X, samples' protected groups G, and $G_A \subseteq G$ is a small portion of protected groups that an adversary can control, the fairness attack is that the adversary aims to reduce the fairness of clusters outputted via $\mathcal{F}$ for samples in $G_D = G \setminus G_A \subseteq G$ by perturbing G_A.*

4.1.2.3. *The Attack Optimization Problem.* Based on the above threat model, the attack optimization problem can be defined analytically. For ease of notation, we define two mapping functions:

- η : Takes G_A and G_D as inputs and gives output $G = \eta(G_A, G_D)$ which is the combined group memberships for the entire dataset. Note that G_A and G_D are interspersed in the entire dataset in an unordered fashion, which motivates the need for this mapping.
- θ : Takes G_D and an output cluster labeling from a clustering algorithm for the entire dataset as input, returns the cluster labels for only the subset of samples that have group memberships in G_D. That is, if the clustering output is $\mathcal{C}(X, K)$, we can obtain cluster labels for samples in G_D as $\theta(\mathcal{C}(X, K), G_D)$.

Based on the above notations, we have the following optimization problem for the attacker:

$$\min_{G_A} \quad \phi(\theta(O, G_D), G_D) \;\; \text{s.t.} \;\; O = \mathcal{F}(X, K, \eta(G_A, G_D)). \tag{4.1}$$

The above problem is a two-level hierarchical optimization problem [209] with optimization variable G_A, where the lower-level problem is the fair clustering problem $\mathcal{F}(X, K, \eta(G_A, G_D))$, and the upper-level problem aims to reduce the fairness utility ϕ of the clustering obtained on the set of samples in G_D. Due to the black-box nature of our attack, both the upper- and lower-level problems are highly non-convex and closed-form solutions to the hierarchical optimization cannot be obtained. In particular, hierarchical optimization even with linear upper- and lower-level problems has been shown to be NP-Hard [210], indicating that such problems cannot be solved by exact algorithms.

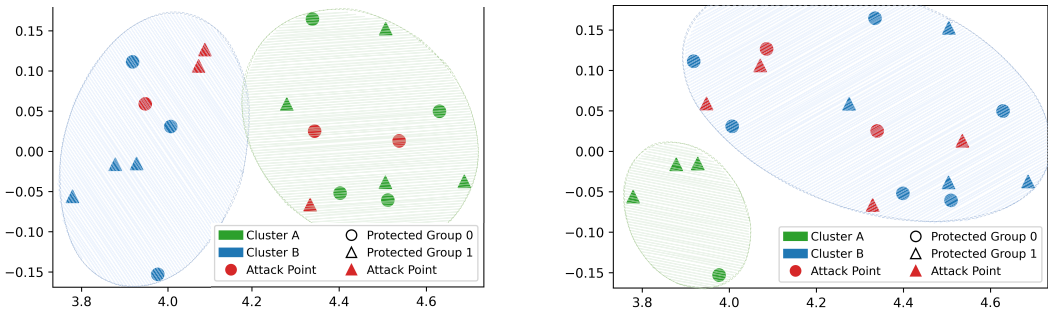
We will thus resort to generally well-performing heuristic algorithms for obtaining solutions to the problem in Eq. (4.1).

The aforementioned attack problem in Eq. (4.1) is a non-trivial optimization problem, where the adversary has to optimize G_A such that overall clustering fairness for the remaining samples in G_D decreases. Since $\mathcal{F}$ is a black-box and unknown to the attacker, first- or second-order approaches (such as gradient descent) cannot be used to solve the problem. Instead, we utilize zeroth-order optimization algorithms to solve the attack problem. In particular, we use RACOS [211] due to its known theoretical guarantees on discrete optimization problems. Moreover, our problem belongs to the same class as protected group memberships are discrete labels.

Remark. Note that in [17] we previously propose a theoretically motivated fairness disrupting attack for k-median clustering; however, this cannot be utilized to tackle the current research problem for the following reasons: (1) the attack in [17] only works for k-median vanilla clustering, thus not constituting a black-box attack on fair algorithms, (2) the attack in [17] aims to poison a subset of the input data and not the protected group memberships thus leading to a more common threat model different from us. We also cannot use existing adversarial attacks against clustering algorithms [11, 26] as they aim to reduce clustering performance and do not optimize for a reduction in fairness utility. Thus, these attacks might not always lead to a reduction in fairness utility.

4.1.3. Results for the Attack.

4.1.3.1. *Datasets.* We utilize one synthetic and four real-world datasets in our experiments. The details of our synthetic dataset will be illustrated below. The other datasets used are as follows– *MNIST-USPS*: Similar to previous work in deep fair clustering [212], we construct *MNIST-USPS* dataset using all the training digital samples from MNIST [127] and USPS dataset [213], and set the sample source as the protected attribute (MNIST/USPS). *Office-31*: The *Office-31* dataset [214] was originally used for domain adaptation and contains images from 31 different categories with three distinct source domains: Amazon, Webcam, and DSLR. Each domain contains all the categories but with different shooting angles, lighting conditions, etc. We use DSLR and Webcam for our experiments and let the domain source be the protected attribute for this dataset. We also conduct experiments on the *Inverted UCI DIGITS* [215] and *Extended Yale Face B* datasets [156].



(a) Pre-attack

(b) Post-attack

FIGURE 4.1. Pre-attack and post-attack clusters of the SFD fair clustering algorithm on the synthetic toy data. The labels of Cluster A and Cluster B are shown in green and blue, and these samples in two clusters belong to G_D. The $\circ$ and $\triangle$ markers represent the two protected groups, and points in red are the attack points that belong to G_A. Observe that before the attack, the SFD algorithm obtains a perfect Balance of 1.0. However, after the attack, once the attacker has optimized the protected group memberships for the attack points, the SFD clustering has become less fair with Balance = 0.5.

4.1.3.2. *Fair Clustering Models.* We include three state-of-the-art fair clustering algorithms: Fair K-Center (KFC) [216], Fair Spectral Clustering (FSC) [103] and Scalable Fairlet Decomposition (SFD) [100] for fairness attack, and these methods employ different traditional clustering algorithms on the backend: KFC uses k-center, SFD uses k-median, and FSC uses spectral clustering.

We implemented the FSC, SFD, and KFC fair algorithms in Python, using the authors' implementations as a reference in case they were not written in Python. To this end, we generally default to using the hyperparameters for these algorithms as provided in the original implementations. However, if needed, we also tuned the hyperparameter values so as to maximize performance on the unsupervised fairness metrics (such as Balance) as this allows us to attack fairness better. Note that this is still an unsupervised parameter selection strategy as Balance is a fully unsupervised metric, as it takes only the clustering outputs and protected group memberships as input, which are also provided as input to the fair clustering algorithms.[1] Such parameter tuning has also been done in previous fair clustering work [103, 217].

For SFD, we set the parameters $p = 2, q = 5$ for all datasets except *DIGITS* for which we set $p = 1, q = 5$. For FSC we use the default parameters and use the nearest neighbors approach [208]

[1]Tuning hyperparameters using NMI/ACC or other performance metrics that take the ground truth cluster labels as input, would violate the unsupervised nature of the clustering problem.

for creating the input graph for which we set the number of neighbors = 3 for all datasets. For KFC we use the default parameter value of $\delta = 0.1$.

4.1.3.3. *Protocol.* Fair clustering algorithms, much like their traditional counterparts, are extremely sensitive to initialization [218]. Differences in the chosen random seed can lead to widely different fair clustering outputs. Thus, we use 10 different random seeds when running the SFD, FSC, and KFC fair algorithms and obtain results. We also uniformly randomly sampled G_A and G_D initially to select these sets such that the fairness utility (i.e. Balance) before the attack is a reasonably high enough value to attack. The size of G_A is varied from 0% to 30% to see how this affects the attack trends. Furthermore, for the zeroth-order attack optimization, we always attack the Balance metric (unless the fair algorithm always achieves 0 Balance in which case we attack Entropy). Note that Balance is a harsher fairness metric than Entropy and hence should lead to a more successful attack. As a performance baseline, we also compare with a random attack where instead of optimizing G_A to reduce fairness utility on G_D, we uniformly randomly pick group memberships in G_A.

4.1.3.4. *Evaluation.* We use four metrics along two dimensions: fairness utility and clustering utility for performance evaluation. For clustering utility, we consider Unsupervised Accuracy (ACC) [132], and Normalized Mutual Information (NMI) [130]. For fairness utility we consider Balance [85] and Entropy [212]. These four metrics are commonly used in the fair clustering literature. Note that for each of these metrics, the higher the value obtained, the better the utility. For fairness, Balance is a better metric to attack, because a value of 0 means that there is a cluster that has 0 samples from one or more protected groups. Finally, the attacker does not care about the clustering utility as long as changes in utility do not reveal that an attack has occurred. We define Balance (the earlier definition was framed as a cost) and Entropy below for reference.

Balance: Balance is a fairness metric proposed by [85] which lies between 0 (least fair) and 1 (most fair). Let there be m protected groups for a given dataset X. Then, define r_X^g and r_k^g to be the proportion of samples of the dataset belonging to protected group g and the proportion of samples in cluster $k \in [K]$ belonging to protected group g. The Balance fairness notion is then defined over all clusters and protected groups as:

$$\text{Balance} = \min_{k\in[K],g\in[m]} \min\{\frac{r_X^g}{r_k^g}, \frac{r_k^g}{r_X^g}\}.$$

Entropy: Entropy is a fairness metric proposed by [212] and similar to Balance, higher values of Entropy, mean that clusters have more fairness. Let $N_{k,g}$ be the set containing the samples of the dataset X that belong to both the cluster $k \in [K]$ and the protected group g. Further, let n_k be the number of samples in cluster k. Then Entropy for group g is defined as follows:

$$\text{Entropy}(g) = -\sum_{k\in[K]} \frac{|N_{k,g}|}{n_k} \log \frac{|N_{k,g}|}{n_k}.$$

Note that in this work, we take the average Entropy over all groups.

4.1.3.5. *Performance on Toy Data.* To demonstrate the effectiveness of the poisoning attack, we also generate a 2-dimensional 20-sample synthetic toy dataset using an isotropic Gaussian distribution, with standard deviation = 0.12, and centers located at (4,0) and (4.5, 0). Out of these 20 points, we designate 14 to belong to G_D and the remaining 6 to belong to G_A. The number of clusters is set to $k = 2$. These are visualized in Figure 4.1. We generate cluster outputs using the SFD fair clustering algorithm before the attack (Figure 4.1a), and after the attack (Figure 4.1b). Before the attack, SFD achieves perfect fairness with a Balance of 1.0 as for each protected group and both Cluster A and Cluster B, we have Balance $\frac{4/8}{7/14} = 1.0$ and $\frac{3/6}{7/14} = 1.0$, respectively. Moreover, performance utility is also high with NMI = 0.695 and ACC = 0.928. However, after the attack, fairness utility decreases significantly. The attacker changes protected group memberships of the attack points, and this leads to the SFD algorithm trying to find a more optimal *global* solution, but in that, it reduces fairness for the points belonging to G_D. Balance drops to 0.5 as for Cluster A and Protected Group 0 we have Balance $\frac{1/4}{7/14} = 0.5$, leading to a 50% decrease. Entropy also drops down to 0.617 from 0.693. Performance utility decreases in this case, but is still satisfactory with NMI = 0.397 and ACC = 0.785. Thus, it can be seen that our attack can disrupt the fairness of fair clustering algorithms significantly.

4.1.3.6. *Performance on Real-World Datasets.* We show the pre-attack and post-attack results *MNIST-USPS* and *Office-31* by our attack and random attack in Figure 4.2. It can be observed that our fairness attack consistently outperforms the random attack baseline in terms of both fairness

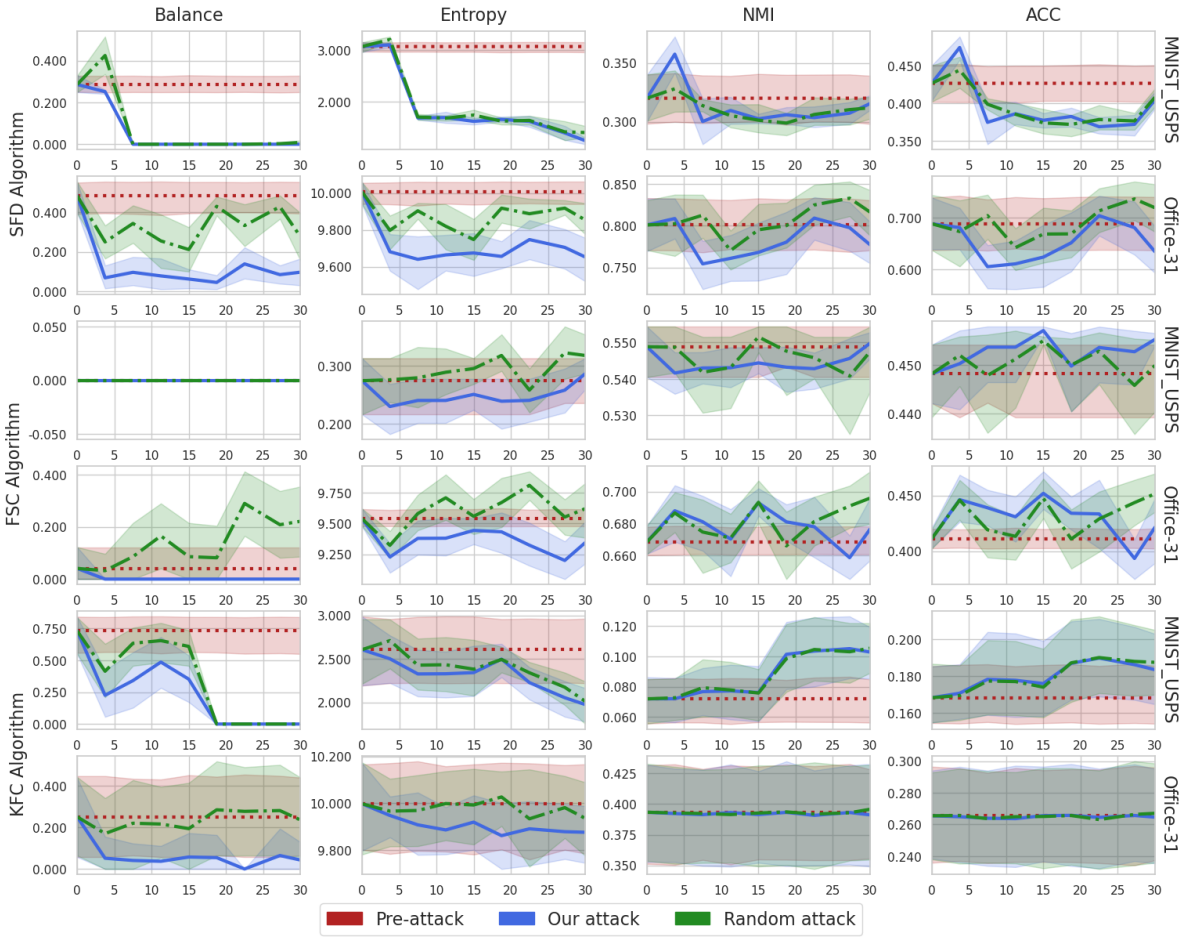


FIGURE 4.2. Attack results for *MNIST-USPS* & *Office-31* (x-axis: % of samples attacker can poison).

metrics: Balance and Entropy. Further, our attack always leads to lower fairness metric values than the pre-attack values obtained, while this is often not the case for the random attack, Balance and Entropy increase for the random attack on the FSC algorithm on *Office-31* dataset. Interestingly, even though we do not optimize for this, clustering performance utility (NMI and ACC) does not drop significantly and even increases frequently. For example, NMI/ACC for FSC on *Office-31* (Figure 4.2, Row 4, Column 3-4) and NMI/ACC for KFC on *MNIST-USPS* (Figure 4.2, Row 5, Column 3-4). Thus, the attacker can easily subvert the defense as the clustering performance before and after the attack does not decrease drastically and at times even increases. We also conduct the Kolmogorov-Smirnov statistical test [219] between our attack and the random attack result distribution for the fairness utility metrics (Balance and Entropy) to see if the mean distributions are

