The VLDB Journal
https://doi.org/10.1007/s00778-022-00775-9

REGULAR PAPER

Data collection and quality challenges in deep learning: a data-centric AI perspective

Steven Euijong Whang[1] · **Yuji Roh**[1] · **Hwanjun Song**[2] · **Jae-Gil Lee**[1]

Received: 12 December 2021 / Revised: 8 November 2022 / Accepted: 10 December 2022

Abstract
Data-centric AI is at the center of a fundamental shift in software engineering where machine learning becomes the new software, powered by big data and computing infrastructure. Here, software engineering needs to be re-thought where data become a first-class citizen on par with code. One striking observation is that a significant portion of the machine learning process is spent on data preparation. Without good data, even the best machine learning algorithms cannot perform well. As a result, data-centric AI practices are now becoming mainstream. Unfortunately, many datasets in the real world are small, dirty, biased, and even poisoned. In this survey, we study the research landscape for data collection and data quality primarily for deep learning applications. Data collection is important because there is lesser need for feature engineering for recent deep learning approaches, but instead more need for large amounts of data. For data quality, we study data validation, cleaning, and integration techniques. Even if the data cannot be fully cleaned, we can still cope with imperfect data during model training using robust model training techniques. In addition, while bias and fairness have been less studied in traditional data management research, these issues become essential topics in modern machine learning applications. We thus study fairness measures and unfairness mitigation techniques that can be applied before, during, or after model training. We believe that the data management community is well poised to solve these problems.

Keywords Data collection · Data quality · Deep learning · Data-centric AI

1 Overview

Deep learning is widely used to glean knowledge from massive amounts of data. There is a wide range of applications including natural language understanding, healthcare, self-driving cars, and more. Deep learning has become so prevalent thanks to its excellent performance with the availability of big data and powerful computing infrastructure. According to the IDC [41], the amount of data worldwide is projected to grow exponentially to 175 zettabytes (ZB) by 2025. In addition, powerful GPUs and TPUs enable software to have superhuman performances in various tasks.

We are going through a fundamental paradigm shift in software engineering where machine learning becomes the new software (referred to as Software 2.0 [134]). Conventional software engineering involves designing, implementing, and debugging code. In comparison, machine learning starts with data and trains a function on the data. It is known that data preparation is an expensive step in end-to-end machine learning. In particular, collecting data, cleaning it, and making it suitable for machine learning training takes 45% [43] or even 80–90% [24,153] of the entire time. In addition, the code on a machine learning platform (e.g., PyTorch [112]) is high level and thus, requires significantly fewer lines compared to conventional software. Finally, the trained model may need to be continuously improved with hyperparameter tuning. This entire process from data preparation to model deployment is widely viewed as a new

This article extends tutorials the authors delivered at the VLDB 2020 [169] and KDD 2021 [94] conferences.

✉ Steven Euijong Whang
swhang@kaist.ac.kr

Yuji Roh
yuji.roh@kaist.ac.kr

Hwanjun Song
hwanjun.song@navercorp.com

Jae-Gil Lee
jaegil@kaist.ac.kr

[1] KAIST, Daejeon, South Korea

[2] Naver AI Lab, Seongnam, South Korea

Published online: 03 January 2023

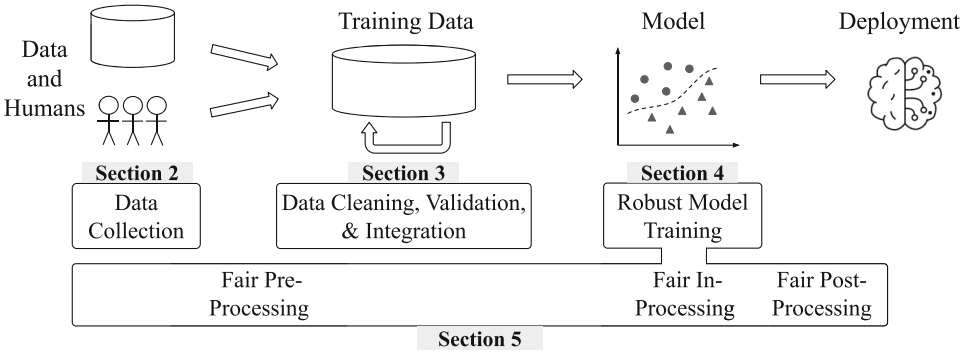

Fig. 1 Deep learning challenges from a data-centric AI perspective. Data collection and quality issues cannot be resolved in a single step, but throughout the entire machine learning process. This survey thus focuses on the breadth of available techniques

software engineering paradigm, and companies have been actively developing open source [13,30] and proprietary Software 2.0 systems [5,6,21,68,71,137]. Solving data issues is increasingly becoming critical in machine learning research.

While data collection and quality issues are important, machine learning research has mainly focused on training algorithms instead of the data. According to [153], a common complaint in the industry is that research institutions spend 90% of their machine learning efforts on algorithms and 10% on data preparation, although based on the amounts of time spent, the numbers should be 10% and 90% the other way.

At the same time, many companies are promising to use responsible and data-centric AI practices. For example, Google [120] says that AI has a significant potential to help solve challenging problems, but it is important to develop responsibly. Microsoft [121] pledges to advance AI using ethical principles that put people first. Other companies make similar statements [109,154]. More recently, data-centric AI [42] is becoming critical where the primary goal is not to improve the model training algorithm, but to improve the data pre-processing for better model accuracy.

These trends motivate us to investigate data collection and quality challenges for deep learning from a data-centric AI perspective. Figure 1 shows a simplified end-to-end process starting from data collection to model deployment. Deep learning systems are more complicated in practice [118,141], and we only show the essential steps. The first topic we cover is *data collection*. In comparison with traditional machine learning, in deep learning feature engineering is less of a concern, but there is instead a need for large amounts of training data. Unfortunately, many industries do not adopt deep learning simply because of the lack of data and the lack of explainability of the trained models. The second topic is *data cleaning and validation*. While there is a vast literature on data cleaning, unfortunately not all the techniques directly benefit deep learning accuracy [96]. In addition, there are recent deep learning issues including data poisoning that needs to be addressed, especially by the data management community. Data poisoning is becoming a significant threat as attackers generate data with a malicious intent to reduce the model accuracy of AI applications. In response, there is a branch of research called data sanitization where the goal is to defend against such attacks. The third topic is *robust model training*. Even after we carefully validate and clean our data, the data quality may still be problematic because there is no guarantee that we fixed all the data problems. Hence, we may still need to cope with dirty, missing, or even poisoned data in model training. Fortunately, there are various robust training techniques [149] available. The fourth topic is *fair model training*. Traditional research on data management has not focused on bias and fairness issues. However, in addition to cleaning and validating data to improve model accuracy, also showing fairness against biased data is becoming essential for responsible AI. In fact, many data validation works now mention that supporting AI ethics including fairness is an important future research direction [18]. Model fairness research [12,36,100,165] largely consists of fairness measures and unfairness mitigation before, during, or after model training. Recent studies are now addressing model fairness and robustness together due to their close relationship where data bias and noise may affect each other in the same training data [84,131,133,148].

While the coverage of this survey is broad, we believe it is important to have a birds-eye view of data issues in the entire deep learning process in order to advance data-centric AI. Each subtopic is not only substantial, but studied by different communities. Data collection, cleaning, and validation have been traditionally studied in the data management community. Robust model training is a central topic in the machine learning and security communities, while fair model training is a popular topic in the machine learning and fairness communities. Both fairness and robustness topics are increasingly being researched in the data management community as well because they are closely related to the input data. Data-centric AI is a nascent field that cannot be covered by solving just one of these areas either, but instead will ultimately need an orchestration within a holistic framework. Our contribution is thus to connect these related topics together at a high level with a focus on recent and significant works. Table 1 shows a taxonomy of the techniques covered in this survey. Figure 2

Table 1 Taxonomy of data collection and quality techniques for deep learning

Operation	Category	Section	Data type	Technique	Key References
Data Collection	Data Acquisition	2.1	All	Data discovery	[23,25,54,64,104,157,183]
				Data augmentation	[40,58–60,70,89,124,181]
				Data generation	[1,86,115,158,159]
	Data Labeling	2.2	All	Utilize existing labels	[160,174,188]
				Manual labeling	[2,56,129,142]
				Automatic labeling	[10,122,123,163]
	Improve Existing Data	2.3	Tabular	Improve labels and model	[62,110,146]
Data Validation, Cleaning, and Integration	Data Validation	3.1	Tabular & Image	Visualization	[93,164]
				False discovery control	[55,184]
				Schema-based validation	[13,22,117]
				New functionalities	[63,63,101,125,138–140,170]
	Data Cleaning	3.2	Tabular & Image	Data-only cleaning	[74,127]
				ML-aware cleaning	[47,48,82,88,96,107,128]
	Data Sanitization	3.3	Tabular & Image	Data poisoning	[143,187]
				Poisoning defenses	[39,72,87,114]
	Data Integration	3.4	Multimodal	Multimodal integration	[11,46,152]
Robust Model Training	Noisy Features	4.1	Image	Adversarial learning	[28,113,126]
	Missing Features	4.2	All	Data imputation	[29]
	Noisy Labels	4.3	Image	Robust learning	[65,78,95,99,149]
	Missing Labels	4.4	Image	Semi-supervised learning	[17,156]
Measuring Fairness	Statistical Fairness	5.1	All	Independence criteria ($\hat{Y} \perp Z$)	[51,53]
				Separation criteria ($\hat{Y} \perp Z \mid Y$)	[15,66,176]
				Sufficiency criteria ($Y \perp Z \mid \hat{Y}$)	[15,35,45]
	Other Fairness	5.1	All	Individual fairness	[51]
				Causal fairness	[83,85,90,106,182]
Unfairness Mitigation	Pre-processing	5.2	Tabular & Image	Repair data	[53,80,136]
				Generate data	[34,171]
				Acquire data	[9,31,155]
	In-processing	5.2	Tabular	Fairness constraints	[4,81,177]
				Adversarial training	[38,131,178]
				Adaptive reweighting	[76,77,132,179]
	Post-processing	5.2	Tabular	Fix predictions	[37,66,116]
	Convergence with Robustness	5.3	Tabular & Image	Fairness-oriented	[67,91,92,168]
				Robustness-oriented	[84,172,180]
				Equal Mergers	[131,133,148,167]

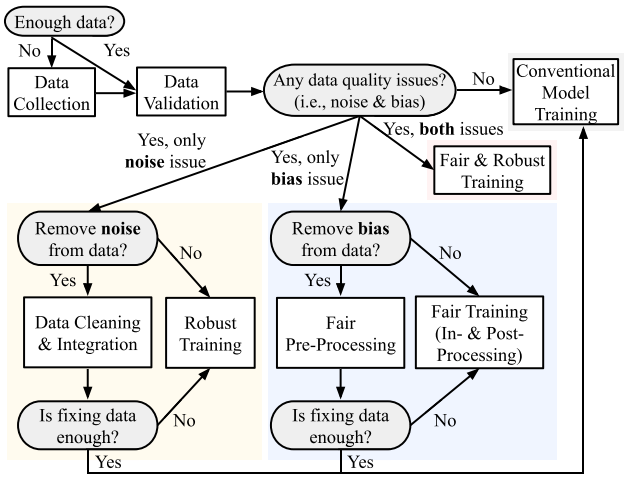

Fig. 2 Decision tree on how data-centric AI techniques connect with each other in one workflow

shows a decision tree of how the techniques connect with each other in one workflow. Our work targets researchers and practitioners who need a starting point of understanding how data plays a key role in data-centric AI.