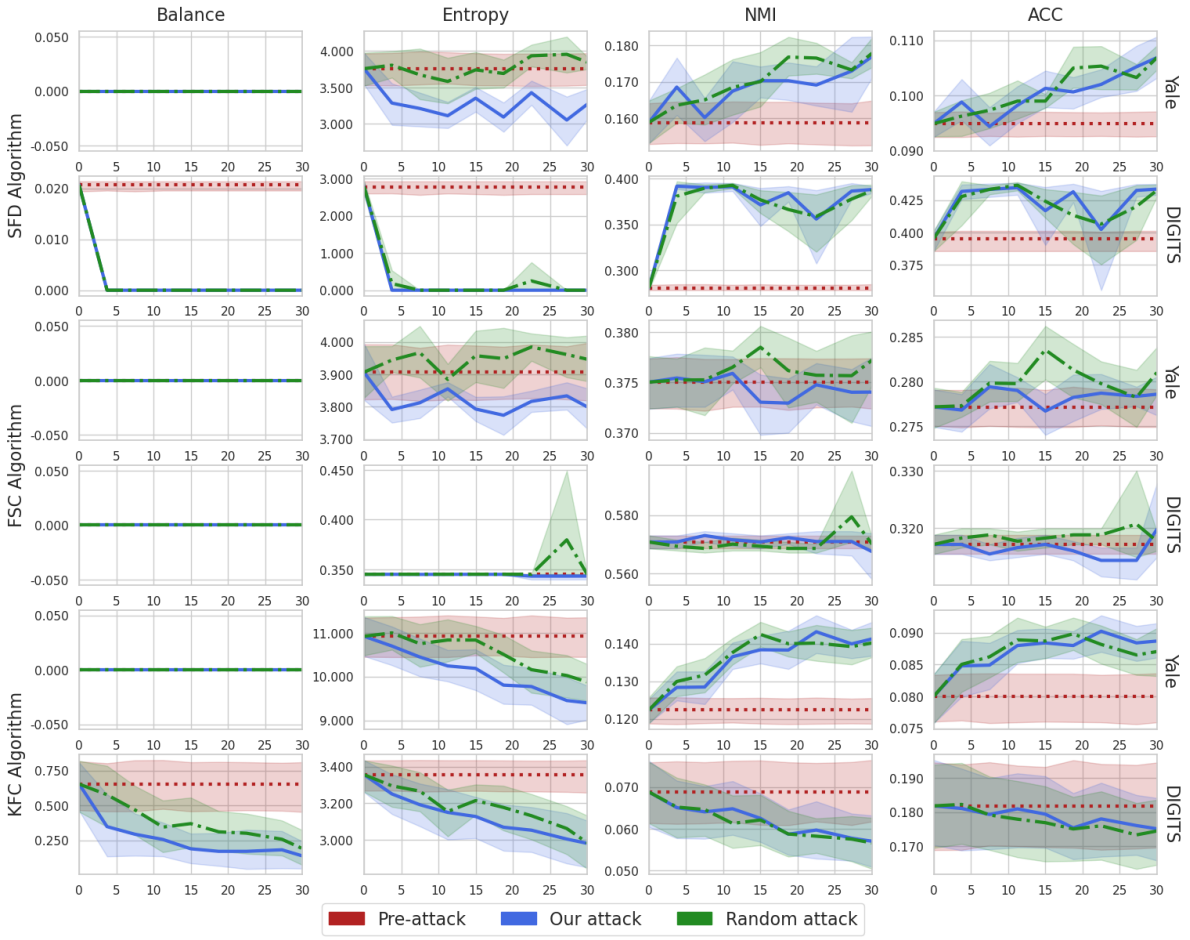


FIGURE 4.3. Attack results for the *Extended Yale Face B* and *Inverted UCI DIGITS* datasets (x-axis denotes % of samples attacker can poison).

significantly different. These results are shown in Table 4.2. We find that for the *Office-31* dataset our attack is statistically significant in terms of fairness values and obtains p-values of less than < 0.01. For *MNIST-USPS*, the results are also statistically significant except for the cases when the utility reduces quickly to the same value. For example, it can be seen that Balance goes to 0 for SFD on *MNIST-USPS* in Figure 4.2 (Row 1, Column 1) fairly quickly. For these distributions, it is intuitive why we cannot obtain statistically significant results, as the two attack distributions become identical. We also provide the attack performance on *Inverted UCI DIGITS dataset* [215] and *Extended Yale Face B* dataset [156] in Figure 4.3.

Furthermore, to better compare our attack with the random attack, we present the results when 15% group memberships are switched in Table 4.1 for *Office-31* and *MNIST-USPS*. As can be seen,

TABLE 4.1. Results for our attack and random attack when 15% group membership labels are switched.

Algorithms	Metrics	*MNIST-USPS*				
		Pre-Attack	Our Attack	Change (%)	Random Attack	Change (%)
SFD	Balance	0.286 ± 0.065	0.000 ± 0.000	(-)100.0	0.000 ± 0.000	(-)100.0
	Entropy	3.070 ± 0.155	1.621 ± 0.108	(-)47.12	1.743 ± 0.156	(-)43.23
	NMI	0.320 ± 0.033	0.302 ± 0.007	(-)5.488	0.301 ± 0.019	(-)5.847
	ACC	0.427 ± 0.040	0.378 ± 0.015	(-)11.54	0.374 ± 0.026	(-)12.35
FSC	Balance	0.000 ± 0.000	0.000 ± 0.000	(-)100.0	0.000 ± 0.000	(-)100.0
	Entropy	0.275 ± 0.077	0.251 ± 0.041	(-)8.576	0.295 ± 0.036	(+)7.598
	NMI	0.549 ± 0.011	0.544 ± 0.007	(-)0.807	0.552 ± 0.006	(+)0.491
	ACC	0.448 ± 0.012	0.457 ± 0.002	(+)1.971	0.455 ± 0.002	(+)1.510
KFC	Balance	0.730 ± 0.250	0.352 ± 0.307	(-)51.87	0.608 ± 0.237	(-)16.72
	Entropy	2.607 ± 0.607	2.343 ± 0.444	(-)10.12	2.383 ± 0.490	(-)8.595
	NMI	0.072 ± 0.024	0.076 ± 0.029	(+)5.812	0.076 ± 0.027	(+)5.330
	ACC	0.168 ± 0.026	0.176 ± 0.034	(+)4.581	0.174 ± 0.031	(+)3.419
Algorithms	Metrics	*Office-31*				
		Pre-Attack	Our Attack	Change (%)	Random Attack	Change (%)
SFD	Balance	0.484 ± 0.129	0.062 ± 0.080	(-)87.21	0.212 ± 0.188	(-)56.34
	Entropy	10.01 ± 0.098	9.675 ± 0.187	(-)3.309	9.748 ± 0.181	(-)2.581
	NMI	0.801 ± 0.050	0.768 ± 0.058	(-)4.110	0.795 ± 0.053	(-)0.726
	ACC	0.688 ± 0.082	0.624 ± 0.098	(-)9.397	0.668 ± 0.091	(-)2.908
FSC	Balance	0.041 ± 0.122	0.000 ± 0.000	(-)100.0	0.086 ± 0.172	(+)110.8
	Entropy	9.538 ± 0.113	9.443 ± 0.178	(-)0.997	9.558 ± 0.226	(+)0.207
	NMI	0.669 ± 0.014	0.693 ± 0.014	(+)3.659	0.693 ± 0.022	(+)3.711
	ACC	0.411 ± 0.014	0.452 ± 0.027	(+)9.904	0.447 ± 0.030	(+)8.817
KFC	Balance	0.250 ± 0.310	0.057 ± 0.172	(-)77.07	0.194 ± 0.301	(-)22.55
	Entropy	9.997 ± 0.315	9.919 ± 0.189	(-)0.786	9.992 ± 0.250	(-)0.051
	NMI	0.393 ± 0.064	0.391 ± 0.063	(-)0.483	0.393 ± 0.067	(-)0.160
	ACC	0.265 ± 0.048	0.266 ± 0.049	(+)0.032	0.265 ± 0.051	(-)0.162

TABLE 4.2. KS test statistic values comparing our attack distribution with the random attack distribution for the Balance and Entropy metrics (** indicates statistical significance i.e., p-value < 0.01).

Dataset	Algorithm	Balance	Entropy
Office-31	SFD	0.889**	0.889**
Office-31	FSC	0.889**	0.778**
Office-31	KFC	0.889**	0.778**
MNIST-USPS	SFD	0.222	0.222
MNIST-USPS	FSC	0.000	0.778**
MNIST-USPS	KFC	0.333	0.333

for all fair clustering algorithms and datasets, our attack leads to a more significant reduction in fairness utility for both the *MNIST-USPS* and *Office-31* datasets. In fact, as mentioned before, the random attack at times leads to an *increase* in fairness utility compared to before the attack

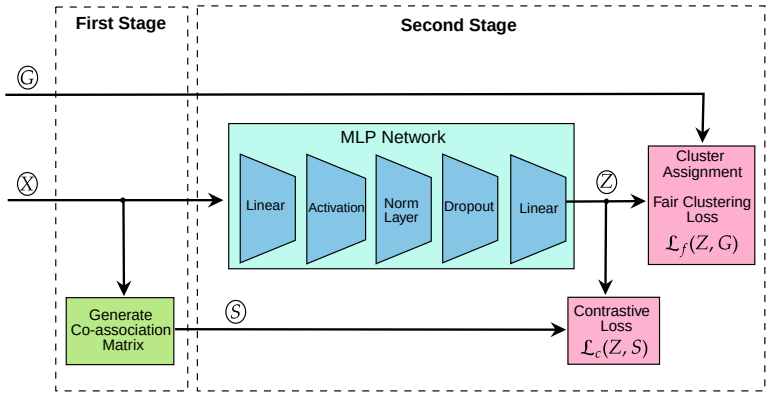


FIGURE 4.4. Our proposed CFC framework.

(refer to FSC Balance/Entropy on *Office-31*). In contrast, our attack always leads to a reduction in fairness performance. For example, for the KFC algorithm and *Office-31* dataset, our attack achieves a reduction in Balance of 77.07% whereas the random attack reduces Balance by only 22.52%. However, it is important to note here that existing fair clustering algorithms are very *volatile* in terms of performance, as the random attack can also lead to fairness performance drops, especially for the SFD algorithm (refer to Figure 4.2 for visual analysis). This further motivates the need for more robust fair clustering.

4.1.4. Fairness Defense: Consensus Fair Clustering. We now provide details on our defense against the proposed attack. In accordance with the fairness attack defined as Definition 4.1.1, we also provide a definition for *robust fair clustering*, followed by our proposed defense algorithm.

DEFINITION 4.1.2 (Robust Fair Clustering). *Given the dataset X, samples' protected groups G, and $G_A \subseteq G$ is a small portion of protected groups that an adversary can control, a fair clustering algorithm $\mathcal{F}$ is considered to be robust to the fairness attack if the change in fairness utility on $G_D = G \setminus G_A \subseteq G$ before the attack and after the attack is by a marginal amount, or if the fairness utility on G_D increases after the attack.*

To achieve robust fair clustering, our defense utilizes consensus clustering [220, 221, 222, 223] combined with fairness constraints to ensure that cluster outputs are robust to the attack. Consensus clustering has been widely renowned for its robustness and consistency properties but to the best of our knowledge, no other work has utilized consensus clustering concepts for fair clustering.

Specifically, we propose Consensus Fair Clustering (CFC) shown in Figure 4.4, which first transforms the consensus clustering problem into a graph partitioning problem, and then utilizes a novel graph-based neural network architecture to learn representations for fair clustering. CFC has two stages to tackle the attack challenge at the data and algorithm level. The first stage is to sample a subset of training data and run cluster analysis to obtain the basic partition and the co-association matrix. Since poisoned samples are a tiny portion of the whole training data, the probability of these being selected into the subset is also small, which decreases their negative impact. In the second stage, CFC fuses the basic partitions with a fairness constraint and further enhances the algorithmic robustness.

First Stage: Generating Co-Association Matrix. In this first stage of CFC, we will generate r basic partitions $\Pi = \{\pi_1, \pi_2, ..., \pi_r\}$. For each basic partition π_i, we first get a sub dataset X_i by random sample/feature selection and run K-means [78] to obtain a basic partition π_i. Such a process is repeated r times such that $\cup_{i=1}^{r} X_i = X$. Given that u and v are two samples and $\pi_i(u)$ is the label category of u in basic partition π_i, and following the procedure of consensus clustering, we summarize the basic partitions into a co-association matrix $S \in \mathbb{R}^{n \times n}$ as $S_{uv} = \sum_{i=1}^{r} \delta(\pi_i(u), \pi_i(v))$, where $\delta(a, b) = 1$ if $a = b$; otherwise, $\delta(a, b) = 0$. The co-association matrix not only summarizes the categorical information of basic partitions into a pair-wise relationship, but also provides an opportunity to transform consensus clustering into a graph partitioning problem, where we can learn a fair graph embedding that is resilient to the protected group membership poisoning attack.

Second Stage: Learning Graph Embeddings for Fair Clustering. In the second stage of CFC, we aim to find an optimal consensus and fair partition based on the feature matrix X, basic partitions Π, and sample sensitive attributes G. The objective function of our CFC consists of a self-supervised contrastive loss, a fair clustering loss, and a structural preservation loss.

Self-supervised Contrastive Loss. To learn a fair graph embedding using X, S, and G, we use a few components inspired by a recently proposed simple graph classification framework called Graph-MLP [224], which does not require message-passing between nodes and outperforms the classical message-passing GNN methods in various tasks [225, 226]. Specifically, it employs the neighboring contrastiveness and considers the R-hop neighbors to each node as the positive samples, and the remaining nodes as negative samples. The loss ensures that positive samples remain closer

to the node, and negative samples remain farther away based on feature distance. Let $\gamma_{uv} = S^R_{uv}$ and S is the co-association matrix, sim denote cosine similarity, and τ be the temperature parameter, then we can write the loss as follows:

$$\mathcal{L}_c(Z, S) \coloneqq -\frac{1}{n}\sum_{i=1}^{n} \log \frac{\sum_{a=1}^{n} \mathbb{1}_{[a\neq i]}\gamma_{ia} \exp(sim(Z_i, Z_a)/\tau)}{\sum_{b=1}^{n} \mathbb{1}_{[b\neq i]} \exp(sim(Z_i, Z_b)/\tau)}. \tag{4.2}$$

Fair Clustering Loss. Similar to other deep clustering approaches [43, 212], we employ a clustering assignment layer based on Student t-distribution and obtain soft cluster assignments P. We also include a fairness regularization term using an auxiliary target distribution Q to ensure that the cluster assignments obtained from the learned embeddings $z \in Z$ are fair. We abuse notation slightly and denote the corresponding learned representation of sample $x \in X$ as $z_x \in Z$. Also let p^x_k represent the probability of sample $x \in X$ being assigned to the k-th cluster, $\forall k \in [K]$. More precisely, p^x_k represents the assigned confidence between representation z_x and cluster centroid c_k in the embedding space. The fair clustering loss term can then be written as:

$$\mathcal{L}_f(Z, G) \coloneqq KL(P||Q) = \sum_{g\in[L]}\sum_{x\in H_g}\sum_{k\in[K]} p^x_k \log \frac{p^x_k}{q^x_k}, \tag{4.3}$$

where $p^x_k = \frac{(1+||z_x - c_k||^2)^{-1}}{\sum_{k'\in[K]}(1+||z_x - c_{k'}||^2)^{-1}}$ and $q^x_k = \frac{(p^x_k)^2/\sum_{x'\in H_{g(x)}} p^{x'}_k}{\sum_{k'\in[K]}(p^x_{k'})^2/\sum_{x'\in H_{g(x)}} p^{x'}_{k'}}$.

Structural Preservation Loss. Since optimizing the fair clustering loss $\mathcal{L}_f$ can lead to a degenerate solution where the learned representation reduces to a constant function [212], we employ a well-known structural preservation loss term for each protected group. Since this loss is applied to the final partitions obtained we omit it for clarity from Figure 4.4 which shows the internal CFC architecture. Let P_g be the obtained soft cluster assignments for protected group g using CFC, and J_g be the cluster assignments for group g obtained using any other well-performing fair clustering algorithm. We can then define this loss as originally proposed in [212]:

$$\mathcal{L}_p \coloneqq \sum_{g\in[L]} ||P_g P_g^\top - J_g J_g^\top||^2. \tag{4.4}$$

TABLE 4.3. Pre/post-attack performance when 15% group membership labels are switched for *Office-31* and *MNIST-USPS* datasets.

Algorithms	Metrics	*Office-31*			*MNIST-USPS*		
		Pre-Attack	Post Attack	Change (%)	Pre-Attack	Post Attack	Change (%)
CFC	Balance	0.609 ± 0.081	0.606 ± 0.047	(-)0.466	0.442 ± 0.002	0.373 ± 0.090	(-)15.54
	Entropy	5.995 ± 0.325	6.009 ± 0.270	(+)0.229	2.576 ± 0.088	2.626 ± 0.136	(+)1.952
	NMI	0.690 ± 0.019	0.698 ± 0.013	(+)1.236	0.269 ± 0.007	0.287 ± 0.008	(+)6.749
	ACC	0.508 ± 0.021	0.511 ± 0.018	(+)0.643	0.385 ± 0.002	0.405 ± 0.015	(+)5.343
SFD	Balance	0.484 ± 0.129	0.062 ± 0.080	(-)87.21	0.286 ± 0.065	0.000 ± 0.000	(-)100.0
	Entropy	10.01 ± 0.098	9.675 ± 0.187	(-)3.309	3.070 ± 0.155	1.621 ± 0.108	(-)47.19
	NMI	0.801 ± 0.050	0.768 ± 0.058	(-)4.110	0.320 ± 0.033	0.302 ± 0.007	(-)5.488
	ACC	0.688 ± 0.082	0.624 ± 0.098	(-)9.397	0.427 ± 0.040	0.378 ± 0.015	(-)11.54
FSC	Balance	0.041 ± 0.122	0.000 ± 0.000	(-)100.0	0.000 ± 0.000	0.000 ± 0.000	(-)100.0
	Entropy	9.538 ± 0.113	9.443 ± 0.178	(-)0.997	0.275 ± 0.077	0.251 ± 0.041	(-)8.576
	NMI	0.669 ± 0.014	0.693 ± 0.014	(+)3.659	0.549 ± 0.011	0.544 ± 0.007	(-)0.807
	ACC	0.411 ± 0.014	0.452 ± 0.027	(+)9.904	0.448 ± 0.012	0.457 ± 0.002	(+)1.971
KFC	Balance	0.250 ± 0.310	0.057 ± 0.172	(-)77.07	0.730 ± 0.250	0.352 ± 0.307	(-)51.87
	Entropy	9.997 ± 0.315	9.919 ± 0.189	(-)0.786	2.607 ± 0.607	2.343 ± 0.444	(-)10.12
	NMI	0.393 ± 0.064	0.391 ± 0.063	(-)0.483	0.072 ± 0.024	0.076 ± 0.029	(+)5.812
	ACC	0.265 ± 0.048	0.266 ± 0.049	(+)0.032	0.168 ± 0.026	0.176 ± 0.034	(+)4.581

The overall objective for CFC algorithm can be written as $\mathcal{L}_c + \alpha\mathcal{L}_f + \beta\mathcal{L}_p$, where α, β are parameters used to control trade-off between individual losses. CFC can then be used to generate hard cluster label predictions using the soft cluster assignments $P \in \mathbb{R}^{n \times K}$.