In summary, deep learning is becoming prevalent thanks to big data and fast computation, and software engineering is going through a new paradigm shift. However, big data for deep learning have been relatively understudied, but is becoming critical in data-centric AI. We cover the following topics in the next sections:

- Data collection techniques for machine learning (Sect. 2).
- Data validation, cleaning, and integration techniques for machine learning (Sect. 3).
- Robust training techniques for coping with noisy and poisoned data (Sect. 4).
- Fair training techniques for coping with biased data (Sect. 5)
- Overall findings and future directions (Sect. 6).

We choose papers using three criteria. First, we include papers to cover the diverse areas in each section. Second, in each area, we select prominent papers preferably with many citations. Third, we cover recent techniques that are emerging, but are yet to be widely cited according to our judgment and tutorials.

Note that Table 1 also specifies the data types each technique focuses on. For both robust and fair training (Sects. 4, 5), we mainly consider supervised learning.

2 Data collection

Our coverage of data collection originates from a survey [130] by two of the authors, so we keep it brief with new updates based on a tutorial [169]. There are three main approaches for data collection. First, *data acquisition* is the problem of discovering, augmenting, or generating new datasets. Second, *data labeling* is the problem of adding informative annotations to data so that a machine learning model can learn from them. Since labeling is expensive, there is a variety of techniques to use including semi-supervised learning, crowdsourcing, and weak supervision. Finally, if one already has data, *improving existing data and models* can be done instead of acquiring or labeling from scratch.

2.1 Data acquisition

If there is not enough data, the first option is to perform data acquisition, which is the process of finding datasets that are suitable for training machine learning models. In this survey, we cover three approaches: data discovery, data augmentation, and data generation. *Data discovery* is the problem of indexing and searching datasets. *Data augmentation* takes labeled examples and distorts or combines them to generate synthetic examples. If there is not enough data around, the last resort is to take matters in one's own hands and create datasets using crowdsourcing or synthetic *data generation* techniques.

2.1.1 Data discovery

Data discovery is the problem of indexing and searching datasets that exist either in corporate data lakes [54,157] or the Web [25]. One example is the Goods system [64], which searches tens of billions of datasets in Google's data lake. Goods takes a post hoc approach where it crawls the datasets from multiple sources and extracts metadata to maintain a central dataset catalog, which does not require any work from the dataset owners. Each entry in the catalog contains metadata about one dataset including its size, provenance on which job created it and which job read it, and schema information. Goods provides search, monitoring, and dataset annotations as well. A public version of Goods called Google Dataset Search [23] supports science dataset searching. More recently, these data discovery tools have become more interactive. A representative system is Juneau [183], which is an interactive data search and management tool built on top of the Jupyter Notebook data science platform. Here, the key technical challenge is finding the related tables. Juneau uses similarity measures for comparing records and schemas and provenance information that intuitively captures the purpose of creating each data set. Finding tables that can be joined or unioned in data lakes efficiently is critical, and LSH-

based algorithms that perform set overlap search or unionable attribute retrieval on tables have been proposed [104].

2.1.2 Data augmentation

For data augmentation, a popular method for generating data in the machine learning community is generative adversarial networks (GANs) [59,60,89]. We start from a training set that has real data. There are two components: a *generator* that generates fake data that is realistic using some random noise as an input and a *discriminator* that tries to distinguish the real data from the fake data of the generator. The generator and discriminator are trained in an adversarial fashion. One limitation of a GAN is that it cannot generate data that is completely different than the existing data. Using policies [70,124] is a way to complement that limitation where one can apply various custom transformations provided by domain experts as long as the data remains realistic. AutoAugment [40] automates this process where the idea is to have a controller that suggests a strategy for applying transformations with certain probabilities and magnitudes on the data. The system then trains a child model on this augmented data and measures the accuracy on a validation set. This result is then used to decide whether the strategy produces useful data that is within realistic bounds and should thus be used.

The data augmentation literature continues to grow rapidly. Mixup [16,17,69,175,181] has been proposed as a simple, but effective augmentation technique where the key idea is to mix pairs of data points of different classes. The additional data effectively regularizes the model to predict in-between training data points assuming linearity. Model patching [58] utilizes GANs to augment the data of specific subgroups of a class so that the model accuracy is similar across subgroups.

2.1.3 Data generation

Another option for collecting or acquiring new data is to generate data. A popular option is to use crowdsourcing platforms like Amazon Mechanical Turk [1] where one can create tasks and pay human workers to create or find data. For example, a task may ask workers to find face images of a certain demographic from public websites [155]. In addition, one can use a simulator or generator for specific domains, e.g., Hermoupolis [115] for mobility data and Crash to Not Crash [86] for driving data. Domain randomization [158,159] is an effective technique for generating a wide range of realistic data from a simulator by varying its parameters. We note that GANs also generate new data, but they require sufficient amounts of real data for model training.

2.2 Data labeling

Once there are enough datasets, the next step is to label the examples. We cover data labeling techniques for utilizing existing labels and manually or automatically labeling from no labels.

2.2.1 Utilize existing labels

The traditional approach for labeling is semi-supervised learning [160,188] where the idea is to use existing labels to predict the other labels. One can utilize existing machine learning benchmarks [50,79] that provide labeled data for a variety of tasks. The simplest form is *Self-training* [174] where a model is trained on the available labeled data and then applied to the unlabeled data. Then, the predictions with the highest confidence values are trusted and added to the training set. This approach assumes that we can trust the high confidence, but there are other techniques including Tri-training [186], Co-learning [185], and Co-training [20] that do not rely on this assumption.

2.2.2 Manual labeling from no labels

If there are no labels to start with, but one has funds to employ workers, a standard approach is to use crowdsourcing platforms like Amazon Mechanical Turk to perform labeling. Since labeling is such an important task, there are labeling-specialized services like Amazon Sagemaker Ground truth [2] and Google Cloud Labeling [56]. When using Sagemaker, one can choose labeling tasks and recruit labelers who are assisted with a UI and tools to label the data. Sometimes, crowdsourcing may not be feasible because the workers do not have the right expertise. Hence, the last resort is to rely on domain experts, but this option can be expensive.

Active learning [129,142] is an effective method to reduce the crowdsourcing cost. The idea is to ask human labelers to label uncertain examples that, when answered, are likely to improve model accuracy the most. While a full coverage of active learning is out of scope, the example selection techniques can largely be categorized into identifying uncertain examples and using decision theoretic approaches to analyze the effect of a newly labeled example on the model accuracy.

2.2.3 Automatic labeling from no labels

Recently, *weak supervision* is becoming popular where the idea is to (semi-)automatically generate labels that are not perfect (therefore called "weak" labels), but at scale where the larger volume may compensate for the lower label quality. Weak supervision is useful in applications where there are few or no labels to start with. Early techniques include crowdsourcing and distant supervision [105], which uses external

knowledge bases to generate labels for the training data. More recently, data programming builds on these techniques where multiple labeling functions are developed and combined to generate weak labels.

Snorkel [10,122,123] is the seminal system for data programming. Given user-provided labeling functions (e.g., Python functions that detect spam), Snorkel combines them in one generative model by intuitively taking a probabilistic consensus. Then, given unlabeled data, Snorkel can generate probabilistic labels. The unlabeled data and the probabilistic labels are used to train a final discriminative model like a deep neural network. Another way to combine labeling functions is to use majority voting. Empirically, the number of labeling functions determines whether a generative model or majority voting is better. Snuba [163] automates the process of constructing labeling functions using a small labeled dataset, if that is available.

2.3 Improving existing data

In addition to searching and labeling datasets, one can also improve the quality of existing data and models. This approach is useful in several scenarios. Suppose the target application is novel or non-trivial where there are no relevant datasets outside, or collecting more data no longer benefits the model's accuracy due to its low quality. Here, a better option may be to improve the existing data. One effective approach is to improve the labels through *re-labeling*. Sheng et al. [146] demonstrates the importance of improving labels by showing the model accuracy trends against more training examples for datasets with different qualities. As the data quality decreases, even if more data is used, the accuracy of the model does not increase from some point and plateaus. In this case, the only way to improve the model accuracy is to improve the label quality, which can be done by re-labeling and taking majority votes on multiple labels per example. In fact, one could clean the entire data including labels, which naturally leads to the next section where we cover data validation, cleaning, and integration.

3 Data validation, cleaning, and integration

It is common for the training data to contain various errors. Machine learning platforms like TensorFlow Extended (TFX) [13] have data validation [117] components to detect such data errors in advance using data visualization and schema generation techniques. Data cleaning can be used to actually fix the data, and there is a heavy literature [74] on various integrity constraints. However, recent studies [75,96] show that cleaning the data before machine learning by only fixing well-defined errors does not necessarily benefit machine learning accuracy. Instead, it is more effective to clean *for machine learning* with the direct purpose of improving accuracy [107] and making the model training more robust to noise in the data [98]. A recent survey [75] mentions that robust training is considered more effective than data cleaning before model training. Data noise can also be adversarial where it contains malicious poisoning, and cleaning against this is called data sanitization in the security community. Yet another issue is incorporating AI ethics like model fairness [18] where data may be biased, which may cause the trained model to be discriminatory. So far the data validation literature does not cover robust and fair training in depth, but these areas are heavily studied in the machine learning community, so we make a connection by elaborating on their techniques in Sects. 4 and 5, respectively.