4.1.5. Results for the Defense. To showcase the efficacy of our CFC defense algorithm, we utilize the same datasets and fair clustering algorithms considered in the experiments for the attack section. Specifically, we show results when 15% of protected group membership labels can be switched for the adversary in Table 4.3 (over 10 individual runs) for *MNIST-USPS* and *Office-31*. The results for *Inverted UCI DIGITS* and *Extended Yale Face B* datasets show similar trends and are provided in Table 4.4. Here, we present pre-attack and post-attack fairness utility (Balance, Entropy) and clustering utility (NMI, ACC) values. We also denote the percent change in these evaluation metrics before the attack and after the attack for further analysis. The detailed implementation and hyperparameter choices for CFC are as follows:

- The hyperparameters such as number of basic partitions r, temperature parameter τ in the contrastive loss $\mathcal{L}_c$, dropout in hidden layers, number of training epochs, the activation

TABLE 4.4. Pre/post-attack performance when 15% group membership labels are switched for *DIGITS* and *Yale* datasets.

Algorithms	Metrics	*DIGITS*			*Yale*		
		Pre-Attack	Post Attack	Change (%)	Pre-Attack	Post Attack	Change (%)
CFC	Balance	0.145 ± 0.002	0.266 ± 0.189	(+)83.62	0.176 ± 0.027	0.047 ± 0.067	(-)73.22
	Entropy	1.963 ± 0.022	2.096 ± 0.153	(+)6.758	5.472 ± 0.669	6.196 ± 0.441	(+)13.22
	NMI	0.225 ± 0.004	0.187 ± 0.013	(-)16.72	0.142 ± 0.004	0.170 ± 0.013	(+)20.16
	ACC	0.287 ± 0.006	0.270 ± 0.008	(-)6.132	0.099 ± 0.003	0.108 ± 0.007	(+)9.381
SFD	Balance	0.021 ± 0.002	0.000 ± 0.000	(-)100.0	0.000 ± 0.000	0.000 ± 0.000	(-)100.0
	Entropy	2.781 ± 0.286	0.000 ± 0.000	(-)100.0	3.757 ± 0.358	3.351 ± 0.277	(-)10.81
	NMI	0.281 ± 0.005	0.371 ± 0.032	(+)32.21	0.159 ± 0.009	0.170 ± 0.007	(+)7.206
	ACC	0.395 ± 0.014	0.417 ± 0.038	(+)5.440	0.095 ± 0.004	0.101 ± 0.005	(+)6.733
FSC	Balance	0.000 ± 0.000	0.000 ± 0.000	(-)100.0	0.000 ± 0.000	0.000 ± 0.000	(-)100.0
	Entropy	0.345 ± 0.000	0.345 ± 0.000	(-)0.000	3.907 ± 0.137	3.793 ± 0.060	(-)2.912
	NMI	0.571 ± 0.004	0.571 ± 0.004	(-)0.000	0.375 ± 0.004	0.373 ± 0.005	(-)0.530
	ACC	0.317 ± 0.003	0.317 ± 0.003	(-)0.000	0.277 ± 0.004	0.277 ± 0.004	(-)0.171
KFC	Balance	0.653 ± 0.290	0.188 ± 0.167	(-)71.14	0.000 ± 0.000	0.000 ± 0.000	(-)100.0
	Entropy	3.356 ± 0.138	3.128 ± 0.193	(-)6.815	10.92 ± 0.744	10.19 ± 0.742	(-)6.730
	NMI	0.069 ± 0.012	0.062 ± 0.010	(-)9.184	0.122 ± 0.006	0.138 ± 0.006	(+)13.08
	ACC	0.182 ± 0.020	0.179 ± 0.017	(-)1.333	0.080 ± 0.006	0.088 ± 0.004	(+)10.50

function, and fair clustering algorithm used to generate J for structural preservation loss $\mathcal{L}_p$ are set to be the same across all datasets.

- These are $r = 100, \tau = 2,$ dropout $= 0.6,$ # epochs $= 3000,$ Gaussian Error Linear Unit [227] is used as the activation function, and we use SFD with default parameters for generating J since it runs faster than other fair clustering algorithms.
- The dimension of the hidden layer is set to 256 for all datasets except for *DIGITS* since *DIGITS* has only 64 features and hence we use the hidden layer dimension as 36 for it.
- We tune the other hyperparameters for the different datasets to optimize for fairness performance. Using grid based search we set the following parameters for the given datasets: for *Office-31* we have $R = 1, \alpha = 1, \beta = 100$; for *MNIST-USPS* we have $R = 2, \alpha = 100, \beta = 25$; for *Yale* we have $R = 2, \alpha = 50, \beta = 10$; and for *DIGITS* we have $R = 2, \alpha = 10, \beta = 50$.

As can be seen in Table 4.3, our CFC clustering algorithm achieves fairness utility and clustering performance utility superlative to the other SOTA fair clustering algorithms. In particular, CFC optimizes for fairness utility and clustering performance utility jointly, which is not the case for the other fair clustering algorithms. Post-attack performance of CFC on all datasets is also always better

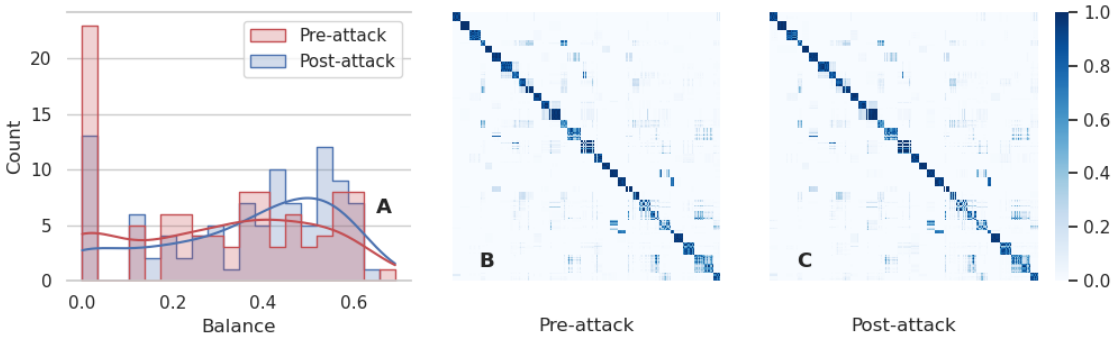


FIGURE 4.5. (A) Histogram representing the distribution of Balance of the Basic Partitions (BPs) before the attack and after the attack. (B) Visualization of the consensus matrix before the attack. (C) Visualization of the consensus matrix after the attack.

compared to the other fair algorithms where Balance often drops close to 0. This indicates that fairness utility has completely been disrupted for these algorithms and the adversary is successful. For CFC, post-attack fairness and performance values are at par with their pre-attack values, and at times even better than before the attack. For example, for Entropy, NMI, and ACC metrics, CFC has even better fairness and clustering performance after the attack than before it is undertaken on both the *Office-31* and *MNIST-USPS* datasets. Balance also decreases only by a marginal amount. Whereas for the other fair clustering algorithms SFD, FSC, and KFC, fairness has been completely disrupted through the poisoning attack. For all the other algorithms, Entropy and Balance decrease significantly with more than 10% and 85% decrease on average, respectively. Next, we analyze CFC in-depth with regards to its stages and overall robustness.

4.1.5.1. *Analyzing the Consensus Clustering Stage of CFC.* We undertake some additional analysis that sheds light on why the performance of CFC remains largely unaffected by the proposed fairness attack. We begin by analyzing the first stage of the CFC pipeline, i.e., the consensus matrix generation stage. In Figure 4.5(A), we plot the distribution of basic partitions' (BPs) Balance values before our fairness attack and after our fairness attack on *Office-31*, as a histogram. Note that $r = 100$, which means that we have 100 basic partitions. It can be seen that before the attack there are more BPs with 0 Balance values, and after the attack these partitions actually decrease. Specifically, the mean Balance of the BPs shifts from 0.3 before the attack to 0.35 after the attack. Moreover, the partition at the 20th percentile has Balance 0 before the attack, but 20th percentile

BP improves to a Balance of 0.14 after the attack. This indicates that the simple basic partition generation strategy will alleviate the negative impact of a fairness attack.

Moreover, it is beneficial to use consensus between BPs as a means of ensuring robustness, i.e, our model is able to generalize from the performance of a number of different clustering results to obtain more robust results. This can also be observed through Figures 4.5(B) and 4.5(C), where we plot the pre-attack and post-attack consensus matrices obtained for *Office-31*, respectively. Visually, both these matrices look similar, indicating that the consensus matrix is largely unaffected by the attack. Note that for the size of these matrices ($n \times n$, where $n = 1,293$ is the number of samples in the *Office-31* dataset), the norm of their difference equals to 19.335, which is relatively small with respect to the number of samples. This indicates that consensus clustering results as part of the first stage are independent of the attack, and hence, can be used to ensure highly resilient and robust performance on the dataset before and after the attack.

4.1.5.2. *Analyzing Overall Adversarial Robustness of CFC.* Next, we conduct experiments on comparing the performance of CFC with the other SOTA fair clustering algorithms. For ease of understanding, we plot the ratio between the mean post-attack and pre-attack values as a function of the percentage of protected group membership labels switched by the attacker. Thus, the ratio is mathematically defined as $\frac{\text{Mean Post-Attack Value}}{\text{Mean Pre-Attack Value}}$. Then, note that higher values of the ratio, indicate more robust performance with regards to fairness metrics such as Balance or Entropy.

We present these results in Figure 4.6. As can be observed, the CFC ratio values are always much higher than the other algorithms for all the attack percentages and for the fairness metrics Balance and Entropy. This is especially observable for certain datasets (such as Yale), where Balance for all other fair algorithms is consistently 0, but CFC is still able to obtain clustering solutions with desirable fairness utility before and after the attack. It is also worthwhile to note that for *Office-31* and *MNIST-USPS*, fairness performance is highly robust as the ratio trends tend to be approximately 1.0, or >1.0, with little to no decrease in utility after the attack. For the NMI and ACC metrics, we find that the ratio is generally distributed very close to 1.0, indicating that clustering performance is very similar before and after the attack. This means that in general, it is hard to tell whether or not a fairness attack has occurred based on clustering performance. This makes it challenging for the defender to defend against such an attack, further mandating the need for robust fair clustering

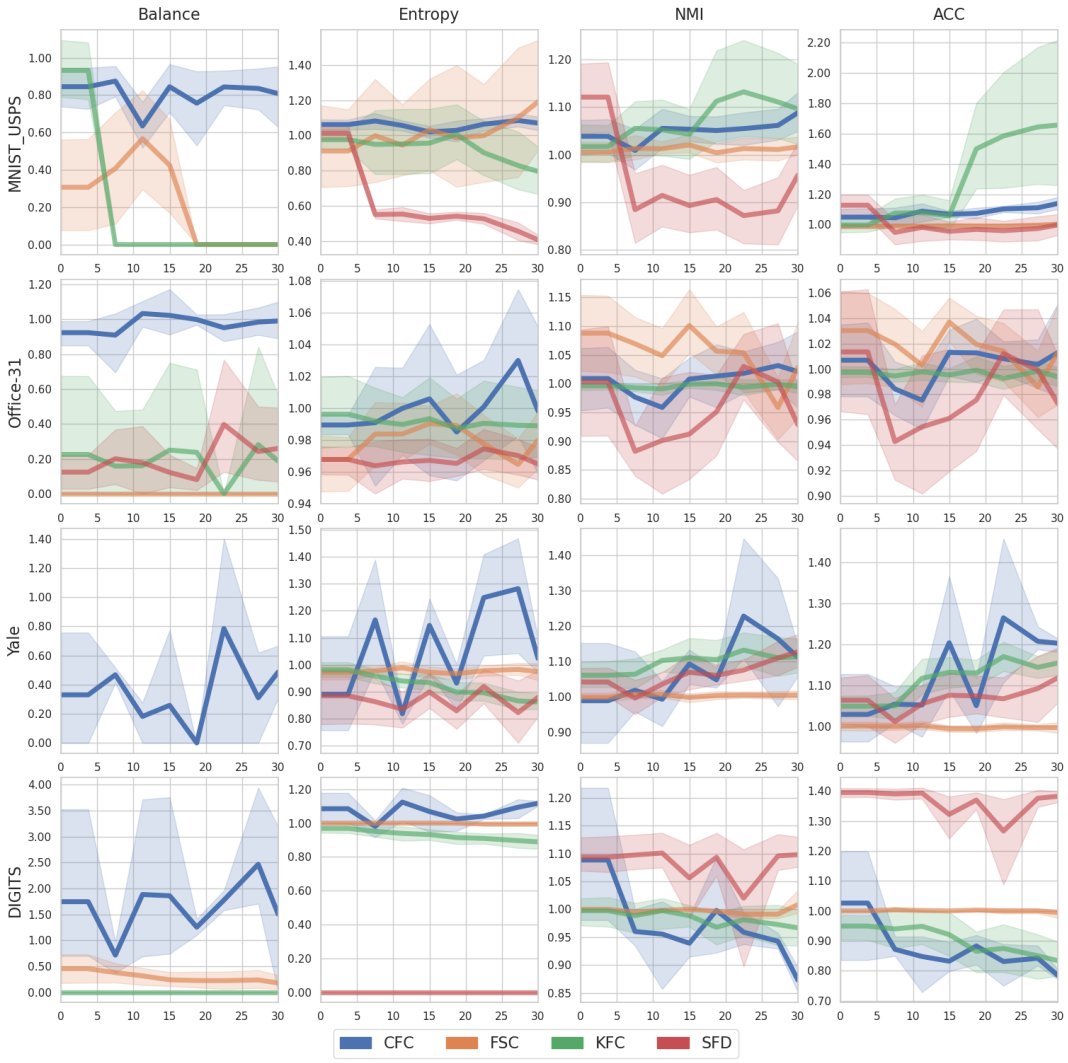


FIGURE 4.6. Post/Pre attack ratio trends for CFC and other fair clustering algorithms (we do not plot curves for which pre-attack values are 0). X-axis denotes the % of samples attacker can poison.

algorithms like CFC. Also note that for some algorithms, pre-attack and/or post-attack values are consistently 0, and we omit their trends from the figure since they are indeterminate.

4.2. Enhancing Classifier Performance and Interpretability through Influence-Based Data Selection

4.2.1. Introduction. ML models have become essential tools in various societal applications for automating processes and generating insights [228, 229]. Along with the choice of model

type/architecture, data is a critical component of the learning process, and the quality and quantity of training data have significant impacts on the model performance [230]. Despite this, current research mainly focuses on proposing high-performing model architectures or learning approaches while keeping the training data fixed. However, it is evident that not every sample in the training set augments model performance. Furthermore, the same data sample could be beneficial or detrimental to performance depending on the type of model being used [230]. Therefore, in this work, we aim to answer the research question "*what data benefits the learning model in a certain aspect?*" and select suitable training data to improve model performance with respect to utility,[2] fairness, and adversarial robustness.

Our work relates to but contrasts with the research on *data valuation*. Data valuation aims to assign a monetary value or worth to a particular set or collection of data, whereas our research investigates how data can be utilized to enhance a model. Data valuation can be performed in a variety of ways, such as using cooperative game theory (Shapley-value or Banzhaf index) [231, 232, 233, 234, 235] and reinforcement learning [236]. Shapley-value based data valuation approaches and their variants [231, 232] are model-agnostic, while our approaches are customized to the specific model being used. Moreover, these Shapley-value based approaches require the model to be retrained and evaluated multiple times, where even the most efficient known non-distributional algorithms [232] require $\mathcal{O}(\sqrt{n}\log(n)^2)$ model retraining steps, and n is the number of training samples. Furthermore, the current Shapley-value based frameworks can only estimate the data value in utility, leaving their estimations for fairness and robustness incomplete and unclear. Reinforcement learning-based data valuation [236] faces similar issues due to its complex design and poor convergence [237].

Our work is also closely related to other work on *data influence*, such as the seminal paper on influence functions by Koh and Liang [22], and influence-based data reweighing for improving fairness [238]. However, these methods only seek to trace model predictions back to the data, and do not directly answer our research question, which is to identify and interpret the feature space for improving the model's performance. Unlike these methods, our data selection approaches can uniquely handle scenarios where data is unlabeled such as in active learning applications. Moreover, our work considers utility, fairness, and robustness under diverse application scenarios, and our data

[2]We interchangeably use utility and accuracy in this section.

selection strategy can even boost the performance of influence-based methods such as [238]. Our work utilizes influence functions as a tool and extends their applicability to diverse domains and scenarios not considered previously [22].

Finally, our work also conceptually relates to several data-related areas, such as *data efficiency, feature selection, active learning*, and *antidote data*. Data efficiency approaches [239, 240, 241, 242] aim to accelerate deep model training by pruning data, which is beyond the scope of our work. Feature selection approaches [243, 244] aim to select important features for training, but are limited in scope as they only work for utility. Active learning [24, 245] partially aligns with our research question, as it involves selecting unlabeled data points to be annotated and added to the training set for improving model performance. However, its applicability is limited to this specific scenario, while our work considers a broader range of applications, including this one. Antidote data approaches [15, 18, 27] aim to add generated data to the training set for mitigating unfairness but cannot be used to interpret the usefulness of existing data points.

In summary, we propose data selection strategies based on influence functions [22] to improve a given classifier with respect to utility, fairness, and robustness. Our key idea is to use tree-based influence estimation models to understand and interpret which sample features contribute positively or negatively to the model's performance with respect to desired evaluation functions on a validation set. Additionally, we design a data selection strategy to achieve performance improvements. Our approaches can be utilized in multiple practical scenarios, where practitioners can either select data by trimming the existing training set, or by identifying and annotating new beneficial samples to include as part of the training set.

4.2.2. Preliminaries and Background. Here we first introduce influence functions and elaborate on how to use them to measure the sample influence on utility, fairness, and robustness.

Influence Functions. Given a training set $Z = \{(x_i, y_i)\}_{i=1}^n$ and a classifier trained using empirical risk minimization by a loss function ℓ, we have the optimal parameters of the classifier $\hat{\theta} = \text{argmin}_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^n \ell(x_i, y_i; \theta)$. Influence functions [22] measure the effect of changing an infinitesimal weight of samples on a validation set, based on an impact function f evaluating the quantity of interest, such as utility, fairness, and robustness. Downweighting a training sample x_j by a very small fraction ϵ leads the model parameter to $\hat{\theta}(x_j; -\epsilon) = \text{argmin}_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^n \ell(x_i, y_i; \theta) - \epsilon \ell(x_j, y_j; \theta)$.

The actual impact of such a change can be written as $\mathcal{I}^*(x_j; -\epsilon) = f(\hat{\theta}(x_j; -\epsilon)) - f(\hat{\theta})$, where $f(\hat{\theta}(x_j; -\epsilon))$ can be obtained by retraining the model without x_j. Assuming l is strictly convex and twice differentiable, and f is also differentiable, the actual impact can be estimated without the expensive model retraining by $\mathcal{I}(x_j; -\epsilon) = \lim_{\epsilon \to 0} f(\hat{\theta}(x_j; -\epsilon)) - f(\hat{\theta}) = \nabla_{\hat{\theta}} f(\hat{\theta})^\top \mathbf{H}_{\hat{\theta}}^{-1} \nabla_{\hat{\theta}} \ell(x_j, y_j; \hat{\theta})$, where $\mathbf{H}_{\hat{\theta}} = \sum_{i=1}^{n} \nabla_{\hat{\theta}}^2 \ell(x_j, y_j; \hat{\theta})$ is the Hessian matrix of ℓ and is invertible since ℓ is assumed to be convex. If we set $\epsilon = \frac{1}{n}$ and n is large ($n \to \infty$), we can write $\mathcal{I}(x_j; -\frac{1}{n}) = \mathcal{I}(-x_j)$. Thus, we can measure the influence of removing sample x_j from the training set.

Within the framework of influence functions, we instantiate the impact functions used to measure utility, fairness, and adversarial robustness of the model on a validation or test set.

Measuring Influence on Utility. If we instantiate the impact function f to calculate the loss value on a validation set V, we can measure a training sample influence on utility as follows:

$$\mathcal{I}^{\text{util}}(-x_i) = \sum_{(x,y) \in V} \nabla_{\hat{\theta}} \ell(x, y; \hat{\theta})^\top \mathbf{H}_{\hat{\theta}}^{-1} \nabla_{\hat{\theta}} \ell(x_i, y_i; \hat{\theta}). \tag{4.5}$$

Measuring Influence on Fairness. Similarly, we can instantiate the impact function f by group fairness [84, 187, 246], such as demographic parity (DP) or equal opportunity (EOP) to measure influence on fairness. Consider a binary sensitive attribute defined as $g \in \{0, 1\}$ and let $\hat{y}$ denote the predicted class probabilities. The fairness metric for DP is defined as: $f^{\text{DP-fair}}(\hat{\theta}, V) = \left|\mathbb{E}_V[\hat{y}|g = 1] - \mathbb{E}_V[\hat{y}|g = 0]\right|$ and for EOP as: $f^{\text{EOP-fair}}(\theta, V) = \Big|\mathbb{E}_V[l(x, y; \theta)|y = 1, g = 1] - \mathbb{E}_V[l(x, y; \theta)|y = 1, g = 0]\Big|$. Based on that, we can calculate the training sample influence on fairness as follows:

$$\mathcal{I}^{\text{DP-fair}}(-x_i) = \nabla_{\hat{\theta}} f^{\text{DP-fair}}(\hat{\theta}, V)^\top \mathbf{H}_{\hat{\theta}}^{-1} \nabla_{\hat{\theta}} \ell(x_i, y_i; \hat{\theta}). \tag{4.6}$$

$$\mathcal{I}^{\text{EOP-fair}}(-x_i) = \nabla_{\hat{\theta}} f^{\text{EOP-fair}}(\hat{\theta}, V)^\top \mathbf{H}_{\hat{\theta}}^{-1} \nabla_{\hat{\theta}} l(x_i, y_i; \hat{\theta}). \tag{4.7}$$

Measuring Influence on Adversarial Robustness. We can also measure which points contribute to adversarial robustness (or vulnerability) using influence functions. To do so, we first define an *adversary*– any attack approaches to craft adversarial samples can be used including black-box, white-box, among others. Here we consider a white-box adversary [247] specific to linear

models, which can be easily extended to other models and settings, such as FGSM [248], GDA [249], PGD [37]. To craft an adversarial sample, we take each sample x of the validation set V and perturb it as $x' = x - \gamma \frac{\hat{\theta}^\top x + b}{\hat{\theta}^\top \hat{\theta}} \hat{\theta}$, where $\hat{\theta} \in \mathbb{R}^d$ are the linear model coefficients, $b \in \mathbb{R}$ is the intercept, and $\gamma > 1$ is a parameter that controls the amount of perturbation added.

Since the decision boundary is a hyperplane, we simply move each sample orthogonal to it by adding minimal perturbation. In this manner, we can obtain an adversarial validation set V' which consists of x' for each sample x of V. The class labels y remain unchanged. Now, we can compute adversarial robustness influence for each training sample as follows:

$$\mathcal{I}^{\text{robust}}(-x_i) = \sum_{(x',y) \in V'} \nabla_{\hat{\theta}} \ell(x', y; \hat{\theta})^\top \mathbf{H}_{\hat{\theta}}^{-1} \nabla_{\hat{\theta}} \ell(x_i, y_i; \hat{\theta}). \tag{4.8}$$

Extension to Non-Convex Models. A current limitation of influence functions is that they require the model to satisfy strict convexity conditions, implying its Hessian is positive definite and invertible, and that it is trained to convergence [22]. To extend influence functions to non-convex models, several possible solutions exist: (1) a linear model (logistic regression) can be used as a surrogate on the embeddings obtained via the non-convex model [238]; (2) a damping term can be added to the non-convex model such that its Hessian becomes positive definite and invertible [250]; and (3) depending on the learning task and output predictions specific influence functions can be derived from first principles [251, 252]. Here, we adopt the aforementioned first strategy for non-linear models.

4.2.3. Proposed Approaches. We now present Algorithm 4 for influence estimation via trees to interpret how data samples and feature ranges impact model performance with respect to utility, fairness, and robustness. Additionally, we propose Algorithm 5 for trimming training samples to improve model performance given a budget.

4.2.3.1. *Estimating the Influence of Samples.* Influence functions can efficiently estimate the data impacts in various aspects. To further provide their interpretations, we employ decision trees to uncover which sample features contribute positively or negatively to the model's performance with respect to desired evaluation functions. Additionally, to address the issue of the tree depth on

interpretability, we further utilize hierarchical shrinkage [253] to regularize the tree. We present our approach for influence estimation in Algorithm 4, where it takes as input a regularization parameter λ, training set Z, validation set V, and a desired influence function definition $\mathcal{I}^{\mathcal{F}}$ for samples in Z on V. Specifically, we iterate over each training set tuple $(x_i, y_i) \in Z$ and create a new regression dataset M where the regressor is $[x_i, y_i]$ (block matrix notation implies appending y_i to the end of x_i) and the response variable is computed via $\mathcal{I}^{M}(-x_i)$. Note that we append the label to our dataset since influence estimation is dependent on class labels.

Algorithm 4 : Influence Estimation Via Trees

Input: Training set Z, Validation set V, Influence function $\mathcal{I}^{\mathcal{F}}$, Hyperparameter λ
Output: Influence Estimator Tree $\hat{h}$
1: **initialize** $M \leftarrow \emptyset$.
2: **for** $(x_i, y_i) \in Z$ **do**
3: $\quad q_i \leftarrow [x_i, y_i]$ //`appending label to` x_i
4: $\quad M \leftarrow M \cup \{(q_i, \mathcal{I}^{\mathcal{F}}(-x_i))\}$
5: **end for**
6: **train** h using CART [254] on M.
7: **return** $\hat{h}$ by using hierarchical shrinkage [253] on h with λ.

Then, we train a regression tree h using CART [254]. To ensure that the tree is interpretable while preserving performance, we utilize hierarchical shrinkage [253] post-training. For our tree h and for a given sample q_i in the dataset M, let its leaf-to-root node path in the tree denote as $t_w \subset t_{w-1} \subset ... \subset t_0$. Here t_w represents the leaf node and t_0 is the root node. Then we define two mapping functions for ease of readability: ϕ and ξ. The function ϕ takes as input a tree node and returns the number of samples it contains. The function ξ takes as input the query sample q and the tree node t and outputs the average predictive response for q at node t. The overall regression tree prediction model for q_i can then be written as: $h(q_i) = \xi(q_i, t_0) + \sum_{j=1}^{w} \xi(q_i, t_j) - \xi(q_i, t_{j-1})$. The hierarchical shrinkage regularizes the tree h by shrinking the prediction over each tree node by the sample means of its parent nodes, $\hat{h}(q_i) = \xi(q_i, t_0) + \sum_{j=1}^{w} \frac{\xi(q_i, t_j) - \xi(q_i, t_{j-1})}{1 + \lambda / \phi(t_{j-1})}$.

4.2.3.2. *Data Trimming for Supervised Models.* We present Algorithm 5 for trimming training datasets which takes as input training data sample-label tuples as Z, validation set V, an influence function definition $\mathcal{I}^{\mathcal{F}}$ for samples in Z on set V, and a budget b for the number of samples to remove. Our algorithm outputs the trimmed dataset Z'. The goal is to remove samples from the dataset that have negative influence. In Line 1 we initialize the sets J, K, and Z'. Then, in Lines 2-4 we

Algorithm 5 : Data Trimming

Input: Training set Z, Validation set V, Influence function $\mathcal{I}^{\mathcal{F}}$, Budget b
Output: Trimmed Dataset Z'

1: **initialize** $J \leftarrow \emptyset$, $K \leftarrow \emptyset$, $Z' \leftarrow \emptyset$.
2: **for** $(x_i, y_i) \in Z$ **do**
3: $\quad J \leftarrow J \cup \{\mathcal{I}^{\mathcal{F}}(-x_i)\}$ //on set V
4: $\quad K \leftarrow K \cup \{i\}$
5: **end for**
6: **sort** J in ascending order.
7: **sort** K using J.
8: $b' \leftarrow \sum \mathbb{1}_{j<0, j \in J}$, $K \leftarrow K_{:\min\{b,b'\}}$
9: **return** $Z' \leftarrow Z' \cup \{x_i\}, \forall i \notin K, x_i \in Z$.

populate J with the influence values of samples in J, and K with the indices of these samples. We then sort J in order of increasing positive influence, and K according to the sorted order obtained via J. In Line 8 we trim K by only selecting the first $\min\{b, b'\}$ indices where b' is the total number of negative influence samples. Finally, we select only those samples to be part of Z' that do not have indices in K and return Z'.

4.2.4. Results for Conventional Classification. In this section, we present results for our algorithms presented in the previous section. We first verify the correctness of our algorithms on synthetically generated toy data. We analyze how our influence estimation algorithm can be used to visualize and interpret regions of positive or negative influence, and trim the training set to improve accuracy, fairness, and robustness on the validation set. We then demonstrate the effectiveness of our algorithms on test sets of four real-world datasets and greatly improve their performance, especially for fairness and robustness.

4.2.4.1. *Correctness Verification on Toy Data.* We demonstrate the correctness of our proposed methods on toy data. We generate 150 train and 100 validation samples using two isotropic 2D-Gaussian distributions and apply a logistic regression model for the binary classification. Based on that, we analyze the training sample positive/negative impact on model's accuracy, fairness and robustness in Figure 4.7 A, D, and G, respectively. The green regions denote the positive influences, while the pink regions denote the negative influences. These regions indicate the feature values derived from training samples that affect the linear model either positively or negatively for the function of interest. In Figure 4.7 A, most samples contribute positively for utility, and hence lead

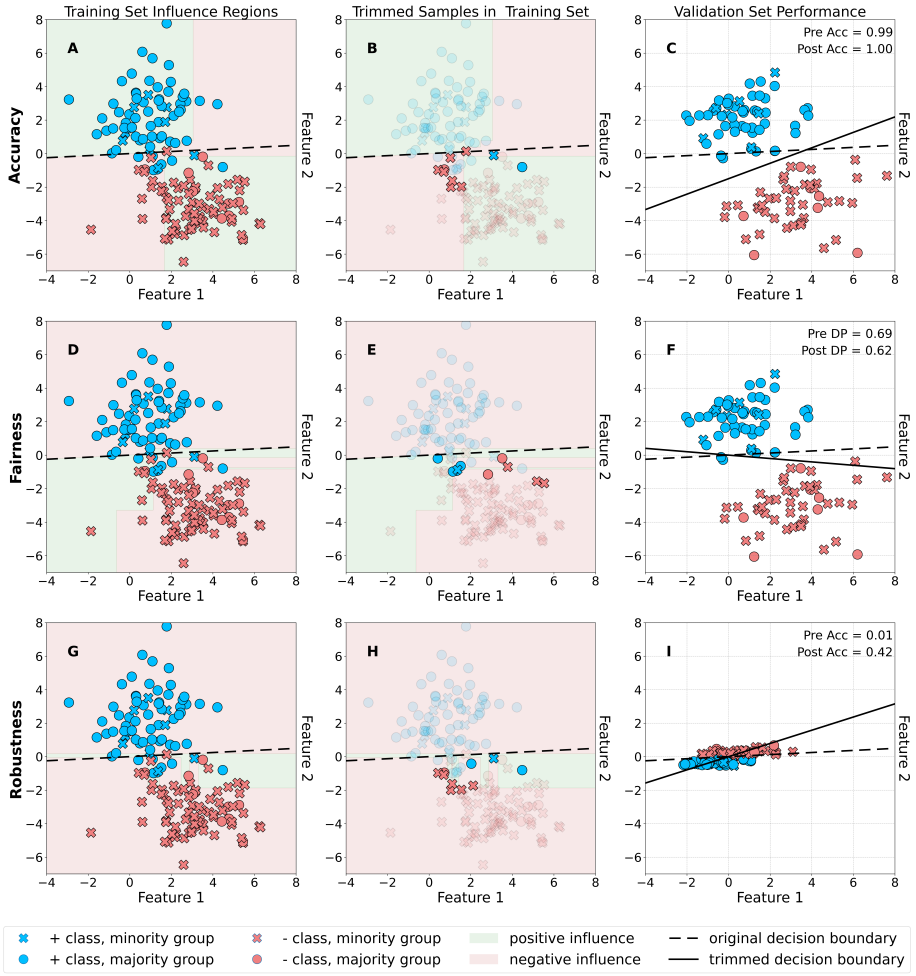


FIGURE 4.7. Correctness verification on toy data for utility (accuracy), fairness, and robustness functions. (+) class samples are denoted in blue, (-) class samples are denoted in orange, majority group samples are denoted by ∘ and minority group samples are denoted by ×. A, D, and G show the training set with the original logistic regression decision boundary, and we visualize the influence regions generated by our tree model from Algorithm 4 as positive (light green) or negative (light red) for each function: utility/fairness/robustness. B, E, and H denote the points to be trimmed as identified by Algorithm 5 for each function type. C and D denote the validation set and how we obtain improved accuracy and fairness performance on it after trimming. I denotes the adversarial validation set and how post trimming we can improve performance on adversarial data.

to positive influence regions. Similarly, in Figure 4.7 D, most of the samples are harmful for fairness as the (+) class has a large number of majority group samples and the (-) class has a large number of minority samples. Thus, most of the regions identified are of negative influence on fairness. To

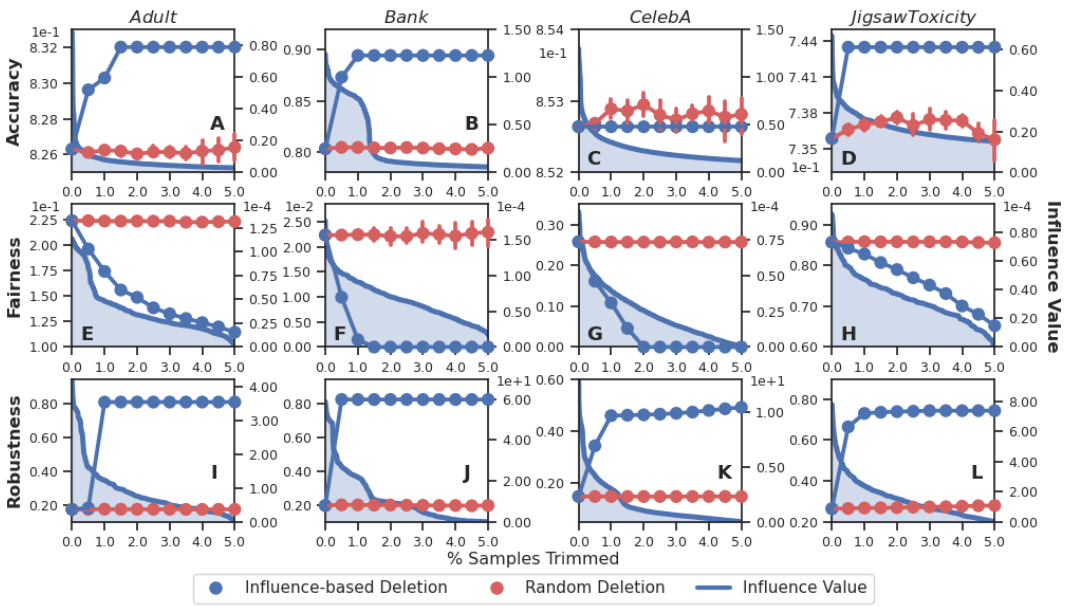


FIGURE 4.8. Double y-plot of our trimming method on the test sets of *Adult*, *Bank*, *CelebA*, and *Jigsaw Toxicity*. The left y-axes present algorithmic accuracy, fairness, and robustness, while the right y-axes present the influence value of the trimmed samples. Higher values indicate better accuracy and robustness, while lower values indicate better fairness. The blue lines represent the performance of our trimming method, while the red lines represent the performance with random trimming.

further validate the sample impact, we use Algorithm 5 with a budget $b = 10$ to obtain the points to be trimmed, shown in Figure 4.7 B, E, and H. We then remove these points from training set, resulting in significant performance improvement on the validation set for each function of interest, as shown in Figure 4.7 C, F, and I. Notably, the robustness is significantly improved from 0.01 to 0.42 by simply removing the identified points. Additionally, Figure 4.7 I shows an adversarial set with perturbed samples, where adversarial points crowd around the original decision boundary and are difficult to classify accurately.