3.1 Data validation

Data visualization is a widely used and effective way to validate data for machine learning (see a tutorial [117] and survey [118]). Compared to traditional data cleaning, visualization is effective for a human to perform quick, but important sanity checks on the data to prevent larger errors downstream. A representative open-source tool is Facets [52], which shows various statistics and the contents of datasets that can be used for sanity checks on data to prevent larger errors downstream. In addition to manual visualization, there has also been research on *automatic generation* of new visualizations [93] that can be used for validation purposes. SeeDB [164] is a seminal framework that repeatedly generates visualizations of interest. To capture the notion of interestingness, SeeDB uses a deviation-based utility metric that gives a high value when groupings of the data result in different probability distributions.

Automatically generating visualizations can run into the problem of false positives, so there is also a line of research that proposes *false discovery control* techniques. CUDE [184] controls false discovery in the context of multiple hypothesis testing for visual interactive data exploration. Here, users can repeatedly generate visualizations and mark the ones that are significant. Based on this user feedback, the goal is to automatically choose the visualizations that are significantly interesting in a statistical sense.

Schema-based validation [13,117] is widely used in practice. Tensorflow Data Validation (TFDV) [22,49] assumes a continuous training setting where input data periodically streams in as shown in Fig. 3. TFDV generates a data schema from previous data sets and uses the schema to validate future data sets and alert users on data anomalies. For each anomaly, TFDV provides concrete action items to possibly fix the root cause. A schema here is different from a traditional database schema where it contains a summary of data statistics of the features. In case a new dataset violates the current schema,

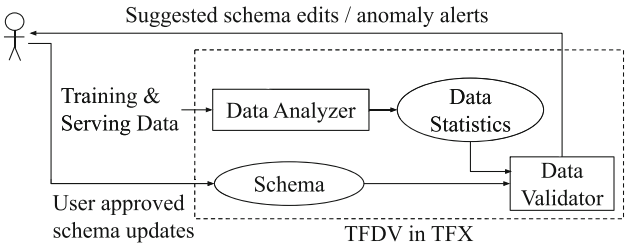

Fig. 3 TensorFlow Data Validation (TFDV) [22] uses a user-approved schema to validate the statistics of the training and serving data

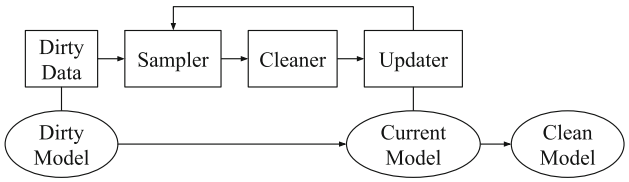

Fig. 4 ActiveClean [88] iteratively selects data that is likely to be dirty and cleans it

either the data needs to be fixed, or the schema needs to be updated, and the user makes the decision.

More recently, data validation systems are equipped with additional functionalities. Deequ [138,139] allows one to write data quality constraints declaratively, which then are automatically generated into unit tests. The mlinspect library [63] enables declarative machine learning pipeline inspection. Other additions include automatic identification of error types [125], testing the impact of errors on models [140], ease of usage [101], and efficient human-in-the-loop validation [170].

3.2 Data cleaning

Data cleaning has a long history of removing various well-defined errors by satisfying integrity constraints including key constraints, domain constraints, referential integrity constraints, and functional dependencies. For an introduction, see the book *Data Cleaning* [74]. There is also a recent survey on data cleaning techniques for machine learning and vice versa [75].

We first introduce one of the state-of-the-art data cleaning techniques to give a sense of how sophisticated these techniques have become. HoloClean [127] repairs data using probabilistic inference using three main ingredients: satisfying various integrity constraints, using external dictionaries to check the validity of values, and using quantitative statistics.

Unfortunately, only focusing on fixing the data does not necessarily guarantee the best model accuracy. At first glance, it seems that perfectly cleaning the data would be most useful for the model training. However, the notion of clean data is not always clear cut, and removing all possible errors is not always feasible. CleanML [96] is a framework that evaluates various data cleaning techniques and seeing if they actually help model accuracy. The authors show that data cleaning does not necessarily improve downstream machine learning models. In fact, the cleaning may sometimes have a negative effect on the models. However, by selecting an appropriate machine learning model, one can eliminate the negative effects of data cleaning. Also there is no single cleaning algorithm that performs the best, and one must adaptively choose the algorithm depending on the type of noise to clean. Moreover, many data cleaning primitives have high-impact parameters like thresholds that need to be tuned, similar to machine learning hyperparameter tuning. Hence, data cleaning techniques that are not originally designed for machine learning must be used carefully.

Recently, there are data cleaning techniques with the specific purpose of improving model accuracy [48]. ActiveClean [88] is a seminal framework that iteratively cleans samples of dirty data and updates the model. Figure 4 shows the workflow where there is a sampler that chooses an example that is likely to be dirty, and data quality rules can be used to identify such dirty samples. The reason for sampling data is that cleaning the entire data is presumed to be very expensive. Each sample can be cleaned by an oracle or domain expert. Then, the model is updated to be more accurate and also chooses the next sample. ActiveClean has theoretical guarantees where, by repeatedly training a model on the clean sample plus previously cleaned data, the model eventually obtains an accuracy as if it was trained on clean data only. ActiveClean assumes convex loss models like SVMs, and the data cleaning is assumed to be done perfectly.

Another branch of research is to clean the labels for the purpose of improving model accuracy. TARS [47] is a system that predicts model accuracy out of noisy labels that are produced from crowdsourcing. TARS first chooses labels that are likely to be flipped because they were labeled by poor-performing workers and thus have low confidence values. TARS then estimates how much the model will improve if the label is flipped after cleaning. The confidence values of labels can be computed by using confusion matrices of workers, which capture the history of how well they performed in past tasks. A confusion matrix thus contains the previous false positive, false negative, true positive, and true negative rates. Given the probability that a label is flipped, TARS estimates the resulting model accuracy and subtracts that by the estimated accuracy of the current model to determine whether the label is worth examining. Hence, TARS can selectively clean labels that are expected to benefit model accuracy the most.

More recently, there are more systematic approaches to support data cleaning for machine learning. One study [128]

shows how data quality issues affect MLOps and proposes various solutions to tackle them. For example, CPClean [82] is proposed to analyze how missing data impacts the certainty of predictions. Another work [107] distinguishes data cleaning *before* machine learning versus *for* machine learning and suggests to clean data throughout the entire machine learning pipeline. Some common challenges include handling multimodal data and data that change over time.

3.3 Data sanitization

Data poisoning has recently become a serious issue because changing a fraction of training data, which may come from an untrusted source, may alter the model's behavior. Compared to dirty data, there is a malicious intent to make the model fail. Data poisoning is a real problem because data are now easier to publish through dataset search engines. A dataset owner can simply post metadata to the public, which will be automatically crawled by the search engine. Then, one can simply harvest that data using web crawlers without knowing that the data is poisoned. Data sanitization [39] is the problem of defending against such poisoning attacks and can be viewed as an extreme version of data cleaning.

A simple type of data poisoning is called label flipping where a label of a training example is flipped from one class to another, but other works generate new data as well. Recently, data poisoning techniques have become much more sophisticated and therefore harder to defend against [143,187]. We illustrate a state-of-the-art data poisoning techniques for deep learning [187]. A major challenge when poisoning data for deep learning is that the victim's model cannot be easily analyzed. Hence, transferable poisoning attacks have been proposed, which can still succeed without accessing the victim's model. The idea is to train an ensemble of substitute models, which are assumed to be similar to the victim's model. Any attack that negatively affects the substitute models will presumably attack the victim's model as well. Given a set of clean data points of different classes, the poisoning algorithm adjusts the clean points to "move closer" to the target within the feature space and form a convex polytope that surrounds it to maximize the chances of the target to be misclassified.

How do we defend against such data poisoning using data sanitization? The main approach is to perform outlier detection to detect poisonings and discard them. Figure 5 shows a simple setting where a classifier's behavior changes after introducing poisoning (top right data points). If the data sanitization can identify and discard these points as outliers, then the model's accuracy can be restored. Compared to traditional outlier detection, the challenge is that poisonings are intentionally designed by the adversary to be difficult to detect while reducing model accuracy. Data sanitization techniques [39,72,114] have been proposed throughout the years, and a recent study [87] evaluates various defenses by developing attacks and seeing if the defenses work are still effective. Unfortunately, the conclusion is that no technique can adequately defend against carefully designed attacks. We suspect that data poisoning and sanitization techniques will continue to evolve and compete with each other.

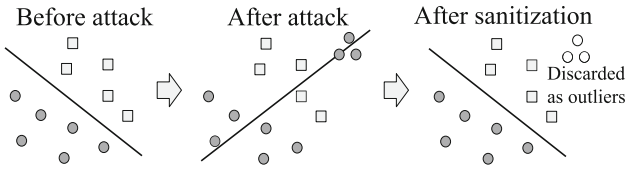

Fig. 5 Data Sanitization [39] identifies and discards data poisoning for better model accuracy

3.4 Multimodal data integration

Another dimension of data management to consider is the issue of multimodal data integration [11]. So far, we implicitly assumed single-source datasets, but in practice, data scientists often deal with multimodal data from multiple sources. For example, autonomous vehicles can generate a wide range of data including multiple video streams, radar and lidar data, and thousands of irregular times series from the Controller Area Network (CAN) of the vehicle. Analyzing all of this data together requires some form of data integration. In machine learning, two relevant integration techniques are alignment and co-learning. Alignment is to find relationships of sub-components of instances that have multiple modalities. For example, if there are multi-view time series, one can perform subsampling, forward or backward filling, or aggregate in time windows so that the time series can be better integrated. Co-learning is to train better on a modality using a different modality. For example, if there are embeddings from different modalities, one approach is to concatenate them together for a multimodal representation. In general, data integration is by itself a large research area that has been studied for decades [46,152], although not all techniques are relevant to machine learning.

4 Robust model training

Even after collecting the right data and cleaning it, data quality may still be an issue during model training. It is widely agreed that real-world datasets are dirty and erroneous despite the data cleaning process. As summarized in Table 2, these flaws in datasets can be categorized depending on whether data values are noisy or missing and depending on whether these flaws exist in data features (attributes) or labels.