4.2.4.2. *Algorithmic Performance on Real-World Datasets.* We demonstrate the performance of our proposed methods on multiple real-world datasets, consisting of two tabular datasets *Adult* [173] and *Bank* [174], one visual dataset *CelebA* [255], and one textual dataset *Jigsaw Toxicity* [256]. For *CelebA* we utilize extracted features provided by the dataset authors, and for *Jigsaw Toxicity* we extract text embeddings via the MiniLM transformer model [257]. We set sex (male/female) as the sensitive attribute for *Adult*, *Bank*, *CelebA*, and ethnicity (black/other) for *Jigsaw Toxicity*. Note

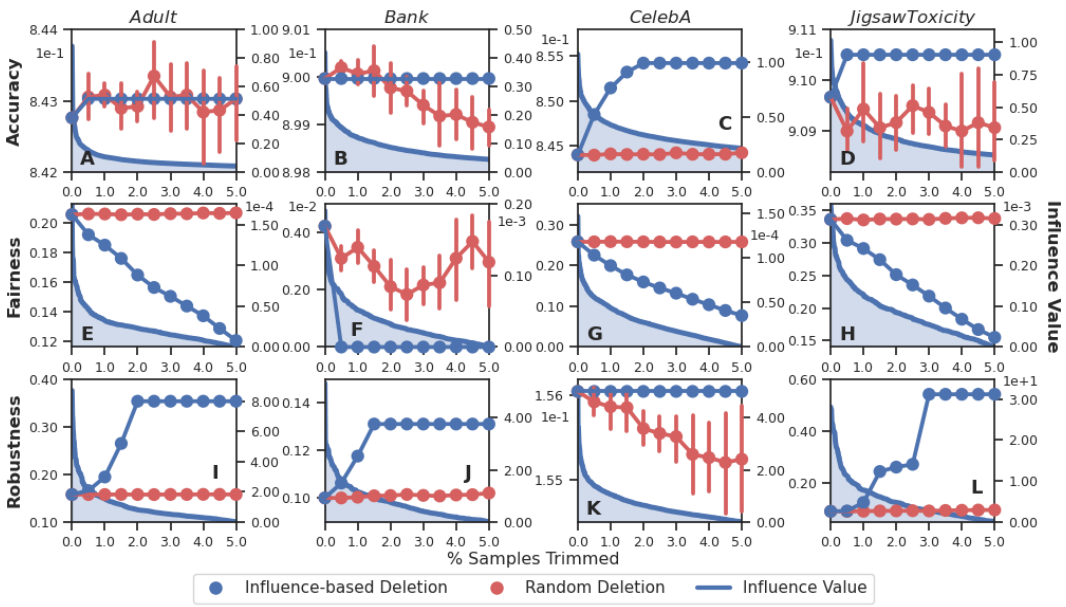


FIGURE 4.9. Double y-plot of our trimming method on the test sets of *Adult*, *Bank*, *CelebA*, and *Jigsaw Toxicity* for the MLP Neural Network as the base learning model. The left y-axes present algorithmic accuracy, fairness (DP), and robustness, while the right y-axes present the influence value of the trimmed samples. Higher values indicate better accuracy and robustness, while lower values indicate better fairness. The blue lines represent the performance of our trimming method, while the red lines represent the performance with random trimming.

that influence for these experiments is only measured on the validation set. Using logistic regression as the learning model, we present results on test sets of these datasets in Figure 4.8, along with influence value magnitudes corresponding to trimmed samples for each function type. Results with using MLP as the learning model are presented in Figure 4.9 and exhibit similar trends.

In Figure 4.8, we demonstrate the effect of increasing the budget b of our trimming approach (up to 5% of training data) for each function and compare it to a baseline that randomly removes b samples from the training set.[3] The results show that our methods are able to improve fairness (DP) and robustness significantly, while the random baseline fails to do so. This is particularly evident for datasets where large fairness disparities exist, except for *Bank*. Additionally, we are able to achieve significant improvements in accuracy on the fully adversarial test set for all datasets, indicating that our trimming can improve the model's robustness by effectively selecting training samples. However,

[3]We fail to compare with Shapley-value based data valuation methods [231] in utility due to their prohibitively long execution time on our datasets.

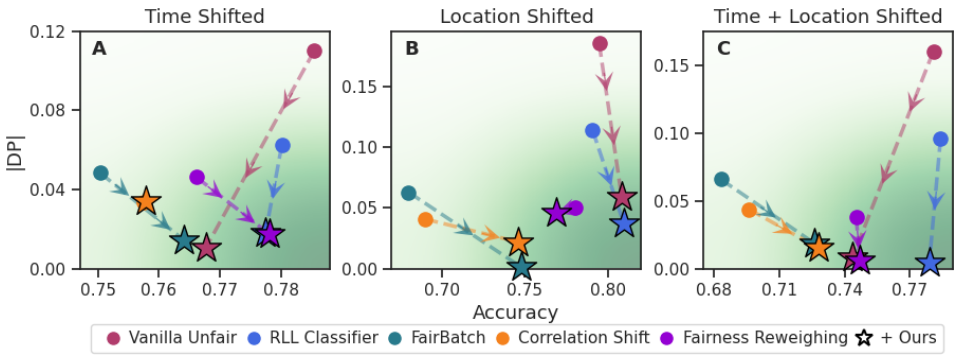


FIGURE 4.10. Performance of fairness and utility of several fairness algorithms under distribution shift. A greener background denotes a better solution in terms of both accuracy and fairness.

we observe that trimming does not lead to much improvement in accuracy for most datasets, except for *Bank*, where we obtain a 10% increase. We conjecture that the current logistic regression models may already achieve optimal performance on the other datasets, as indicated by the quick decrease in utility influence value, which suggests little room for further improvement. This also implies that we may not require as much training data to achieve similar utility results.

4.2.5. Application Scenarios Beyond Conventional Classification. We extend the conventional classification setting of the previous section by conducting extensive experiments that cover various practical use cases in deployment and maintenance scenarios, such as correcting fairness in distribution shift, combating fairness poisoning attacks, defending against adaptive adversaries undertaking evasion attacks, online learning with streaming batch data, and analyzing unlabeled sample effectiveness for active learning. We thus demonstrate the effectiveness of our influence-based estimation and trimming methods in improving model performance across diverse application scenarios either by trimming the training set, or by identifying new beneficial samples to add to it.

4.2.5.1. *Mitigating Unfairness Under Distribution Shift.* We demonstrate how our proposed method can serve as an effective fairness intervention and enhance the current methods under distribution shift. Fairness-specific distribution shifts [258] occur when the distribution of sensitive attributes (and class labels) between the training and test sets have drifted apart, resulting in severe issues of unfairness on the test set. Although it is a newly emerging direction, some pioneering attempts [258, 259, 260] have been taken to address the change in bias between the original and

test distribution. Typically, these methods have access to some samples from the test set to study the change in bias. In our experiments, these samples can constitute the validation set V given that the validation and test set are correlated with each other. Here we employ multiple fair intervention approaches as baselines– such as Correlation Shift [258], FairBatch [261], Robust Fair Logloss Classifier (RLL) [262], Influence-based Reweighing [238], and vanilla logistic regression. We use the *ACS Income* [263] dataset and construct three different distribution shift scenarios– time-based, location-based, time&location-based. In all three scenarios, we use the same training data constituting California (2014) data with 18000 samples, while the test sets consist of 6000 different samples: for time-based shift, this comprises California (2018) data, for location-based shift, it is Michigan (2014) data, and for the time&location-based shift, we use Michigan (2018) data. We set the budget $b = 600$ which constitutes only 3.33% of the training set.

We present results as fairness-accuracy plots in Figure 4.10. We show that our trimming based approach can greatly improve fairness under distribution shift even when utilized with the vanilla model, making it competitive with the fairness intervention baselines. Moreover, except for the vanilla model (Figure 4.10 A&C), our trimming approach with the fairness intervention methods improves their accuracy significantly as well. This demonstrates the generalizability of our approach, as it can be used as a pre-processing step to boost the performance of other mitigation strategies.

4.2.5.2. *Combating Poisoning Attacks Against Unfairness.* We use our influence-based trimming approach as an effective defense for mitigating poisoning attacks against fairness. Recently, several attacks [21, 107, 264] have been proposed to reduce fairness of learning models. Random Anchoring Attack (RAA) and Non-random Anchoring Attack (NRAA)[107] are two representative ones. These first select a set of target points to attack, and then poison a small subset of the training data by placing additional anchoring points near the targets with the opposite class label. The target samples are uniformly randomly chosen in RAA, while NRAA selects “popular" target samples that result in the maximization of bias for all samples in the test set post the attack. Here we follow the original implementations of RAA and NRAA and use their same datasets including *German* [265], *Compas* [266], and *Drug* [267]. Since all these are different sizes we set the trimming budget $b \leq 10\%$ of training samples and the attacker's poisoning budget is 10% of training samples.

TABLE 4.5. Performance of fairness poisoning attacks

Method	*German*		*Compas*		*Drug*	
	DP	ACC	DP	ACC	DP	ACC
Before	0.117	**0.670**	-0.281	0.652	0.371	**0.668**
RAA	-0.029	0.655	**-0.060**	0.630	0.225	0.657
RAA + Ours	**0.000**	0.650	-0.082	0.643	**0.090**	0.616
NRAA	0.268	0.665	0.168	0.652	0.501	0.665
NRAA + Ours	**0.005**	0.650	**0.048**	**0.654**	**0.142**	0.657

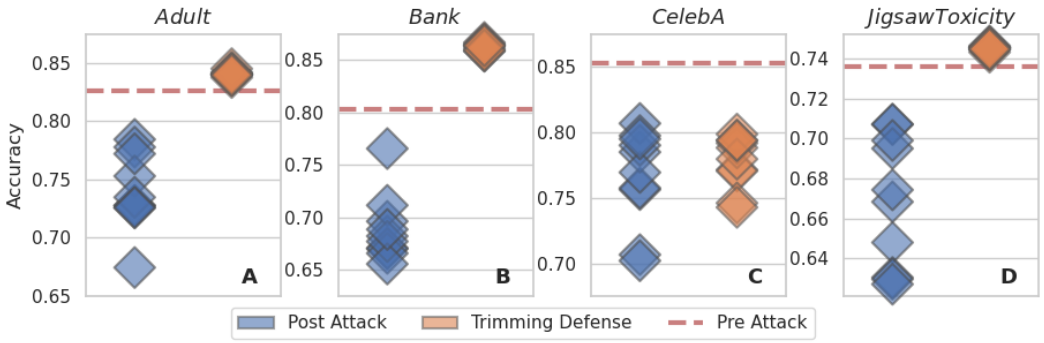


FIGURE 4.11. Defending with our trimming method against adaptive evasion attacks on four datasets.

Table 4.5 shows the results obtained. Our trimming approach is a useful approach for combating fairness-reducing poisoning attacks, as it improves fairness (and utility for *Compas*) performance compared to the metric values obtained before and after the attacks. It is also significant to note that there are currently no well-performing defenses proposed for the aforementioned attack in the supervised setting [21]. Our trimming approach thus shows promise as potentially the first defense against such attacks as it improves fairness significantly without sacrificing utility. Also note that since the defender does not know how many points the attacker poisons, the goal here is not to discover poisoned samples, but maximizing performance on the test set compared to both pre-attack and post-attack performance by trimming a small number of samples.

4.2.5.3. *Defending Against Adaptive Evasion Attacks.* In addition to fairness shifts and poisoning defense, we show how the influence-based trimming approach can help defend against an *adaptive adversary* [268, 269] that conducts evasion attacks on the given learning model. Here, adaptive refers to the adversary being able to target the given defense. In our case, the attacker can randomly

select test samples to conduct the attack. Notably, this is significantly different from our robustness analysis results, where we create an adversarial set using all test samples and then trim samples to optimize robustness accuracy. Here we defend by choosing to trim the training set by a fixed amount, but we have no knowledge of what samples are adversarial during inference.

We use the attack described previously to attack logistic regression. The attacker is adaptive to randomly choose between 5-25% of the test set samples to perturb. Correspondingly, we let the defender trim only 2.5% of the training set using Algorithm 5. We repeat these experiments between the attacker and defender over 10 runs for the 4 real-world datasets. In Figure 4.11, we plot the pre-attack accuracy, the post-attack accuracy distribution, as well as the post-attack accuracy distribution after trimming the training set and retraining the model, on the test set. These results indicate that trimming the training set after the attack can be a viable defense. With just 2.5% of the training data trimmed, we find that we can increase performance well beyond pre-attack values, and hence, mitigate the threat of the attack.

4.2.5.4. *Online Learning with Streaming Data and Noisy Labels.* So far, we considered classification problems with a fixed training set. In this part, we consider the online classification setting with streaming batch data. Online learning is a popular choice when learning models need to be implemented on memory constrained systems, or to combat distribution shift [25, 270]. Online learning assumes that the training data arrives in sequential batches and the learning model trains/updates on each batch sequentially. Here we reuse the 4 real-world datasets and trim samples in each batched stream of data using Algorithm 5. Note that we estimate influence over each batch independently. To make the setting more practical, we consider that the data stream might consist of noisy class labels– where one-third of samples that arrive in a batch have noisy labels flipped from the original ground truth labels. We set the trimming budget b to be 10% of the batch size, and train on 10 sequential batches of data. We then measure test set accuracy with and without trimming each batch for both the online models. Figure 4.12 A-D show the performance of logistic regression and linear Support Vector Machines (SVMs) [271] trained in an online manner with Stochastic Gradient Descent [272] with and without our trimming processing. It can be seen that trimming combined with our influence estimation model can lead to much better performance for all the given datasets (except *Bank*) and learning models.

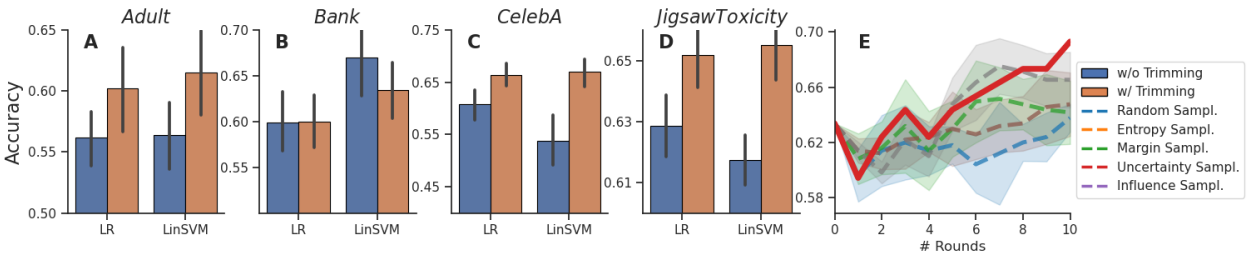


FIGURE 4.12. **A - D**: Online learning with noisy labels using logistic regression and linear Support Vector Machines with and without our influence-based trimming strategy on four datasets; **E**: Comparison of different active learning strategies on the *Diabetic Retinopathy* dataset.

4.2.5.5. *Active Learning.* Finally, we employ our approaches for active learning by choosing beneficial samples for annotation. Active learning[24] constitutes a small labeled training set, and a large pool of unlabeled samples, where the goal is to pick samples from the unlabeled set for annotation and then retrain the model on the combined data for boosting performance. It is generally employed in fields where annotation is very expensive to undertake, such as in medicine and health.