Table 2 Types of data poisoning covered in this survey

	Noisy	Missing
Features	Adversarial Learning (Sect. 4.1)	Data Imputation (Sect. 4.2)
Labels	Robust Learning (Sect. 4.3)	Semi-Supervised Learning (Sect. 4.4)

The problem of data poisoning has been studied in theory (i.e., robust statistics) and practice for over fifty years and has gained a lot of attention in the machine learning community [73,161]. It starts with a basic question, 'can the machine learning model learn and predict as if the data was not corrupted?' and aims to develop machine learning algorithms robust to the worst-case corruptions where we cannot recover the entire clean information from the data. It mainly considers the corruptions in data features, which include outliers and adversarial examples.

Statistical approaches like robust mean estimation [97] aim to recover the mean of the distribution in the presence of data flaws. Convex programming [44] and filtering [33] address the problems by assigning a score to each data point based on the degree to which the sample is considered corrupted. This series of studies have been inspiring a lot of machine learning robustness optimization techniques such as loss reweighting and sample selection. In addition, robust machine learning involves many problems depending on what sorts of damages we consider. For example, privacy machine learning aims to respect the privacy of the users providing the data [119].

4.1 Noisy features

Noisy features are often introduced by adversarial attacks. Among several types of attacks, we focus on the *poisoning attack*, which is known as contamination of the training data, to be aligned with the main theme of this survey. During the training phase of a machine learning model, an adversary tries to poison the training data by injecting maliciously designed data to deceive the training procedure. Besides the adversarial noise, noisy features can include natural noise like image blurring and color noise, possibly not removed by data cleaning. There have been some approaches to successfully denoising the natural noise using Sparse Coding [144] and Feature Attention [8], but they are out of the scope of this survey.

Either features or labels or both can be the target of the poisoning attack. The poisoning attack can be done in three ways depending on the capability of adversaries. First, an adversary can randomly perturb the labels, i.e., by assigning other incorrect labels, picked from a random distribution, to a subset of training data. Since the label flipping result in overfitting to wrong labels like noisy labels, the robust training methods for this type of attack will be discussed in Sect. 4.3. Second, an adversary is more powerful and can corrupt the features of the examples possibly determined by analyzing the training algorithm [19]. The corrupted features deceive the model into making wrong predictions. Third, unlike manipulating the features, an adversary may add adversarial examples into the training data such as out-of-distribution examples. These examples lead to a sharp drop in generalization capability of machine learning models under distributional shifts. For more details, the reader can refer to [27,145].

Various defense strategies have been actively studied for robust training on adversarial examples (e.g., noisy features). Most of the current strategies are not adaptive to all types of attacks, but are effective to only a specific type. We summarize a few well-known, representative strategies in this section.

Most notably, in *adversarial training*, the robustness of a model can be improved using a modified objective function based on the fast gradient sign method [61]. It is defined as a weighted sum of an usual loss function on clean examples and the loss function on adversarial examples. By this regularization, the model is forced to predict the same class for legitimate and perturbed examples in the same direction.

Knowledge distillation has been shown to be effective for adversarial attacks [111]. *Defensive distillation* is almost the same as typical knowledge distillation, except that the same network architecture is used for both the original network and the distilled network. Specifically, instead of hard labels, where only the true label has the probability 1 in a probability vector, *soft targets*, which are generated by the original network as the prediction result, are used for training the distilled network. The benefit of using soft targets comes from the additional knowledge found in probability vectors compared to hard class labels [111].

Feature squeezing [173] reduces the degree of freedom available to an adversary by squeezing out unnecessary input features. If the original and squeezed inputs result in substantially different outputs by a model, the corresponding input is determined to be adversarial. A popular squeezing technique for images is reducing the color depth on a pixel level.

Another idea is to detect adversarial examples using separate classification networks [103]. A sub-network, called an *adversary detection network* or simply a detector, is trained to produce an output that indicates the probability of the input being adversarial. For this purpose, a classification network is trained using only non-adversarial examples, and adversarial examples are generated for each example in the training set. Then, the detector is trained using both the original dataset and the corresponding adversarial dataset. MagNet [102] falls into this category, and it also contains a

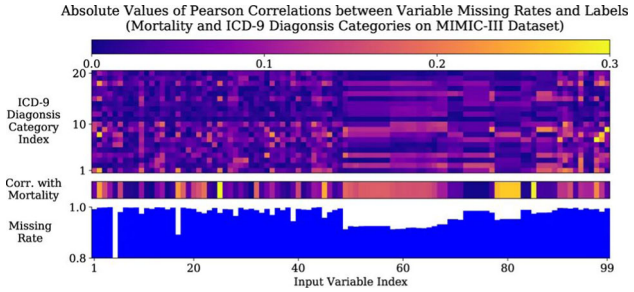

Fig. 6 Informative missingness in the MIMIC-III dataset [29]

reformer that corrects an adversarial example to a similar legitimate example using autoencoders.

4.2 Missing features

Since missing data can reduce the statistical power and produce biased estimates, data imputation has been an active research topic in statistics and machine learning. Missing features can occur in any types of data, but, in this paper, we focus on *multivariate time-series* data because its high input rate and sensor malfunction generate missing values very often.

Missing values in multivariate time-series data are ubiquitous in many practical applications ranging from healthcare, geoscience, astronomy, to biology and others. They often inevitably carry missing observations due to various reasons, such as medical events, saving costs, anomalies, inconvenience, and so on. These missing values are usually informative where the missing value and patterns provide rich information about target labels in supervised learning tasks.

We first describe informative missingness. Figure 6 shows MIMIC-III critical care dataset [29]. Starting from the bottom, there are the missing rate of each variable, the correlation between missing rate of each variable and mortality, and the correlation between missing rate of each variable and each ICD-9 diagnosis category. Here, we observe that the values of missing rates are correlated with labels, where the values with low missing rates are highly correlated with the labels. In other words, the missing rate of variables of each patient is useful, and this information is more useful for the variables that are more often observed in the dataset.

For existing approaches, a simple solution is to omit the missing data and to perform analysis only on the observed data, but it does not provide good performance when the missing rate is high and the samples are inadequate. Another solution is to fill in the missing values with substituted values, which is known as *data imputation*. However, these methods do not capture variable correlations and may not capture complex patterns to perform imputation. Combining the imputation methods with prediction models often results in a two-step process where imputation and prediction models are separated; the missing patterns are not effectively explored in the prediction model, thus leading to suboptimal analysis results.

GRU-D [29] is a deep learning model based on the gated recurrent unit (GRU) to effectively exploit two representations of informative missingness patterns—masking and time interval. Masking informs the model of which inputs are observed or missing, while time interval encapsulates the input observation patterns. GRU-D captures the observations and their dependencies by applying masking and time interval, which are implemented using a decay term, to the inputs and network states of the GRU, and jointly train all model components through back-propagation. GRU-D not only captures the long-term temporal dependencies of time-series observations, but also utilizes the missing patterns to improve the prediction results.

We elaborate on the two components of GRU-D: *making* and *time interval*. The value of a missing variable tends to be close to some default value if its last observation happened a long time ago, because the influence of the last observation fades away over time. As lots of missing patterns are informative and potentially useful in prediction tasks, but unknown and possibly complex, the goal is to learn decay rates from the training data rather than fixing them a priori. The GRU-D model incorporates two different trainable decay mechanisms. For a missing variable, an input decay γ_x is added to decay it over time toward the empirical mean, instead of using the last observation as it is. A hidden state decay γ_h in GRU-D has an effect of decaying the extracted features (GRU hidden states) rather than raw input variables directly.

We now extend the discussion to cover tabular data and present interesting studies in statistics, machine learning, and query optimization.

- *Statistics* MICE [162], which is one of the most commonly used packages in R, creates multiple imputed datasets to take care of uncertainty in missing values. By default, linear regression is applied to predict missing values. Besides, users can build models on all imputed datasets for evaluation and combine the results from these models to obtain a consolidated output.
- *Machine learning* XGBoost [32] internally handles missing values. It implements gradient boosted decision trees, and node splits are determined by considering missing values. In more detail, when a value is missing, the instance is classified into the default direction because there is nothing to evaluate for the split criteria. Here, the optimal default directions are learned from the data.
- *Query optimization* ImputeDB [26] selectively applies imputation to a subset of records dynamically during

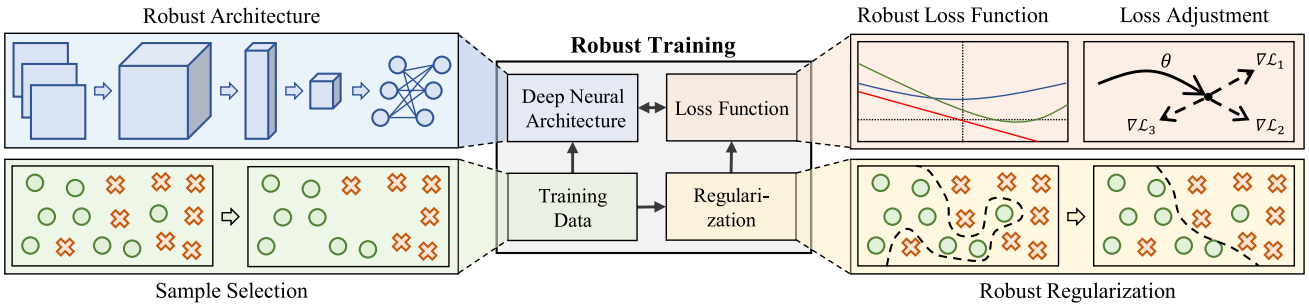

Fig. 7 Robust training categorization [149]

query execution. The rationale behind this optimization is that the subset which imputation is needed for a *specific* query is generally much smaller than the entire database. Thus, the computation spent for imputation is significantly saved.

4.3 Noisy labels

Because data labeling is done manually in many cases, incorrect or missing labels are, in fact, very common; the proportion of incorrect labels is reported to be 8–38% in several real-world datasets [150]. As an example of ANIMAL-10N [149], which is real-world noisy data with human annotation errors, human annotators mistakenly classified the Cheetah images as other animals like Jaguars instead of Cheetahs. In this example, it may be difficult to distinguish the patterns of Cheetahs and Jaguars, resulting in noisy labels in training data. So wrong annotations can be caused by such human errors. Similarly, labeling errors occur with data types other than images. For sentiment analysis, annotators often disagree on the polarity (e.g., positive or negative) of the sentiment expressed in the text [166]. Another type of error is software error. If there are many images to annotate, one may use automatic object recognition software. However, the object recognition itself may have errors. Thus, many deep learning techniques have been developed to consider the existence of label noises and errors, which are more critical in deep learning than in conventional machine learning as a deep neural network completely memorizes such noises and errors because of its high expressive power.