Here, we use a linear SVM as our learning model since a number of active learning strategies work optimally for SVM models [273]. We simulate an active learning use-case, following prior work [274, 275], by using the *Diabetic Retinopathy* [276] dataset. The dataset consists of 1151 retina images of patients and can be used for predicting whether they suffer from diabetic retinopathy or not. We use the image features as extracted in [277]. In this setting, there are 10 rounds of querying for annotations, where each round allows for 10 samples to be annotated. As a result, by the end of the final round we will have an additional 100 labeled samples available. We consider the following baselines for active learning– random sampling, entropy sampling [278], margin sampling [279], and uncertainty sampling [280]. For our influence-based sampling, since we do not have access to labels in the unlabeled set, we train our tree estimation model (Algorithm 4) without labels. Then we use the estimator to predict the influence of the unlabeled samples. Out of these we select the 10 highest influence samples available in each round for labeling. Note that unlike the other baselines, our influence-based sampling is deterministic over multiple runs since our tree estimator is deterministic when provided the same input. We present the results for this experiment in Figure 4.12 E over 5 runs. It can be seen that our influence-based sampling generally outperforms the other baselines in most rounds, and ends up with the best performance after the final round, demonstrating its effectiveness.

CHAPTER 5

Conclusion

In this thesis, we have formulated and investigated various learning problems centered around robust and fair ML/DL. Primarily, we study adversarial robustness (model *security*), social robustness (model *fairness*), and the interplay between these two dimensions of robustness for numerous learning models and application pipelines. Our work aims to pave the way for analyzing model robustness, and for developing models that can improve robustness along one or more dimensions– thus accelerating the integration of these models into society. For adversarial robustness, we propose novel poisoning and evasion attacks against deep and traditional unsupervised classification models, and also showcase how our attacks can be used in the real-world by attacking a production-level ML-as-a-Service face clustering API. For social robustness, we study alternate approaches to generalized fair clustering via data augmentation, and formulate the fair video summarization problem. We also provide new datasets and preliminary methods for the latter. Finally, we study two problems at the intersection of adversarial and social robustness– undertaking secure and fair data clustering, and utilizing influence-based data selection approaches for individually optimizing classifier fairness, accuracy, and robustness. Through these contributions, we seek to galvanize efforts exploring model robustness for novel and traditional problem settings alike.

Bibliography

[1] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587): 484–489, 2016.

[2] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

[3] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

[4] Md Zahangir Alom, Tarek M Taha, Christopher Yakopcic, Stefan Westberg, Paheding Sidike, Mst Shamima Nasrin, Brian C Van Esesn, Abdul A S Awwal, and Vijayan K Asari. The history began from alexnet: A comprehensive survey on deep learning approaches. *arXiv preprint arXiv:1803.01164*, 2018.

[5] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[6] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in Machine Learning: From Phenomena to Black-box Attacks using Adversarial Samples. *arXiv preprint arXiv:1605.07277*, 2016.

[7] Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, AISec '17, pages 3–14, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-5202-4. doi: 10.1145/3128572.3140444. URL `http://doi.acm.org/10.1145/3128572.3140444`.

[8] Nathan Drenkow, Numair Sani, Ilya Shpitser, and Mathias Unberath. A systematic review of robustness in deep learning for computer vision: Mind the gap? *arXiv preprint arXiv:2112.00639*, 2021.

[9] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A Survey on Bias and Fairness in Machine Learning. *ACM Computing Surveys*, 54(6):1–35, 2021.

[10] Leilani H Gilpin, David Bau, Ben Z Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*. IEEE, 2018.

[11] Anshuman Chhabra, Abhishek Roy, and Prasant Mohapatra. Suspicion-Free Adversarial Attacks on Clustering Algorithms. In *AAAI*, pages 3625–3632, 2020.

[12] Anshuman Chhabra, Ashwin Sekhari, and Prasant Mohapatra. On the Robustness of Deep Clustering Models: Adversarial Attacks and Defenses. *Advances in Neural Information Processing Systems*, 35:20566–20579, 2022.

[13] Bruno Lepri, Nuria Oliver, Emmanuel Letouzé, Alex Pentland, and Patrick Vinck. Fair, transparent, and accountable algorithmic decision-making processes: The premise, the proposed solutions, and the open challenges. *Philosophy & Technology*, 31:611–627, 2018.

[14] Anshuman Chhabra, Karina Masalkovaitė, and Prasant Mohapatra. An overview of fairness in clustering. *IEEE Access*, 9:130698–130720, 2021.

[15] Anshuman Chhabra, Adish Singla, and Prasant Mohapatra. Fair clustering using antidote data. In *Algorithmic Fairness through the Lens of Causality and Robustness workshop*, pages 19–39. PMLR, 2022.

[16] Anshuman Chhabra and Prasant Mohapatra. Fair algorithms for hierarchical agglomerative clustering. In *2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 206–211. IEEE, 2022.

[17] Anshuman Chhabra, Adish Singla, and Prasant Mohapatra. Fairness degrading adversarial attacks against clustering algorithms. *arXiv preprint arXiv:2110.12020*, 2021.

[18] Bashir Rastegarpanah, Krishna P. Gummadi, and Mark Crovella. Fighting Fire with Fire: Using Antidote Data to Improve Polarization and Fairness of Recommender Systems. In *WSDM*, pages 231–239, 2019.

[19] Evlampios Apostolidis, Eleni Adamantidou, Alexandros I Metsai, Vasileios Mezaris, and Ioannis Patras. Video summarization using deep neural networks: A survey. *Proceedings of the IEEE*, 109(11):1838–1863, 2021.

[20] Anshuman Chhabra, Chandana Kuntala, Kartik Patwari, Sristi, Deepak Kumar Sharma, and Prasant Mohapatra. FairVidSum: Benchmarking Video Summarization Models for Unfairness. *Under Submission*, 2023.

[21] Anshuman Chhabra, Peizhao Li, Prasant Mohapatra, and Hongfu Liu. Robust Fair Clustering: A Novel Fairness Attack and Defense Framework. In *International Conference on Learning Representations*, 2023.

[22] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*, 2017.

[23] Anshuman Chhabra, Peizhao Li, Prasant Mohapatra, and Hongfu Liu. "What Data Benefits My Classifier?" Enhancing Model Performance and Interpretability through Influence-Based Data Selection. *Under Submission*, 2023.

[24] David A Cohn, Zoubin Ghahramani, and Michael I Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 1996.

[25] Steven CH Hoi, Doyen Sahoo, Jing Lu, and Peilin Zhao. Online learning: A comprehensive survey. *Neurocomputing*, 2021.

[26] Antonio Emanuele Cinà, Alessandro Torcinovich, and Marcello Pelillo. A Black-box Adversarial Attack for Poisoning Clustering. *Pattern Recognition*, 122:108306, 2022.

[27] Peizhao Li, Ethan Xia, and Hongfu Liu. Learning antidote data to individual unfairness. In *International Conference on Machine Learning*, 2023.

[28] Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song. Using self-supervised learning can improve model robustness and uncertainty. *Advances in neural information processing systems*, 32, 2019.

[29] Dan Hendrycks, Kimin Lee, and Mantas Mazeika. Using pre-training can improve model robustness and uncertainty. In *International conference on machine learning*, pages 2712–2721. PMLR, 2019.

[30] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1944–1952, 2017.

[31] Dan Hendrycks, Mantas Mazeika, Duncan Wilson, and Kevin Gimpel. Using trusted data to train deep networks on labels corrupted by severe noise. *Advances in neural information processing systems*, 31, 2018.

[32] Adarsh Subbaswamy, Roy Adams, and Suchi Saria. Evaluating model robustness and stability to dataset shift. In *International conference on artificial intelligence and statistics*, pages 2611–2619. PMLR, 2021.

[33] Joel L Horowitz and Charles F Manski. Identification and robustness with contaminated and corrupted data. *Econometrica: Journal of the Econometric Society*, pages 281–302, 1995.

[34] Dong Yin, Raphael Gontijo Lopes, Jon Shlens, Ekin Dogus Cubuk, and Justin Gilmer. A fourier perspective on model robustness in computer vision. *Advances in Neural Information Processing Systems*, 32, 2019.

[35] Wei Emma Zhang, Quan Z Sheng, Ahoud Alhazmi, and Chenliang Li. Adversarial attacks on deep-learning models in natural language processing: A survey. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 11(3):1–41, 2020.

[36] Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *Ieee Access*, 6:14410–14430, 2018.

[37] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[38] Yansong Gao, Bao Gia Doan, Zhi Zhang, Siqi Ma, Jiliang Zhang, Anmin Fu, Surya Nepal, and Hyoungshick Kim. Backdoor attacks and countermeasures on deep learning: A comprehensive review. *arXiv preprint arXiv:2007.10760*, 2020.

[39] Han Xu, Xiaorui Liu, Yaxin Li, Anil Jain, and Jiliang Tang. To be robust or to be fair: Towards fairness in adversarial training. In *International conference on machine learning*, pages 11492–11501. PMLR, 2021.

[40] Haipei Sun, Kun Wu, Ting Wang, and Wendy Hui Wang. Towards fair and robust classification. In *2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P)*, pages 356–376. IEEE, 2022.

[41] Yuji Roh, Kangwook Lee, Steven Whang, and Changho Suh. Sample selection for fair and robust training. *Advances in Neural Information Processing Systems*, 34:815–827, 2021.

[42] Bo Yang, Xiao Fu, Nicholas D Sidiropoulos, and Mingyi Hong. Towards k-means-friendly Spaces: Simultaneous Deep Learning and Clustering. In *ICML*, pages 3861–3870. PMLR, 2017.

[43] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised Deep Embedding for Clustering Analysis. In *ICML*, pages 478–487. PMLR, 2016.

[44] Peihao Huang, Yan Huang, Wei Wang, and Liang Wang. Deep Embedding Network for Clustering. In *2014 22nd International Conference on Pattern Recognition*, pages 1532–1537. IEEE, 2014.

[45] Pan Ji, Tong Zhang, Hongdong Li, Mathieu Salzmann, and Ian Reid. Deep Subspace Clustering Networks. *Advances in Neural Information Processing Systems*, 30, 2017.

[46] Zhuxi Jiang, Yin Zheng, Huachun Tan, Bangsheng Tang, and Hanning Zhou. Variational Deep Embedding: An Unsupervised and Generative Approach to Clustering. *arXiv preprint arXiv:1611.05148*, 2016.

[47] Nat Dilokthanakul, Pedro AM Mediano, Marta Garnelo, Matthew CH Lee, Hugh Salimbeni, Kai Arulkumaran, and Murray Shanahan. Deep Unsupervised Clustering with Gaussian Mixture Variational Autoencoders. *arXiv preprint arXiv:1611.02648*, 2016.

[48] Warith Harchaoui, Pierre-Alexandre Mattei, and Charles Bouveyron. Deep Adversarial Gaussian Mixture Auto-Encoder for Clustering. Technical report, 2017.

[49] Jost Tobias Springenberg. Unsupervised and Semi-Supervised Learning with Categorical Generative Adversarial Networks. *arXiv preprint arXiv:1511.06390*, 2015.

[50] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets. *Advances in Neural Information Processing Systems*, 29, 2016.

[51] Chuang Niu, Hongming Shan, and Ge Wang. SPICE: Semantic Pseudo-labeling for Image Clustering. *arXiv preprint arXiv:2103.09382*, 2021.

[52] Yunfan Li, Peng Hu, Zitao Liu, Dezhong Peng, Joey Tianyi Zhou, and Xi Peng. Contrastive Clustering. In *AAAI*, 2021.

[53] Sungwon Park, Sungwon Han, Sundong Kim, Danu Kim, Sungkyu Park, Seunghoon Hong, and Meeyoung Cha. Improving Unsupervised Image Clustering with Robust Learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12278–12287, 2021.

[54] Ke Zhang, Wei-Lun Chao, Fei Sha, and Kristen Grauman. Video summarization with long short-term memory. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part VII 14*, pages 766–782. Springer, 2016.

[55] Yujia Zhang, Michael Kampffmeyer, Xiaoguang Zhao, and Min Tan. Dtr-gan: Dilated temporal relational adversarial network for video summarization. In *Proceedings of the ACM Turing Celebration Conference-China*, pages 1–6, 2019.

[56] Cheng Huang and Hongmei Wang. A novel key-frames selection framework for comprehensive video summarization. *IEEE Transactions on Circuits and Systems for Video Technology*, 30 (2):577–589, 2019.

[57] Tsu-Jui Fu, Shao-Heng Tai, and Hwann-Tzong Chen. Attentive and adversarial learning for video summarization. In *2019 IEEE Winter Conference on applications of computer vision (WACV)*, pages 1579–1587. IEEE, 2019.

[58] Luis Lebron Casas and Eugenia Koblents. Video summarization with lstm and deep attention models. In *MultiMedia Modeling: 25th International Conference, MMM 2019, Thessaloniki, Greece, January 8–11, 2019, Proceedings, Part II 25*, pages 67–79. Springer, 2019.

[59] Behrooz Mahasseni, Michael Lam, and Sinisa Todorovic. Unsupervised video summarization with adversarial lstm networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 202–211, 2017.

[60] Li Yuan, Francis EH Tay, Ping Li, Li Zhou, and Jiashi Feng. Cycle-sum: Cycle-consistent adversarial lstm networks for unsupervised video summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9143–9150, 2019.

[61] Xufeng He, Yang Hua, Tao Song, Zongpu Zhang, Zhengui Xue, Ruhui Ma, Neil Robertson, and Haibing Guan. Unsupervised video summarization with attentive conditional generative adversarial networks. In *Proceedings of the 27th ACM International Conference on multimedia*, pages 2296–2304, 2019.

[62] Gokhan Yaliniz and Nazli Ikizler-Cinbis. Using independently recurrent networks for reinforcement learning based unsupervised video summarization. *Multimedia Tools and Applications*, 80:17827–17847, 2021.

[63] Evlampios Apostolidis, Alexandros I Metsai, Eleni Adamantidou, Vasileios Mezaris, and Ioannis Patras. A stepwise, label-based approach for improving the adversarial training in unsupervised video summarization. In *Proceedings of the 1st International Workshop on AI for Smart TV Content Production, Access and Delivery*, pages 17–25, 2019.

[64] Kaiyang Zhou, Yu Qiao, and Tao Xiang. Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[65] Mayu Otani, Yuta Nakashima, Esa Rahtu, Janne Heikkilä, and Naokazu Yokoya. Video summarization using deep semantic features. In *Computer Vision–ACCV 2016: 13th Asian Conference on Computer Vision, Taipei, Taiwan, November 20-24, 2016, Revised Selected Papers, Part V 13*, pages 361–377. Springer, 2017.

[66] Huawei Wei, Bingbing Ni, Yichao Yan, Huanyu Yu, Xiaokang Yang, and Chen Yao. Video summarization via semantic attended networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[67] Jie Lei, Qiao Luan, Xinhui Song, Xiao Liu, Dapeng Tao, and Mingli Song. Action parsing-driven video summarization based on reinforcement learning. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(7):2126–2137, 2018.

[68] Kaiyang Zhou, Tao Xiang, and Andrea Cavallaro. Video summarisation by classification with deep reinforcement learning. *arXiv preprint arXiv:1807.03089*, 2018.

[69] Yitian Yuan, Tao Mei, Peng Cui, and Wenwu Zhu. Video summarization by learning deep side semantic embedding. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(1): 226–237, 2017.

[70] Xinhui Song, Ke Chen, Jie Lei, Li Sun, Zhiyuan Wang, Lei Xie, and Mingli Song. Category driven deep recurrent neural network for video summarization. In *2016 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, pages 1–6. IEEE, 2016.

[71] Wencheng Zhu, Jiwen Lu, Jiahao Li, and Jie Zhou. Dsnet: A flexible detect-to-summarize network for video summarization. *IEEE Transactions on Image Processing*, 30:948–962, 2020.

[72] Evlampios Apostolidis, Georgios Balaouras, Vasileios Mezaris, and Ioannis Patras. Combining global and local attention with positional encoding for video summarization. In *2021 IEEE International Symposium on Multimedia (ISM)*, pages 226–234. IEEE, 2021.

[73] Evlampios Apostolidis, Georgios Balaouras, Vasileios Mezaris, and Ioannis Patras. Summarizing videos using concentrated attention and considering the uniqueness and diversity of the video frames. In *Proceedings of the 2022 International Conference on Multimedia Retrieval*, pages 407–415, 2022.

[74] Evlampios Apostolidis, Eleni Adamantidou, Alexandros I Metsai, Vasileios Mezaris, and Ioannis Patras. Ac-sum-gan: Connecting actor-critic and generative adversarial networks for unsupervised video summarization. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(8):3278–3292, 2020.

[75] Evlampios Apostolidis, Eleni Adamantidou, Alexandros I Metsai, Vasileios Mezaris, and Ioannis Patras. Unsupervised video summarization via attention-driven adversarial learning. In *MultiMedia Modeling: 26th International Conference, MMM 2020, Daejeon, South Korea, January 5–8, 2020, Proceedings, Part I 26*, pages 492–504. Springer, 2020.

[76] Yale Song, Jordi Vallmitjana, Amanda Stent, and Alejandro Jaimes. Tvsum: Summarizing web videos using titles. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5179–5187, 2015.

[77] Michael Gygli, Helmut Grabner, Hayko Riemenschneider, and Luc Van Gool. Creating summaries from user videos. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VII 13*, pages 505–520. Springer, 2014.

[78] Stuart Lloyd. Least Squares Quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

[79] Battista Biggio, Ignazio Pillai, Samuel Rota Bulò, Davide Ariu, Marcello Pelillo, and Fabio Roli. Is Data Clustering in Adversarial Settings Secure? In *Proceedings of the 2013 ACM workshop on Artificial Intelligence and Security*, pages 87–98, 2013.

[80] Battista Biggio, Samuel Rota Bulò, Ignazio Pillai, Michele Mura, Eyasu Zemene Mequanint, Marcello Pelillo, and Fabio Roli. Poisoning Complete-linkage Hierarchical Clustering. In *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*, pages 42–52. Springer, 2014.

[81] Jonathan Crussell and Philip Kegelmeyer. Attacking DBSCAN for Fun and Profit. In *Proceedings of the 2015 SIAM International Conference on Data Mining*, pages 235–243. SIAM, 2015.

[82] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *KDD*, volume 96, pages 226–231, 1996.

[83] Alekh Agarwal, Alina Beygelzimer, Miroslav Dudík, John Langford, and Hanna Wallach. A reductions approach to fair classification. In *International Conference on Machine Learning*, pages 60–69. PMLR, 2018.

[84] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rogriguez, and Krishna P Gummadi. Fairness constraints: Mechanisms for fair classification. In *Artificial intelligence and statistics*, pages 962–970. PMLR, 2017.

[85] Flavio Chierichetti, Ravi Kumar, Silvio Lattanzi, and Sergei Vassilvitskii. Fair Clustering Through Fairlets. In *NeurIPS*, pages 5029–5037, 2017.

[86] Evaggelia Pitoura, Kostas Stefanidis, and Georgia Koutrika. Fairness in rankings and recommendations: an overview. *The VLDB Journal*, pages 1–28, 2022.

[87] Hadis Anahideh, Abolfazl Asudeh, and Saravanan Thirumuruganathan. Fair active learning. *Expert Systems with Applications*, 199:116981, 2022.

[88] Jie Shen, Nan Cui, and Jing Wang. Metric-fair active learning. In *International Conference on Machine Learning*, pages 19809–19826. PMLR, 2022.

[89] Hanyu Song, Peizhao Li, and Hongfu Liu. Deep clustering based fair outlier detection. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*,

pages 1481–1489, 2021.

[90] Ian Davidson and Selvan Suntiha Ravi. A framework for determining the fairness of outlier detection. In *ECAI 2020*, pages 2465–2472. IOS Press, 2020.

[91] Matthäus Kleindessner, Pranjal Awasthi, and Jamie Morgenstern. Fair k-center clustering for data summarization. In *Proceedings of the International Conference on Machine Learning*, 2019.

[92] Ashish Chiplunkar, Sagar Kale, and Sivaramakrishnan Natarajan Ramamoorthy. How to solve fair k-center in massive data models. In *International Conference on Machine Learning*, pages 1877–1886. PMLR, 2020.

[93] Haris Angelidakis, Adam Kurpisz, Leon Sering, and Rico Zenklusen. Fair and fast k-center clustering for data summarization. In *International Conference on Machine Learning*, pages 669–702. PMLR, 2022.

[94] Anurag Shandilya, Kripabandhu Ghosh, and Saptarshi Ghosh. Fairness of extractive text summarization. In *Companion Proceedings of the The Web Conference 2018*, pages 97–98, 2018.

[95] Vijay Keswani and L Elisa Celis. Dialect diversity in text summarization on twitter. In *Proceedings of the Web Conference 2021*, pages 3802–3814, 2021.

[96] Teofilo F Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical computer science*, 38:293–306, 1985.

[97] Simon Caton and Christian Haas. Fairness in Machine Learning: A Survey. *CoRR*, abs/2010.04053, 2020. URL `https://arxiv.org/abs/2010.04053`.

[98] Suman Kalyan Bera, Deeparnab Chakrabarty, Nicolas Flores, and Maryam Negahbani. Fair Algorithms for Clustering. In *NeurIPS*, pages 4955–4966, 2019.

[99] Ioana Oriana Bercea, Martin Groß, Samir Khuller, Aounon Kumar, Clemens Rösner, Daniel R. Schmidt, and Melanie Schmidt. On the Cost of Essentially Fair Clusterings. In *APPROX/RANDOM*, volume 145 of *LIPIcs*, pages 18:1–18:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[100] Arturs Backurs, Piotr Indyk, Krzysztof Onak, Baruch Schieber, Ali Vakilian, and Tal Wagner. Scalable Fair Clustering. In *ICML*, volume 97, pages 405–413, 2019.

[101] Melanie Schmidt, Chris Schwiegelshohn, and Christian Sohler. Fair Coresets and Streaming Algorithms for Fair k-means. In *Approximation and Online Algorithms - 17th International Workshop, WAOA*, volume 11926 of *Lecture Notes in Computer Science*, pages 232–251, 2019.

[102] Imtiaz Masud Ziko, Eric Granger, Jing Yuan, and Ismail Ben Ayed. Clustering with Fairness Constraints: A Flexible and Scalable Approach. *CoRR*, abs/1906.08207, 2019.

[103] Matthäus Kleindessner, Samira Samadi, Pranjal Awasthi, and Jamie Morgenstern. Guarantees for Spectral Clustering with Fairness Constraints. In *ICML*, volume 97, pages 3458–3467, 2019.

[104] Mehrdad Ghadiri, Samira Samadi, and Santosh S. Vempala. Socially Fair k-means Clustering. In *FAccT '21: 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 438–448, 2021.

[105] Ian Davidson and S. S. Ravi. Making Existing Clusterings Fairer: Algorithms, Complexity Results and Insights. In *AAAI*, pages 3733–3740, 2020.

[106] David Solans, Battista Biggio, and Carlos Castillo. Poisoning attacks on algorithmic fairness. In *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2020.

[107] Ninareh Mehrabi, Muhammad Naveed, Fred Morstatter, and Aram Galstyan. Exacerbating algorithmic bias through fairness attacks. In *AAAI Conference on Artificial Intelligence*, 2021.

[108] L Elisa Celis, Lingxiao Huang, Vijay Keswani, and Nisheeth K Vishnoi. Fair classification with noisy protected attributes: A framework with provable guarantees. In *Proceedings of International Conference on Machine Learning*, 2021.

[109] L Elisa Celis, Anay Mehrotra, and Nisheeth Vishnoi. Fair classification with adversarial perturbations. *Advances in Neural Information Processing Systems*, 2021.

[110] Joe H Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244, 1963.

[111] Anshuman Chhabra, Vidushi Vashishth, and Prasant Mohapatra. Fair algorithms for hierarchical agglomerative clustering. *arXiv preprint arXiv:2005.03197*, 2020.

[112] Lior Rokach and Oded Maimon. Clustering Methods. In *Data Mining and Knowledge Discovery Handbook*, pages 321–352. Springer, 2005.

[113] Andrew Ng, Michael Jordan, and Yair Weiss. On Spectral Clustering: Analysis and an Algorithm. *Advances in Neural Information Processing Systems*, 14, 2001.

[114] Erxue Min, Xifeng Guo, Qiang Liu, Gen Zhang, Jianjing Cui, and Jun Long. A Survey of Clustering with Deep Learning: From the Perspective of Network Architecture. *IEEE Access*, 6:39501–39514, 2018.

[115] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. *Advances in Neural Information Processing Systems*, 27, 2014.

[116] Eric P Xing, Michael I Jordan, Stuart J Russell, and Andrew Y Ng. Distance metric learning with application to clustering with side-information. In *Advances in neural information processing systems*, pages 521–528, 2003.

[117] Jörg Sander, Martin Ester, Hans-Peter Kriegel, and Xiaowei Xu. Density-based clustering in spatial databases: The algorithm gdbscan and its applications. *Data mining and knowledge discovery*, 2(2):169–194, 1998.

[118] Jonathan Crussell, Clint Gibler, and Hao Chen. Andarwin: Scalable detection of semantically similar android applications. In *European Symposium on Research in Computer Security*, pages 182–199. Springer, 2013.

[119] David E Goldberg. *Genetic algorithms*. Pearson Education India, 2006.

[120] Scott Kirkpatrick, C Daniel Gelatt, and Mario P Vecchi. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.

[121] Paul Knysh and Yannis Korkolis. Blackbox: A procedure for parallel optimization of expensive black-box functions. *arXiv preprint arXiv:1605.00998*, 2016.

[122] Rommel G. Regis and Christine A. Shoemaker. Constrained global optimization of expensive black box functions using radial basis functions. *Journal of Global Optimization*, 31(1):153–171, Jan 2005. ISSN 1573-2916. doi: 10.1007/s10898-004-0570-0. URL `https://doi.org/10.1007/s10898-004-0570-0`.

[123] Michael D McKay, Richard J Beckman, and William J Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 42(1):55–61, 2000.

[124] Kenneth Holmström, Nils-Hassan Quttineh, and Marcus M Edvall. An adaptive radial basis algorithm (arbf) for expensive black-box mixed-integer constrained global optimization. *Optimization and Engineering*, 9(4):311–339, 2008.

[125] Davy Paindaveine and Germain Van Bever. From depth to local depth: a focus on centrality. *Journal of the American Statistical Association*, 108(503):1105–1119, 2013.

[126] E Alpaydin and C Kaynak. Optical Recognition of Handwritten Digits. *http://archive.ics.uci.edu/ml/datasets/ Optical+Recognition+of+Handwritten+Digits*, 1995. [Online; accessed 1-November-2018].

[127] Yann LeCun, Corinna Cortes, and CJ Burges. MNIST Handwritten Digit Database. *http://yann.lecun.com/exdb/mnist*, 2, 2010.

[128] Małgorzata Charytanowicz, Jerzy Niewczas, Piotr Kulczycki, Piotr A Kowalski, Szymon Łukasik, and Sławomir Żak. Complete gradient clustering algorithm for features analysis of x-ray images. In *Information technologies in biomedicine*, pages 15–24. Springer, 2010.

[129] Andrew Gardner, Jinko Kanno, Christian A Duncan, and Rastko Selmic. Measuring distance between unordered sets of different sizes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 137–143, 2014.

[130] Alexander Strehl and Joydeep Ghosh. Cluster Ensembles—a Knowledge Reuse Framework for Combining Multiple Partitions. *Journal of Machine Learning Research*, 3:583–617, 2002.

[131] Lawrence Hubert and Phipps Arabie. Comparing Partitions. *Journal of Classification*, 2(1): 193–218, 1985.

[132] Tao Li and Chris Ding. The Relationships Among Various Nonnegative Matrix Factorization Methods for Clustering. In *Proceedings of the International Conference on Data Mining*, 2006.

[133] Harold W Kuhn. The Hungarian Method for the Assignment Problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955.

[134] Claude Elwood Shannon. A Mathematical Theory of Communication. *The Bell System Technical Journal*, 27(3):379–423, 1948.

[135] William M Rand. Objective Criteria for the Evaluation of Clustering Methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

[136] Yinpeng Dong, Qi-An Fu, Xiao Yang, Tianyu Pang, Hang Su, Zihao Xiao, and Jun Zhu. Benchmarking adversarial robustness on image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 321–331, 2020.

[137] Huichen Li, Linyi Li, Xiaojun Xu, Xiaolu Zhang, Shuang Yang, and Bo Li. Nonlinear Projection based Gradient Estimation for Query Efficient Blackbox Attacks. In *AISTATS*, pages 3142–3150. PMLR, 2021.

[138] Arjun Nitin Bhagoji, Warren He, Bo Li, and Dawn Song. Practical Black-box Attacks on Deep Neural Networks using Efficient Query Mechanisms. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 154–169, 2018.

[139] Seong Joon Oh, Bernt Schiele, and Mario Fritz. Towards Reverse-engineering Black-box Neural Networks. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, pages 121–144. Springer, 2019.

[140] Huiying Li, Shawn Shan, Emily Wenger, Jiayun Zhang, Haitao Zheng, and Ben Y Zhao. Blacklight: Defending Black-box Adversarial Attacks on Deep Neural Networks. *arXiv preprint arXiv:2006.14042*, 2020.

[141] Alireza Abedin, Farbod Motlagh, Qinfeng Shi, Hamid Rezatofighi, and Damith Ranasinghe. Towards deep clustering of human activities from wearables. In *Proceedings of the 2020 International Symposium on Wearable Computers*, pages 1–6, 2020.

[142] Md Zia Uddin and Ahmet Soylu. Human activity recognition using wearable sensors, discriminant analysis, and long short-term memory-based neural structured learning. *Scientific Reports*, 11(1):1–15, 2021.

[143] Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Song. Generating Adversarial Examples with Adversarial Networks. *arXiv preprint arXiv:1801.02610*, 2018.

[144] Surgan Jandial, Puneet Mangla, Sakshi Varshney, and Vineeth Balasubramanian. AdvGAN++: Harnessing Latent Layers for Adversary Generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[145] Jing Guo, Wei Ji, and Yun Li. Generative Networks for Adversarial Examples with Weighted Perturbations. In *2019 IEEE 14th International Conference on Intelligent Systems and Knowledge Engineering (ISKE)*, pages 778–784. IEEE, 2019.

[146] Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. SCAN: Learning to Classify Images Without Labels. In *European Conference on Computer Vision (ECCV)*, pages 268–285. Springer, 2020.

[147] Tsung Wei Tsai, Chongxuan Li, and Jun Zhu. MiCE: Mixture of Contrastive Experts for Unsupervised Image Clustering. In *ICLR*, 2020.

[148] Zhiyuan Dang, Cheng Deng, Xu Yang, Kun Wei, and Heng Huang. Nearest Neighbor Matching for Deep Clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13693–13702, 2021.

[149] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[150] Vikash Sehwag, Mung Chiang, and Prateek Mittal. SSD: A Unified Framework for Self-Supervised Outlier Detection. In *ICLR*, 2021.

[151] Benchmark: State-of-the-art Anomaly Detection Models on CIFAR-10. `https://paperswithcode.com/sota/anomaly-detection-on-anomaly-detection-on`, 2021. [Online; accessed 11-May-2021].

[152] Svante Wold, Kim Esbensen, and Paul Geladi. Principal Component Analysis. *Chemometrics and Intelligent Laboratory Systems*, 2(1-3):37–52, 1987.

[153] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. The CIFAR-10 Dataset. *http://www.cs.toronto.edu/kriz/cifar.html*, 2014.

[154] Alex Krizhevsky. Learning Multiple Layers of Features from Tiny Images. Technical report, 2009.

[155] Adam Coates, Andrew Ng, and Honglak Lee. An Analysis of Single-layer Networks in Unsupervised Feature Learning. In *AISTATS*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011.

[156] Kuang-Chih Lee, J. Ho, and D.J. Kriegman. Acquiring Linear Subspaces for Face Recognition Under Variable Lighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(5):684–698, 2005. doi: 10.1109/TPAMI.2005.92.

[157] Bing Nan Li, Chee Kong Chui, Stephen Chang, and Sim Heng Ong. Integrating Spatial Fuzzy Clustering with Level Set Methods for Automated Medical Image Segmentation. *Computers*

in biology and medicine, 41(1):1–10, 2011.

[158] Le Lu and René Vidal. Combined Central and Subspace Clustering for Computer Vision Applications. In *ICML*, volume 148, pages 593–600, 2006.

[159] Kevin Dela Rosa, Rushin Shah, Bo Lin, Anatole Gershman, and Robert Frederking. Topical Clustering of Tweets. *Proceedings of the ACM SIGIR: SWSM*, 63, 2011.

[160] Vijay Hanagandi, Amitava Dhar, and Kevin Buescher. Density-based Clustering and Radial Basis Function Modeling to Generate Credit Card Fraud Scores. In *Proceedings of the IEEE/IAFE Conference on Computational Intelligence for Financial Engineering*, pages 247–251, 1996.

[161] Xingyu Chen, Brandon Fain, Liang Lyu, and Kamesh Munagala. Proportionally Fair Clustering. In *ICML*, volume 97, pages 1032–1041, 2019.

[162] Stephan Dempe. *Foundations of Bilevel Programming*. Springer Science & Business Media, 2002.

[163] Ankur Sinha, Pekka Malo, and Kalyanmoy Deb. A Review on Bilevel Optimization: From Classical to Evolutionary Approaches and Applications. *IEEE Trans. Evol. Comput.*, 22(2): 276–295, 2018.

[164] Mohammad Javad Abdi. *Cardinality Optimization Problems*. PhD thesis, University of Birmingham, 2013.

[165] Fredrik Lindsten, Henrik Ohlsson, and Lennart Ljung. *Just Relax and Come Clustering!: A Convexification of k-means Clustering*. Linköping University Electronic Press, 2011.

[166] Toby Hocking, Jean-Philippe Vert, Francis R. Bach, and Armand Joulin. Clusterpath: an Algorithm for Clustering using Convex Fusion Penalties. In *ICML*, pages 745–752, 2011.

[167] André A. Keller. Chapter 3 - elements of technical background. In *Mathematical Optimization Terminology*, pages 239–298. 2018.

[168] Steven Diamond and Stephen P. Boyd. CVXPY: A Python-Embedded Modeling Language for Convex Optimization. *JMLR*, 17:83:1–83:5, 2016.

[169] Yang Yu, Hong Qian, and Yi-Qi Hu. Derivative-Free Optimization via Classification. In *AAAI*, pages 2286–2292, 2016.

[170] Nikolaus Hansen, Sibylle D. Müller, and Petros Koumoutsakos. Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES). *Evol. Comput.*, 11(1):1–18, 2003.

[171] Kenji Kawaguchi, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Bayesian Optimization with Exponential Convergence. In *NeurIPS*, pages 2809–2817, 2015.

[172] Hong Qian, Yi-Qi Hu, and Yang Yu. Derivative-Free Optimization of High-Dimensional Non-Convex Functions by Sequential Random Embeddings. In *IJCAI*, pages 1946–1952, 2016.

[173] Ron Kohavi. Scaling Up the Accuracy of Naive-Bayes Classifiers: A Decision-Tree Hybrid. In *KDD*, pages 202–207, 1996.