We explain what kinds of problems occur with noisy labels. In standard supervised learning, training data consist of example and label pairs $\{(x_i, y_i)\}_{i=1}^{N}$. In a practical setting, however, the label y_i is actually $\tilde{y}_i$, which means it can be incorrect. If one trains powerful models like VGG-19 on noisy data, the model may simply memorize the noise as well and perform worse on clean data. The goal of the noisy label problem is to train the network as if there are no noisy labels.

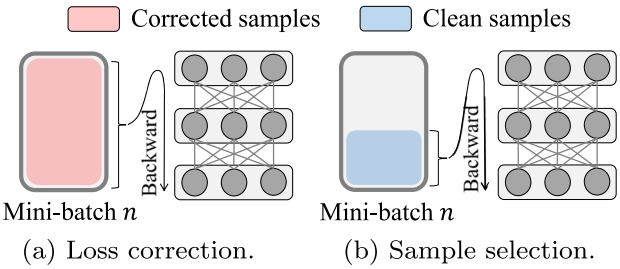

Fig. 8 Two directions of robust training covered in this survey

Figure 7 from a recent survey [149] shows the categorization of robust training techniques. There are largely four components in the training procedure: deep neural architecture, loss function, input training data, and regularization. For each component, there are relevant robust training techniques. For deep neural architectures, robust architectures have been developed. For training data, various sample selection techniques have been proposed. For the loss function, robust loss functions and loss adjustment techniques have been proposed. More specifically, loss adjustment can be further divided into loss correction [113], loss reweighting [28], and label refurbishment [126]. For regularization, robust regularization techniques have been proposed.

In this survey, we focus on the most representative techniques: sample selection and loss correction techniques as illustrated in Fig. 8. *Loss correction* is to correct the loss of *all* samples before a backward step. The representative techniques include Bootstrap [126], F-correction [113], and ActiveBias [28]. *Sample selection* is to select *expectedly clean* samples to update the network. The representative techniques include Decouple [99], MentorNet [78], and Coteaching [65].

We first introduce ActiveBias [28], which is a loss correction technique. ActiveBias performs a forward step on a given mini-batch and computes the sample importance for each sample. There are many statistics for the importance, e.g., variance of predictions. ActiveBias then corrects the loss by multiplying the normalized importance. If a label is noisy, then its importance μ_i decreases. The corrected loss is used

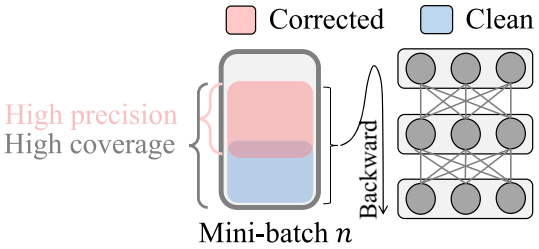

Fig. 9 SELFIE is a hybrid of loss correction and sample selection [149]

to update the network. We then explain sampling selection through its representative technique Coteaching [65], where the noise rate τ is assumed to be given. It then performs a forward step on a given mini-batch and selects the $(100 - \tau)\%$ low-loss samples as clean samples. The network is updated using the loss of the clean samples.

Although these two methods have improved the robustness to noisy labels, there are limitations of the two methods. Loss correction suffers from accumulated noise due to the large number of false corrections. Since all the examples are used for the training step, false corrections can accumulate for heavily noisy data. On the other hand, sample selection uses only clean samples having low losses and easy to classify. Hence, we may end up ignoring many useful, but hard samples that are classified as unclean.

SELFIE [149] was proposed to overcome the above limitations by using a hybrid of loss correction and sample selection (see Fig. 9). SELFIE introduces *refurbishable* samples where labels can be corrected with high precision. The key issue of SELFIE is constructing the refurbishable and clean samples. For the clean samples, SELFIE adopts the small-loss trick [65] and uses the $(100 - \tau)\%$ low-loss samples in the mini-batch. The refurbishable samples are ones that have consistent label predictions. Each label is replaced with the most frequently predicted label during the training step. For example, if an image is predicted mostly as a dog and only sometimes a cat, then the label predictions are considered consistent, and such a cat label is considered noisy and corrected to a dog. Finally, the loss is calculated for the refurbishable samples with correct labels and the clean samples; the samples that are neither refurbishable nor clean are discarded. The advantage of SELFIE is that it minimizes false corrections during the model training by selectively correcting refurbishable samples. As a result, the correction error of refurbishable samples is low. Also as the training progresses, the number of refurbishable samples also increases, so most training samples are exploited in the end.

Prestopping is another technique [149] for avoiding overfitting to noisy labels by early stopping the training of a deep neural network before the noisy labels are severely memorized. The algorithm resumes training the early stopped network using a maximal safe set, which maintains a collection of almost certainly true-labeled samples. MORPH [151] further improves Prestopping through a novel concept of *self-transitional learning*, which automatically switches its learning phase at the transition point. The optimal transition point is determined without any supervision such as a true noise rate and a clean validation set, which are usually hard to acquire in real-world scenarios. MORPH rather estimates the noise rate by fitting the loss distribution to a one-dimensional and two-component Gaussian mixture model (GMM).

DivideMix [95] is a recent technique that trains two networks simultaneously. At each epoch, a network models its per-sample loss distribution with a GMM to divide the dataset into a labeled set (mostly clean) and an unlabeled set (mostly noisy), which is then used as training data for the other network (i.e., co-divide). At each mini-batch, a network performs semi-supervised training using an improved MixMatch [17] method, which we cover in the next section. When training on the CIFAR-10 dataset with 40% asymmetric noise, standard training with cross-entropy loss causes the model to overfit and produce over-confident predictions, making the loss difficult to be modeled by the GMM. Also, adding a confidence penalty during the warm up leads to more evenly distributed loss. Finally, training with DivideMix can effectively reduce the loss for clean samples while keeping the loss larger for most noisy samples.

4.4 Missing labels

We cover the issue of missing labels where training labels may not exist for either some or all examples. There are largely semi-supervised and unsupervised approaches. In semi-supervised approaches, clean labeled data exists together with unlabeled (or incorrectly labeled) data. The goal is to exploit unlabeled data to improve accuracy as much as possible. Here, the loss is defined as the supervised loss for labeled data plus the unsupervised loss for unlabeled data. The representative techniques include unsupervised loss (e.g., consistency loss) like Mean-Teacher [156] and augmentation techniques like MixMatch [17]. For unsupervised approaches, the representative techniques include self-supervised learning and generative models, and we will cover a self-supervised learning technique called JigsawNet [108].

In Mean-Teacher [156], the teacher model is the average of consecutive student models. Both the student and teacher models evaluate the input in a training batch. The softmax output of the student model is compared with the one-hot label using a classification cost. Additionally, the output is compared with the teacher output using the consistency loss. After the weights of the student models are updated via gradient descent, the teacher model weights are updated as an exponential moving average of the student model weights.

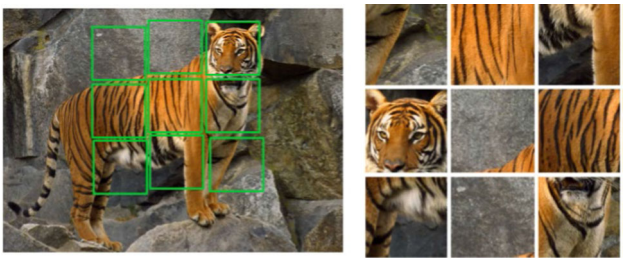

Fig. 10 Example of the jigsaw puzzle task for a given unlabeled image [108]

A training step with *unlabeled* examples is done without the classification cost.

In MixMatch [17], to exploit an unlabeled dataset, it performs label guessing where stochastic data augmentation is applied to an unlabeled image K times; then, each augmented image is fed through the classifier. The average of these K predictions is sharpened by adjusting the distribution's temperatures. The MixMatch algorithm mixes both labeled examples and unlabeled example with label guesses. In more detail, when mixing two images, the images are overlayed, and the labels are averaged, following the MixUp augmentation [181].

We now proceed to unsupervised techniques. Since there are no labels, we need to develop new tasks exploiting labels that can be obtained from the data for free. JigsawNet [108] is one of such techniques. If an image is divided into smaller regions without labels, we can randomize the regions and solve the jigsaw puzzle where we know the correct order and positions, as illustrated in Fig. 10. JigsawNet trains a context-free network (CFN) to solve the jigsaw task. The trained network can be transferred or fine-tuned to solve the given task using a small amount of labeled data.

5 Fair model training

We now focus on the issue of model fairness where biased data may cause a model to be discriminating and thus unfair. This problem is closely related to robust model training where instead of addressing noise in the training data, the goal is to address bias. A famous example is the COMPAS tool by Northpointe, which predicts a defendant's risk of committing another crime. According to an analysis by ProPublica [7], black defendants are far more likely to be judged as high risk compared to white defendants, which turns out to be inaccurate in practice. Other popular examples include an AI-based recruiting system discriminating against job applicants by gender [3], an AI-based photo app tagging people of a certain race inappropriately [57], and an AI chatbot generating hate speech towards minorities [135]. These incidences fueled the new research area of algorithmic fairness.

There could be multiple reasons why COMPAS discriminates. The training data could be biased where there is more data for certain demographics. Or there can be external factors where the surrounding environment may have caused more crime than race itself. Even the fairness measure can be in question where it does not accurately reflect reality. In general, analyzing fairness can be an extremely complicated issue that involves factors outside the data.

An extensive discussion on fairness and ethics can be found in the recent fair ML book [12], and here, we only focus on fairness issues with technical solutions. In particular, we discuss how to measure fairness and how to mitigate unfairness. In addition, we discuss a recent trend of how fair and robust techniques are converging. This trend is natural, as bias and noise can affect each other, and only addressing fairness may negatively affect robustness and vice versa. This section extends recent tutorials [94,169] by the authors.

5.1 Fairness measures

Fairness cannot be described by one notion, and there are tens of possible definitions summarized in various surveys [12,36, 100,165] used for predicting crime, hiring, giving loans, and more. We illustrate representative measures using a running example and then categorize them according to reference [12] as shown in Table 1 on Page 3. We use the following notations: Y denotes the label of a sample, $\hat{Y}$ the prediction of a model, and Z is a sensitive attribute like race or gender. Choosing a sensitive attribute depends on what is considered sensitive in the application. For example, if a company may run into trouble by discriminating based on age, then an attribute that is related to age can be considered sensitive.