[174] Sérgio Moro, Paulo Cortez, and Paulo Rita. A Data-driven Approach to Predict the Success of Bank Telemarketing. *Decis. Support Syst.*, 62:22–31, 2014.

[175] I-Cheng Yeh and Che-hui Lien. The Comparisons of Data Mining Techniques for the Predictive Accuracy of Probability of Default of Credit Card Clients. *Expert Syst. Appl.*, 36(2):2473–2480, 2009.

[176] Gary B Huang, Marwan Mattar, Tamara Berg, and Eric Learned-Miller. Labeled Faces in the Wild: A Database For Studying Face Recognition in Unconstrained Environments, 2008.

[177] Peter J. Rousseeuw. Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987. ISSN 0377-0427.

[178] T. Caliński and J Harabasz. A Dendrite Method for Cluster Analysis. *Communications in Statistics*, 3(1):1–27, 1974.

[179] David L. Davies and Donald W. Bouldin. A Cluster Separation Measure. *IEEE Trans. Pattern Anal. Mach. Intell.*, 1(2):224–227, 1979.

[180] A Senthil Murugan, K Suganya Devi, A Sivaranjani, and P Srinivasan. A study on various methods used for video summarization and moving object detection for video surveillance applications. *Multimedia Tools and Applications*, 77(18):23273–23290, 2018.

[181] Sinnu Susan Thomas, Sumana Gupta, and Venkatesh K Subramanian. Smart surveillance based on video summarization. In *2017 IEEE region 10 symposium (TENSYMP)*, pages 1–5. IEEE, 2017.

[182] Yihong Gong and Xin Liu. Video summarization and retrieval using singular value decomposition. *Multimedia Systems*, 9(2):157–168, 2003.

[183] Yuxin Peng and Chong-Wah Ngo. Clip-based similarity measure for query-dependent clip retrieval and video summarization. *IEEE Transactions on Circuits and Systems for Video Technology*, 16(5):612–627, 2006.

[184] Gina Neff. Talking to bots: Symbiotic agency and the case of tay. *International Journal of Communication*, 2016.

[185] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. Machine bias. In *Ethics of data and analytics*, pages 254–264. Auerbach Publications, 2016.

[186] Richard Berk, Hoda Heidari, Shahin Jabbari, Matthew Joseph, Michael Kearns, Jamie Morgenstern, Seth Neel, and Aaron Roth. A convex framework for fair regression. *arXiv preprint arXiv:1706.02409*, 2017.

[187] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Innovations in Theoretical Computer Science Conference*, 2012.

[188] Alexander Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, 1998.

[189] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[190] Silvano Martello, David Pisinger, and Paolo Toth. Dynamic programming and strong bounds for the 0-1 knapsack problem. *Management science*, 45(3):414–424, 1999.

[191] Jon Kleinberg, Jens Ludwig, Sendhil Mullainathan, and Cass R Sunstein. Discrimination in the age of algorithms. *Journal of Legal Analysis*, 10:113–174, 2018.

[192] Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*, pages 191–198, 2016.

[193] Lee J Cronbach. Coefficient alpha and the internal structure of tests. *psychometrika*, 16(3): 297–334, 1951.

[194] Jiri Fajtl, Hajar Sadeghi Sokeh, Vasileios Argyriou, Dorothy Monekosso, and Paolo Remagnino. Summarizing videos with attention. In *Computer Vision–ACCV 2018 Workshops: 14th Asian*

Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers 14, pages 39–54. Springer, 2019.

[195] Hussain Kanafani, Junaid Ahmed Ghauri, Sherzod Hakimov, and Ralph Ewerth. Unsupervised video summarization via multi-source features. In *Proceedings of the 2021 International Conference on Multimedia Retrieval*, pages 466–470, 2021.

[196] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[197] Chih-Fong Tsai and Ming-Lun Chen. Credit rating by hybrid machine learning techniques. *Applied Soft Computing*, 10(2):374–380, 2010.

[198] Richard Berk, Hoda Heidari, Shahin Jabbari, Michael Kearns, and Aaron Roth. Fairness in criminal justice risk assessments: The state of the art. *Sociological Methods & Research*, 50(1): 3–44, 2021.

[199] Andrew Guthrie Ferguson. Big data and predictive reasonable suspicion. *University of Pennsylvania Law Review*, 163:327, 2014.

[200] Pradeep Kumar Roy, Sarabjeet Singh Chowdhary, and Rocky Bhatia. A machine learning approach for automation of resume recommendation system. *Procedia Computer Science*, 167: 2318–2327, 2020.

[201] José Luís Fava de Matos Pombo. *Landing on the right job: a machine learning approach to match candidates with jobs applying semantic embeddings*. PhD thesis, 2019.

[202] Peizhao Li and Hongfu Liu. Achieving fairness at no utility cost via data reweighing with influence. In *Proceedings of the 39th International Conference on Machine Learning*, 2022.

[203] Hanyu Song, Peizhao Li, and Hongfu Liu. Deep clustering based fair outlier detection. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021.

[204] Sanghamitra Dutta, Dennis Wei, Hazar Yueksel, Pin-Yu Chen, Sijia Liu, and Kush Varshney. Is there a trade-off between fairness and accuracy? a perspective using mismatched hypothesis testing. In *Proceedings of the International Conference on Machine Learning*, 2020.

[205] Rui Xu and Donald Wunsch. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16(3):645–678, 2005.

[206] Jian Liu and Y Zhou. Customer segmentation and fairness: A queueing perspective. Technical report, Working paper, University of Washington, Seattle, 2017.

[207] Mohsen Nazari and Seyed Parham Sheikholeslami. The impact of price segmentation based on customer characteristics, product bundle or product features on price fairness perceptions. *Journal of International Marketing Modeling*, 2(2):104–114, 2021.

[208] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

[209] G Anandalingam and Terry L Friesz. Hierarchical optimization: An introduction. *Annals of Operations Research*, 34(1):1–11, 1992.

[210] Omar Ben-Ayed and Charles E Blair. Computational difficulties of bilevel linear programming. *Operations Research*, 38(3):556–560, 1990.

[211] Yang Yu, Hong Qian, and Yi-Qi Hu. Derivative-free optimization via classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2016.

[212] Peizhao Li, Han Zhao, and Hongfu Liu. Deep fair clustering for visual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[213] Yann LeCun. USPS Dataset. `https://www.kaggle.com/bistaumanga/usps-dataset`, 1990. [Online; accessed June 30th 2022].

[214] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *Proceedings of the European Conference on Computer Vision*, 2010.

[215] Lei Xu, Adam Krzyzak, and Ching Y Suen. Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, 22(3):418–435, 1992.

[216] Elfarouk Harb and Ho Shan Lam. Kfc: A scalable approximation algorithm for k- center fair clustering. *Advances in Neural Information Processing Systems*, 2020.

[217] Hongjing Zhang and Ian Davidson. Deep fair discriminative clustering. *arXiv preprint arXiv:2105.14146*, 2021.

[218] M Emre Celebi, Hassan A Kingravi, and Patricio A Vela. A comparative study of efficient initialization methods for the k-means clustering algorithm. *Expert Systems with Applications*, 40(1):200–210, 2013.

[219] Frank J Massey Jr. The kolmogorov-smirnov test for goodness of fit. *Journal of the American statistical Association*, 46(253):68–78, 1951.

[220] Hongfu Liu, Zhiqiang Tao, and Zhengming Ding. Consensus clustering: an embedding perspective, extension and beyond. *arXiv preprint arXiv:1906.00120*, 2019.

[221] Ana LN Fred and Anil K Jain. Combining multiple clusterings using evidence accumulation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(6):835–850, 2005.

[222] André Lourenço, Samuel Rota Bulò, Nicola Rebagliati, Ana LN Fred, Mário AT Figueiredo, and Marcello Pelillo. Probabilistic consensus clustering using evidence accumulation. *Machine Learning*, 98(1):331–357, 2015.

[223] Xiaoli Zhang Fern and Carla E Brodley. Solving cluster ensemble problems by bipartite graph partitioning. In *Proceedings of the International Conference on Machine Learning*, 2004.

[224] Yang Hu, Haoxuan You, Zhecan Wang, Zhicheng Wang, Erjin Zhou, and Yue Gao. Graph-mlp: node classification without message passing in graph. *arXiv preprint arXiv:2106.04051*, 2021.

[225] Hao Wang, Defu Lian, Hanghang Tong, Qi Liu, Zhenya Huang, and Enhong Chen. Decoupled representation learning for attributed networks. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[226] Nan Yin, Fuli Feng, Zhigang Luo, Xiang Zhang, Wenjie Wang, Xiao Luo, Chong Chen, and Xian-Sheng Hua. Dynamic hypergraph convolutional network. In *Proceedings of the IEEE International Conference on Data Engineering*, 2022.

[227] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[228] Samira Pouyanfar, Saad Sadiq, Yilin Yan, Haiman Tian, Yudong Tao, Maria Presa Reyes, Mei-Ling Shyu, Shu-Ching Chen, and Sundaraja S Iyengar. A survey on deep learning: Algorithms, techniques, and applications. *ACM Computing Surveys (CSUR)*, 2018.

[229] Weibo Liu, Zidong Wang, Xiaohui Liu, Nianyin Zeng, Yurong Liu, and Fuad E Alsaadi. A survey of deep neural network architectures and their applications. *Neurocomputing*, 2017.

[230] Avrim L Blum and Pat Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 1997.

[231] Amirata Ghorbani and James Zou. Data shapley: Equitable valuation of data for machine learning. In *International Conference on Machine Learning*, 2019.

[232] Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nick Hynes, Nezihe Merve Gürel, Bo Li, Ce Zhang, Dawn Song, and Costas J Spanos. Towards efficient data valuation based on the shapley value. In *International Conference on Artificial Intelligence and Statistics*, 2019.

[233] Yongchan Kwon and James Zou. Beta shapley: a unified and noise-reduced data valuation framework for machine learning. In *International Conference on Artificial Intelligence and Statistics*, 2022.

[234] Amirata Ghorbani, Michael Kim, and James Zou. A distributional framework for data valuation. In *International Conference on Machine Learning*, 2020.

[235] Jiachen T Wang and Ruoxi Jia. Data banzhaf: A robust data valuation framework for machine learning. In *International Conference on Artificial Intelligence and Statistics*, 2023.

[236] Jinsung Yoon, Sercan Arik, and Tomas Pfister. Data valuation using reinforcement learning. In *International Conference on Machine Learning*, 2020.

[237] Florentin Woergoetter and Bernd Porr. Reinforcement learning. *Scholarpedia*, 2008.

[238] Peizhao Li and Hongfu Liu. Achieving fairness at no utility cost via data reweighing with influence. In *International Conference on Machine Learning*, 2022.

[239] Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. Deep learning on a data diet: Finding important examples early in training. In *Advances in Neural Information Processing Systems*, 2021.

[240] C Coleman, C Yeh, S Mussmann, B Mirzasoleiman, P Bailis, P Liang, J Leskovec, and M Zaharia. Selection via proxy: Efficient data selection for deep learning. In *International Conference on Learning Representations*, 2020.

[241] Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. Coresets for data-efficient training of machine learning models. In *International Conference on Machine Learning*, 2020.

[242] Yanyao Shen and Sujay Sanghavi. Learning with bad training data via iterative trimmed loss minimization. In *International Conference on Machine Learning*, 2019.

[243] Jie Cai, Jiawei Luo, Shulin Wang, and Sheng Yang. Feature selection in machine learning: A new perspective. *Neurocomputing*, 2018.

[244] Mark A Hall. *Correlation-based feature selection for machine learning*. PhD thesis, The University of Waikato, 1999.

[245] Kai Wei, Rishabh Iyer, and Jeff Bilmes. Submodularity in data subset selection and active learning. In *International Conference on Machine Learning*, 2015.

[246] Michele Donini, Luca Oneto, Shai Ben-David, John S Shawe-Taylor, and Massimiliano Pontil. Empirical risk minimization under fairness constraints. In *Advances in Neural Information Processing Systems*, 2018.

[247] István Megyeri, István Hegedűs, and Márk Jelasity. Adversarial robustness of linear models: regularization and dimensionality. 2019.

[248] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. *arXiv preprint arXiv:1412.6572*, 2014.

[249] Ali Rahmati, Seyed-Mohsen Moosavi-Dezfooli, Pascal Frossard, and Huaiyu Dai. Geoda: a geometric framework for black-box adversarial attacks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[250] Xiaochuang Han, Byron C Wallace, and Yulia Tsvetkov. Explaining black box predictions and unveiling data artifacts through influence functions. In *Annual Meeting of the Association for Computational Linguistics*, 2020.

[251] Zhun Deng, Cynthia Dwork, Jialiang Wang, and Linjun Zhang. Interpreting robust optimization via adversarial influence functions. In *International Conference on Machine Learning*, 2020.

[252] Ahmed Alaa and Mihaela Van Der Schaar. Discriminative jackknife: Quantifying uncertainty in deep learning via higher-order influence functions. In *International Conference on Machine Learning*, 2020.

[253] Abhineet Agarwal, Yan Shuo Tan, Omer Ronen, Chandan Singh, and Bin Yu. Hierarchical shrinkage: Improving the accuracy and interpretability of tree-based models. In *International Conference on Machine Learning*, 2022.

[254] Leo Breiman, Jerome H Friedman, Richard A Olshen, and Charles J Stone. *Classification and regression trees*. Routledge, 2017.

[255] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Large-scale celebfaces attributes (celeba) dataset. *Retrieved August*, 2018.

[256] David Noever. Machine learning suites for online toxicity detection. *arXiv preprint arXiv:1810.01869*, 2018.

[257] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems*, 2020.

[258] Yuji Roh, Kangwook Lee, Steven Euijong Whang, and Changho Suh. Improving fair training under correlation shifts. In *International Conference on Machine Learning*, 2023.

[259] Subha Maity, Debarghya Mukherjee, Mikhail Yurochkin, and Yuekai Sun. Does enforcing fairness mitigate biases caused by subpopulation shift? In *Advances in Neural Information Processing Systems*, 2021.

[260] Stephen Giguere, Blossom Metevier, Bruno Castro da Silva, Yuriy Brun, Philip Thomas, and Scott Niekum. Fairness guarantees under demographic shift. In *International Conference on Learning Representations*, 2022.

[261] Yuji Roh, Kangwook Lee, Steven Euijong Whang, and Changho Suh. Fairbatch: Batch selection for model fairness. In *International Conference on Learning Representations*, 2021.

[262] Ashkan Rezaei, Rizal Fathony, Omid Memarrast, and Brian Ziebart. Fairness for robust log loss classification. In *AAAI Conference on Artificial Intelligence*, 2020.

[263] Frances Ding, Moritz Hardt, John Miller, and Ludwig Schmidt. Retiring adult: New datasets for fair machine learning. In *Advances in Neural Information Processing Systems*, 2021.

[264] David Solans, Battista Biggio, and Carlos Castillo. Poisoning attacks on algorithmic fairness. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD*. Springer, 2021.

[265] Nan-Chen Hsieh. Hybrid mining approach in the design of credit scoring models. *Expert Systems with Applications*, 2005.

[266] Tim Brennan, William Dieterich, and Beate Ehret. Evaluating the predictive validity of the compas risk and needs assessment system. *Criminal Justice and Behavior*, 2009.

[267] Elaine Fehrman, Awaz K Muhammad, Evgeny M Mirkes, Vincent Egan, and Alexander N Gorban. The five factor model of personality and evaluation of drug consumption risk. In *Data Science: Innovative Developments in Data Analysis and Clustering*. Springer, 2017.

[268] Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. In *Advances in Neural Information Processing Systems*, 2020.

[269] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD*. Springer, 2013.

[270] Jacob Montiel, Max Halford, Saulo Martiello Mastelini, Geoffrey Bolmier, Raphael Sourty, Robin Vaysse, Adil Zouitine, Heitor Murilo Gomes, Jesse Read, Talel Abdessalem, et al. River: machine learning for streaming data in python. *The Journal of Machine Learning Research*, 2021.

[271] Thorsten Joachims. Training linear svms in linear time. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2006.

[272] Nikhil Ketkar. Stochastic gradient descent. *Deep learning with Python: A hands-on introduction*, 2017.

[273] Jan Kremer, Kim Steenstrup Pedersen, and Christian Igel. Active learning with support vector machines. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2014.

[274] Yinan Zhang and Mingqiang An. An active learning classifier for further reducing diabetic retinopathy screening system cost. *Computational and Mathematical Methods in Medicine*, 2016.

[275] Muhammad Ahtazaz Ahsan, Adnan Qayyum, Adeel Razi, and Junaid Qadir. An active learning method for diabetic retinopathy classification with uncertainty quantification. *Medical & Biological Engineering & Computing*, 2022.

[276] Etienne Decencière, Xiwei Zhang, Guy Cazuguel, Bruno Lay, Béatrice Cochener, Caroline Trone, Philippe Gain, Richard Ordonez, Pascale Massin, Ali Erginay, et al. Feedback on a publicly distributed image database: the messidor database. *Image Analysis & Stereology*,

2014.

[277] Bálint Antal and András Hajdu. An ensemble-based system for automatic screening of diabetic retinopathy. *Knowledge-based Systems*, 2014.

[278] Alex Holub, Pietro Perona, and Michael C Burl. Entropy-based active learning for object recognition. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, 2008.

[279] Maria-Florina Balcan, Andrei Broder, and Tong Zhang. Margin based active learning. In *Annual Conference on Learning Theory, COLT*. Springer, 2007.

[280] Vu-Linh Nguyen, Mohammad Hossein Shaker, and Eyke Hüllermeier. How to measure uncertainty in uncertainty sampling for active learning. *Machine Learning*, 2022.