We illustrate fairness using the simplest-possible model: a perceptron, which is the most basic unit in a neural network. Suppose the perceptron receives three input features: "Race = black" has a value of one if the person is black (e.g., Z = 0) or zero otherwise (Z = 1). "Race = white" has one if it is a white person or zero otherwise. "Previous crime" is one if the person has a previous crime and zero otherwise. The last feature is a constant to make the prediction threshold equal to zero. Given an example, we take the weighted sum by multiplying the feature values with the weights [2, 1, 1, -2] and, if the sum is at least zero, the model predicts (i.e., the $\hat{Y}$ value) recidivism (i.e., $Y = 1$) and otherwise not (i.e., $Y = 0$). For example, if a white person committed a previous crime, the weighted sum is $0 \times 2 + 1 \times 1 + 1 \times 1 - 1 \times 2 = 0$, which is larger or equal to the threshold zero. The interpretation is that the person previously committed a crime, so is likely to re-offend. A black person who committed a previous crime gets the same prediction. However, for people who did not commit a previous crime, the model starts to discriminate where only a black person is predicted to still re-offend as shown in Fig. 11. The prediction is obviously unfair and is

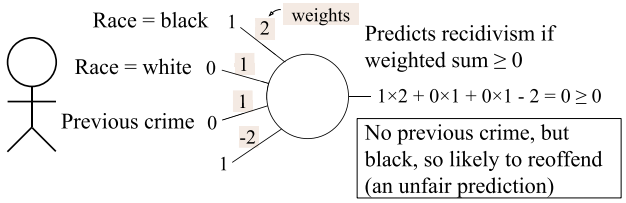

Fig. 11 A perceptron receiving input features and performing a weighted sum. Even a model as simple as a perceptron may show unfairness

shown for illustration purposes to show how even a model as simple as a perceptron can be discriminating.

For our running example, let us assume there are four people: (*a*) one white person who committed a crime before and committed a crime again, (*b*) one white person who never committed a crime, (*c*) one black person who committed a crime before and committed a crime again, and (*d*) one black person who never committed a crime. In this example, the perceptron correctly predicts for a, b, and c but wrongly predicts for d.

We summarize the prominent group fairness measures for fairness.

- *Demographic parity* [51,53] requires that sensitive groups must have the same positive prediction rates. The formulation is as follows: $P(\hat{Y} = 1|Z = 0) = P(\hat{Y} = 1|Z = 1)$ where the Z value indicates the sensitive group. $\hat{Y} = 1$ means that the prediction of the model is positive, e.g., predicting recidivism. Demographic parity says that the positive prediction rates of the two groups must be the same. In our running example, $P(\hat{Y} = 1|Z = 0) = 0.5$ while $P(\hat{Y} = 1|Z = 1) = 1$, which shows unfairness.
- *Equalized odds* [15,66,176] is defined as $P(\hat{Y} = 1|Z = 0, Y = A) = P(\hat{Y} = 1|Z = 1, Y = A), A \in \{0, 1\}$. That is, we would like to guarantee demographic parity when the label Y is zero (in our example, the person did not commit crime again) and when Y is one (the person committed crime) separately. In other words, equalized odds says that the accuracy conditioned on the true label must be the same for the groups. In our running example, $P(\hat{Y} = 1|Z = 0, Y = 1) = P(\hat{Y} = 1|Z = 1, Y = 1) = 1$, but $P(\hat{Y} = 1|Z = 0, Y = 0) = 0 \neq P(\hat{Y} = 1|Z = 1, Y = 0) = 1$, so there is some unfairness.
- *Predictive parity* [15,35,45] is defined as $P(Y = 1|Z = 0, \hat{Y} = 1) = P(Y = 1|Z = 1, \hat{Y} = 1)$. That is, given that the predictions are positive, we would like the actual likelihood of the label being positive to also be the same. Note that this measure can be extended to other label classes (e.g., $Y = 0, \hat{Y} = 0$). In our running example, $P(Y = 1|Z = 0, \hat{Y} = 1) = 1 \neq P(Y = 1|Z = 1, \hat{Y} = 1) = 0.5$, which shows unfairness.

Interestingly, many statistical fairness measures are equivalent to or variants of the following fairness criteria [12]: independence: $\hat{Y} \perp Z$, separation: $\hat{Y} \perp Z|Y$, and sufficiency: $Y \perp Z|\hat{Y}$. Note that demographic parity is equivalent to independence, equalized odds is equivalent to separation, and predictive parity is equivalent to sufficiency. An impossibility result says that no two fairness criteria can be fully satisfied together (see proofs in [12]).

There are remaining fairness criteria beyond the three above, and we cover the two popular ones: individual fairness and causality fairness.

- *Individual fairness* [51] only uses the classifier for its definition and is defined as $D(f(x), f(x')) \leq d(x, x')$ where d is a distance function among examples, and D is a distance function between outcome distributions. Intuitively, the predictions for similar people must be similar as well. For example, if two people are similar to each other, then their recidivism rates must be similar as well. Choosing proper distance functions is a key challenge in individual fairness.
- *Causality fairness* [83,85,90,106,182] assumes a causal model, which is a diagram of relationships between attributes. An edge from attribute A to attribute B means that A's value affects B's value. For example, suppose that race not only affects crime, but also the zip code of a person's address, which provides an environment for committing more or less crime. A causal graph could have three nodes race, zip code, and crime with edges from race to zip code, race to crime, and zip code to crime. One can perform a counterfactual analysis to see if zip code indeed affects crime rates by comparing similar people that live or do not live in that zip code.

5.2 Unfairness mitigation

Although there are many ways to measure fairness, one would ultimately like to perform *unfairness mitigation* [12,14]. Data bias can be addressed either before, during, or after model training. These approaches are referred to as *pre-processing*, *in-processing*, and *post-processing* approaches, respectively. Pre-processing approaches can be viewed as data cleaning, but with a focus on improving fairness. For each approach, we cover representative techniques.

Pre-processing mitigation

The goal is to fix the unfairness before model training by removing data bias. The advantage is that we may be able to solve the root cause of unfairness within the data. A disadvantage is that it may be tricky to ensure that the model fairness actually improves when we only operate on the data. A naïve approach that does not work is to remove sensitive attributes (also referred to as unawareness) because they are

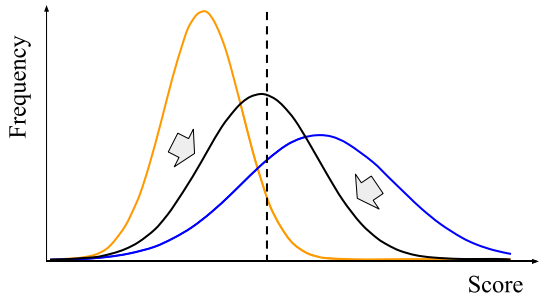

Fig. 12 Repairing distribution by averaging scores of the same percentiles [53]

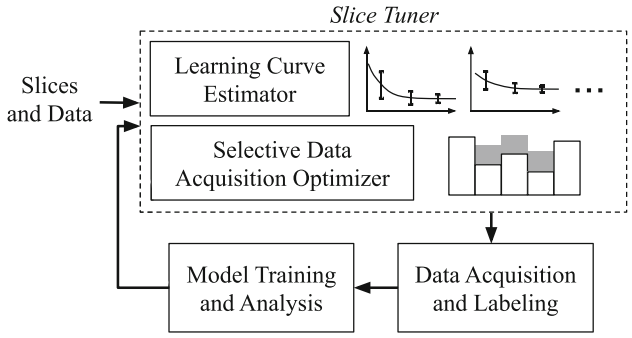

Fig. 13 Slice Tuner [155] is a selective acquisition framework for accurate and fair models where it iteratively estimates learning curves to determine how much data to acquire per data slice

usually correlated with other attributes. For example, removing sensitive attributes like race, income, and gender does not ensure fairness because their values can be inferred using correlated attributes like zip code, credit score, and browsing history, respectively. We cover three natural approaches for pre-processing—repairing data, generating data, and acquiring data.

For fair data repairing, we first cover a method [53] that guarantees demographic parity while preserving important statistics like the ranking of data. As an example (Fig. 12), let us say that we have test scores and distributions for different genders where the data follow normal distributions, but the women's distribution has a higher mean and larger variance. Now, let us say that we want to train a model that uses the test score to make a prediction. If we keep the men and women distributions as they are, then a model using a single threshold is going to be unfair for male versus female and violate demographic parity. Hence, the idea is to combine the two distributions by averaging the scores of the same percentile without losing the ranking information. This method can be extended to more than two sensitive groups.

A more recent system called Cappucin [136] repairs data such that a new causality-based fairness called interventional fairness is satisfied. The key insight is that satisfying interventional fairness can be reduced to satisfying multivalued functional dependencies (MVDs). The authors then propose minimal repair methods for MVDs by reducing the problem to MaxSAT or matrix factorization problems.

If there is not enough data to satisfy fairness, an alternative is to generate new data using the available data. A recent method [34] is to generate unbiased data using weak supervision. The input is biased data and an unbiased data that is smaller than the biased data that we have some control on. The idea is to train a generative model on the bias data except that we are adjusting the example weights such that it is as if the generative model is being trained on unbiased data. Then, the generative model generates new data that is unbiased. An example weight reflects how likely the example is part of the biased or unbiased data and can be computed by training a separate classifier for distinguishing the biased data from the unbiased data. The generative model that uses the example weights for training is guaranteed to produce unbiased synthetic data. In addition, GANs [171] have also been used for data generation where a generator competes with two discriminators: one for telling apart real and fake data and another for predicting the sensitive attribute.

As data are increasingly available, acquiring data from external data sources is also becoming a viable option [9, 31]. A recent approach called Slice Tuner [155] selectively acquires examples with the purpose of maximizing both accuracy and fairness of the trained model (Fig. 13). Slice Tuner assumes a set of non-overlapping data slices (e.g., regions), and the fairness measure is equal error rates [165] where the model's accuracies on different slices must be similar. The key idea is to maintain learning curves of slices, which can be used to predict accuracies on those slices given more data. Slice Tuner then solves a convex optimization problem to determine the amount of data to acquire per slice. Two challenges are that learning curves may be unreliable and that acquiring data for one slice may influence the model's accuracy on another slice. Slice Tuner solves these problems by iteratively updating the learning curves using a proxy for estimating influence. As a result, a model can obtain better accuracy and fairness compared to various baselines given a fixed budget for data acquisition. Another system called Deepdiver [9] performs data acquisition such that all possible slices contain sufficient amounts of data. Here, the slices may overlap with each other, and the objective is to guarantee minimum coverage instead of improving model accuracy or fairness.

In-processing mitigation We now cover representative in-processing techniques for unfairness mitigation where the model training is fixed. The advantage is that one can directly optimize accuracy and fairness. On the other hand, the downside is that the model training itself may have to change significantly, which may not be feasible in many applications. There are largely three in-processing approaches. The first is to directly modify the objective function of the model train-

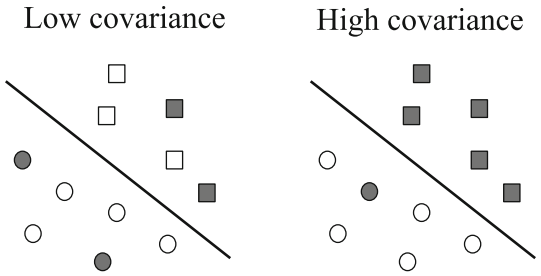

Fig. 14 Suppose we are classifying circles versus squares where the color of the shapes indicate the sensitive group. The left image has little correlation between the sensitive group and being on one side of the decision boundary, which means the covariance is low. The right image, on the other hand, shows a high covariance where just by looking at the sensitive group, one can figure out whether the data point is on which side of the decision boundary. Hence, by minimizing the covariance, one can also make the sensitive attribute more independent of the model predictions and thus satisfy demographic parity as well

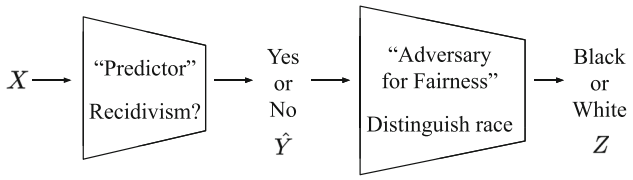

Fig. 15 Adversarial de-biasing [178] competes a classifier with a fair discriminator

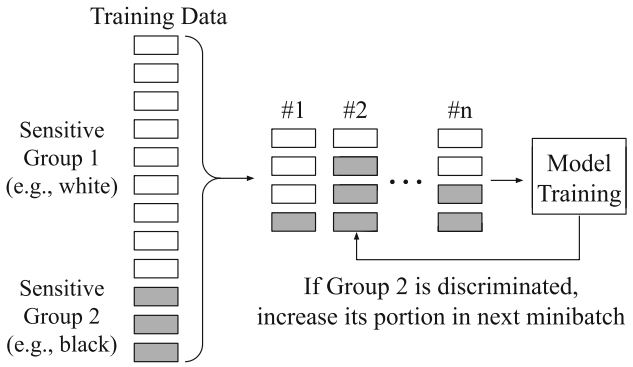

Fig. 16 FairBatch [132] is a batch selection framework for model fairness where sensitive group ratios are adjusted based on intermediate model fairness

ing by adding fairness constraints. The second is to make the model compete with a fairness discriminator via adversarial training. The third is adaptive sample reweighting techniques that re-weight input samples for fairness.

Directly adding fairness constraints to the model training objective function is an effective way to optimize for fairness. Zafar et al. propose fairness constraints techniques [177] to use in the objective function of model training to satisfy demographic parity. The focus is on convex margin classifiers like SVMs. However, as the demographic parity constraint is not convex, it cannot be directly added to the objective function. Instead, the idea is to use a proxy that approximates demographic parity and is convex. For the proxy, the authors use the covariance between the sensitive attribute and the signed distance to the decision boundary. Figure 14 provides an intuition why covariance is a good proxy. A limitation of fairness constraints is that it does not readily generalize to deep neural networks that are not convex. Other optimization techniques [4,81] for maximizing fairness and accuracy have been proposed as well.

If one does not want to modify the loss function in the model, another approach is to perform adversarial training with another model for fairness. Adversarial de-biasing [178] is a representative work in this direction. Here, the idea is to do adversarial training between a binary classifier and an adversary that tries to infer the sensitive attribute value (Fig. 15). For example, the classifier may predict recidivism while the adversary infers the gender of the person based on the classifier predictions. Suppose the fairness measure is demographic parity. A key theoretical result is that, if the adversary optimally predicts the sensitive attribute, but the classifier completely fools the adversary, it means that the model prediction is independent of the sensitive attribute. In our example, recidivism will have nothing to do with the gender. One downside of adversarial debiasing or adversarial training in general is that stability is sometimes an issue where the model training may not easily converge to a single solution.

Adversarial training can also be used to attain both fair and robust training. FR-Train [131] uses a mutual information-based approach to train a model that is both fair and robust. The classifier's fairness-accuracy tradeoff is harmed when the data is poisoned. FR-train avoids this problem by competing a classifier with two discriminators for fairness and robustness. The robustness discriminator uses a clean validation set that can be constructed using crowdsourcing techniques. We will continue discussing this work in Sect. 5.3.

A major downside of the previous methods is that the model training needs to be replaced or modified significantly, and a more convenient approach is to only re-weight the samples in order to obtain similar fairness results. FairBatch [132] is a batch selection technique with the purpose of improving fairness. During batch selection, it is common to select a random sample from the training set. Instead, the idea of FairBatch is to adjust the sensitive group ratios within each batch of examples being used for training as illustrated in Fig. 16. For example, suppose the training set is biased where a certain sensitive group has very few examples. If an inter-

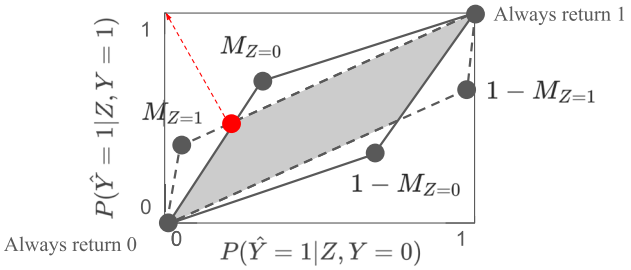

Fig. 17 Post-processing unfairness mitigation [66] involves combining models using randomization to attain the desired fairness

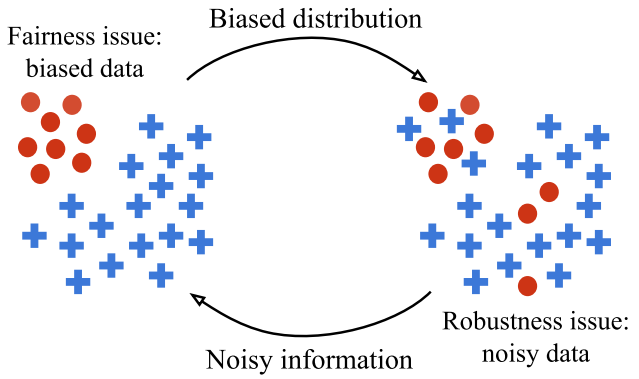

Fig. 18 Fairness and robustness issues may negatively affect each other. Noisy or missing group information may result in inaccurate results after unfairness mitigation. A biased distribution in the data may result in disproportionate accuracies after robust training

mediate model shows poor fairness, then the next batch of examples will contain more examples of that sensitive group. How exactly the sensitive group ratio should be adjusted is the technical challenge. OmniFair [179] is a declarative system for supporting group fairness for any model by reweighting samples. While the goals are similar to FairBatch, the specific optimization techniques differ where Omnifair uses a Lagrangian multiplier to translate a constraint optimization problem into an unconstrained optimization problem and leverages a monotonicity property. Other techniques include an adaptive sample reweighting approach that corrects label bias [77] and an adaptive boosting technique for maximizing fairness [76].

Post-processing mitigation The final approach for unfairness mitigation is to fix model predictions for fairness, which is the only option if the data and model cannot be modified. However, post-processing usually results in a tradeoff of worse accuracy.

We introduce a representative work [66] that combines models to adjust fairness. The method we explain here assumes equalized odds for binary classifiers, although other settings are supported in the paper as well. We assume a model M for each Z value and then construct the following models: a trivial model that only returns 0, another trivial model that only returns 1, and the "inverted" model $1 - M$, which returns the opposite prediction of M. The idea is to combine these models using randomization such that the fairness criteria is satisfied. Figure 17 illustrates how this combination can be done for $Z = 0$ and $Z = 1$. In each case, we can generate a model with the desired positive prediction rate as long as it is inside the parallelogram. If we generate a model in the intersection of the two parallelograms, we can find a model where $P(\hat{Y} = 1|Z = 0, Y = A) = P(\hat{Y} = 1|Z = 1, Y = A), A \in \{0, 1\}$, which is exactly the definition of equalized odds. Among the possible combined models, we then choose the one with the lowest expected loss (i.e., closest to the top left as highlighted in the figure). Other post-processing approaches leverage unlabeled data [37] and calibration [116].

5.3 Convergence with robustness techniques

Most recently, we are witnessing a convergence of fairness and robustness techniques. This direction is inevitable because both techniques address flaws in the data, but one does not subsume the other. Fair training assumes that the data are clean and only focus on removing its bias. However, the sensitive attribute itself can be noisy or even missing. On the other hand, robust training primarily focuses on improving the overall accuracy, but does not consider disproportionate performances between different sensitive groups. In general, fairness and robustness are not necessarily aligning goals. For example, if the data are already biased, then removing noisy data for robust training may end up worsening the bias by removing too much data from an underrepresented group [133]. Figure 18 illustrates these dynamics. There are three directions for the convergence: making fairness approaches more robust (fairness-oriented), making robust approaches fairer (robust-oriented), and equal mergers of fair and robust training. We summarize the recent research for each of the three approaches.

Fairness-oriented approaches The first direction of convergence is to make fair training more robust. This research currently has two directions: when the sensitive group information is noisy or entirely missing.

The first scenario may occur if some users may want to hide or mistakenly omit their group memberships. An analysis of fair training results on noisy sensitive group information [168] shows that the true fairness violation on a clean sensitive group can be bounded by a distance between this group and its noisy version. In addition, noise-tolerant fair training techniques [92] have been proposed where the idea is to change the unfairness tolerance to estimate the fairness of the true data distribution.

The second scenario is when the sensitive attribute is fully missing. Here, the data collection sometimes does

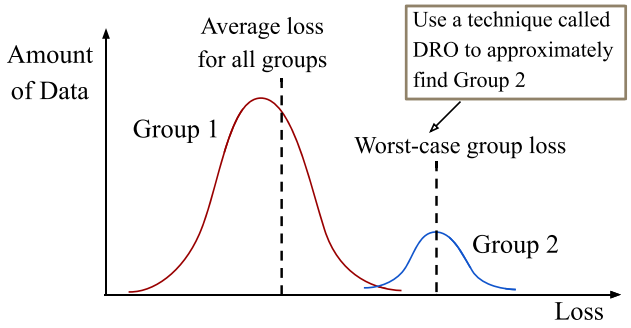

Fig. 19 A DRO-based fair algorithm [67] improves model fairness without using the sensitive group information by identifying the worst-performing samples

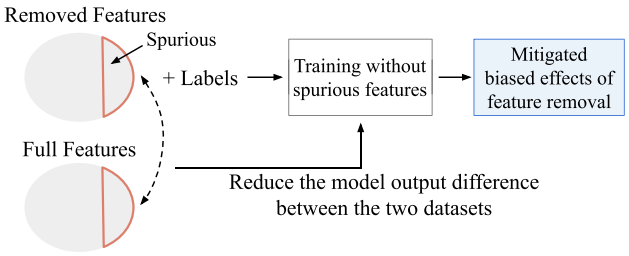

Fig. 20 A self-training technique [84] can mitigate the biased effects of spurious feature removal by also using full-featured data

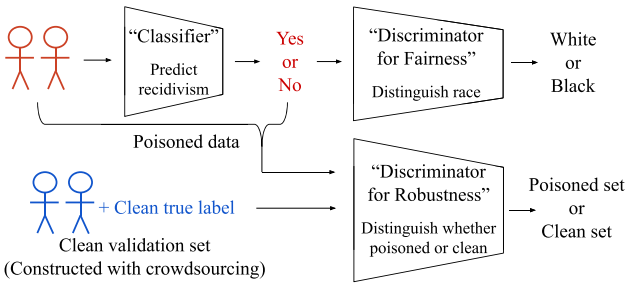

Fig. 21 FR-Train [131] is a mutual information-based approach for achieving both fairness and robustness, which competes one classifier with two discriminators

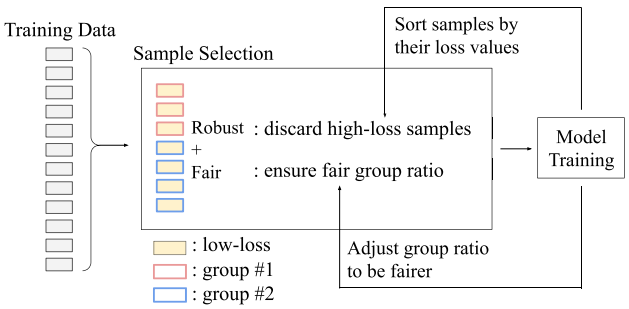

Fig. 22 Adaptive sample selection [133] can be another solution for improving fairness and robustness. The key idea is to utilize only clean and fair samples in training

not gather the group information due to various reasons like legal restrictions. Distributionally Robust Optimization (DRO) [147] has been used to improve the model performance for minority sensitive groups without using the group information [67]. The idea is to approximately minimize the worst-case (latent) group loss by identifying the worst-performing samples (Fig. 19) and giving them more weight. Adversarially reweighted learning for fairness [91] makes the assumption that unobserved sensitive attributes are correlated with the features and labels, and performs adversarial training between a classifier versus an adversary that finds less accurate clustered regions and gives more weights on those regions.

Robustness-oriented approaches Robust training is designed to improve the overall accuracy of a model, but may discriminate groups where some have much worse accuracy than others. There are three directions of research: finding anomalies in the data, training without spurious features, and improving robustness via adversarial training. Fair anomaly detection [180] has been proposed to prevent anomaly detection from discriminating specific groups. The idea is to compete a classifier that finds abnormal data and a discriminator that predicts the sensitive group from the classifier's prediction. After training, the classifier's output becomes independent of the sensitive group. Fair training without spurious features [84] addresses the problem of preventing feature removal from being discriminating. A self-training technique is proposed to mitigate accuracy degradation and biased effects (Fig. 20). Finally, fair adversarial training [172] prevents adversarial training from discriminating groups by adding constraints for equalizing accuracy and robustness.

Equal mergers Robust and fair training can be combined in equal terms as well. One direction is to make the model training fair and robust at the same time. FR-Train [131] is a mutual information-based framework that competes a classifier, discriminator for fairness, and discriminator for robustness to make the classifier fair and robust (Fig. 21). A recent sample selection framework [133] adaptively selects training samples for fair and robust model training (Fig. 22). This framework does not require modifying the model or leveraging additional clean data. A fairness-aware ERM framework [167] has been proposed based on the observation that group-dependent label noises may reduce both model accuracy and fairness. The solution is to use surrogate loss where the label distribution is corrected based on the noise rates of groups. The surrogate loss better reflects the true loss and thus mitigates the negative effects of group-dependent label noises. Another direction of robust and fair training is to take a role of an adversary and generate attacks that not only reduce accuracy, but also harm fairness. Fairness-targeted poisoning attacks [148] proposes a gradient-based attack method that finds the optimal attack locations that reduce the fairness the most.

6 Overall findings and future directions

We summarize our findings. In Sect. 2, we explained that data collection techniques consist of data acquisition, data labeling, and improving existing data and models. Some of the techniques have been studied by the data management community while others by the machine learning community. In Sect. 3, we covered key approaches in data validation, data cleaning, data sanitization, and data integration. Data validation can be performed using visualizations and schema information. Data cleaning has been heavily studied where recent techniques are more tailored to improving model accuracy. Data sanitization has the different flavor of defending against poisoning attacks. Data integration is challenging due to multimodal data. In Sect. 4, we explained that noisy or missing labels incur poor generalization on test data. Existing work for noisy labels suffers from either (i) accumulated noise or (ii) partial exploration of training data. Hybrid (e.g., SELFIE) and semi-supervised techniques (e.g., DivideMix) can achieve very high accuracy with noisy training data. Semi-supervised (e.g., MixMatch) and self-supervised (e.g., JigsawNet) techniques are actively developed to exploit abundant unlabeled data. In Sect. 5, we covered fairness measures, unfairness mitigation techniques, and convergence with robustness techniques. The mitigation can be done before, during, or after model training. Pre-processing is useful when training data can be modified. In-processing is useful when the training algorithm can be modified. Post-processing can be used when we cannot modify the data and model training. The convergence with robustness techniques can be categorized into fair-robust techniques, robust-fair techniques, and equal mergers.

As data-centric AI becomes more established, we believe there will be various convergences of these research areas. Our list is certainly not exhaustive, but we attempt to identify the major trends.

- *Data cleaning and robust training* Currently, data cleaning is becoming more machine learning oriented, but is considered less effective than robust training. We believe that the two techniques should continue integrating for the best results.
- *Data validation and model fairness* The recent works in data validation point to AI ethics as one of the challenging aspects to validate. We believe that model fairness will eventually be merged into the data validation process.
- *Data collection* So far, most of the machine learning literature assumes that the input data are already given. At the same time, data collection for accurate machine learning is now an active research direction in the data management community. We believe this trend will continue to expand where data collection needs to also consider fairness and robustness.
- *Model training and testing* Improving model training and testing protocols is becoming another solution for dealing with data quality issues. The output of the model on data samples provides useful knowledge for evaluating the data, helping to develop accurate and robust inference pipelines. We believe that the learning dynamics of models provide new perspectives for interpreting robustness and fairness.
- *Model fairness and robustness* Trustworthy AI is becoming increasingly critical in the machine learning community, and we believe its various aspects including fairness and robustness will have to be addressed together instead of one at a time. There are other elements of Trustworthy AI including privacy and explainability that should eventually be part of data-centric AI as well.

Concluding remark In the data-centric AI era, collecting data and improving its quality will only become more critical for deep learning. We covered four major topics (data collection, data cleaning, validation, and integration, robust model training, and fair model training), which have been studied by different communities, but need to be used together. We believe all the data techniques will eventually converge with the robust and fair training techniques as data-centric AI matures, and hope that our survey plays a catalyst role.

Acknowledgements This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2022-0-00157 and 2020-0-00862) and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. NRF-2018R1A5A1059921 and NRF-2022R1A2C2004382).

References

1. Amazon Mechanical Turk. https://www.mturk.com/. Accessed 13 July 2022
2. Amazon SageMaker Ground Truth. https://aws.amazon.com/sagemaker/groundtruth/. Accessed 13 July 2022
3. Amazon scraps secret AI recruiting tool that showed bias against women. https://www.reuters.com/article/us-amazon-com-jobs-automation-insight-idUSKCN1MK08G. Accessed 13 July 2022
4. Agarwal, A., Beygelzimer, A., Dudík, M., Langford, J., Wallach., H.M.: A reductions approach to fair classification. In: ICML, pp. 60–69 (2018)
5. Agrawal, P., Arya, R., Bindal, A., Bhatia, S., Gagneja, Godlewski, J., Low, Y., Muss, T., Paliwal, M.M., Raman, S., Shah, V., Shen, Sugden, L., Zhao, K., Wu, M.-C.: Data platform for machine learning. In: SIGMOD, pp. 1803–1816 (2019)
6. Amershi, S., Begel, A., Bird, C., DeLine, R., Gall, H.C., Kamar, E., Nagappan, N., Nushi, B., Zimmermann, T.: Software engineering for machine learning: a case study. In: ICSE, pp. 291–300 (2019)

7. Angwin, J., Larson, J., Mattu, S., Kirchner, L.: Machine bias: there's software used across the country to predict future criminals. And its biased against blacks (2016)
8. Anwar, S., Barnes, N.: Real image denoising with feature attention. In: CVPR, pp. 3155–3164 (2019)
9. Asudeh, A., Jin, Z., Jagadish, H.V.: Assessing and remedying coverage for a given dataset. In: ICDE, pp. 554–565 (2019)
10. Bach, S.H., Rodriguez, D., Liu, Y., Luo, C., Shao, H., Xia, C., Sen, S., Ratner, A., Hancock, B., Alborzi, H., Kuchhal, R., Ré, C.,Malkin, R.: Snorkel Drybell: a case study in deploying weak supervision at industrial scale. In: SIGMOD, pp. 362–375 (2019)
11. Baltrusaitis, T., Ahuja, C., Morency, L.-P.: Multimodal machine learning: a survey and taxonomy. IEEE Trans. Pattern Anal. Mach. Intell. **41**(2), 423–443 (2019)
12. Barocas, S., Hardt, M., Narayanan, A.: Fairness and machine learning. fairmlbook.org. http://www.fairmlbook.org (2019)
13. Baylor, D., Breck, E., Cheng, H.-T., Fiedel, N., Foo, C.Y., Haque, Z., Haykal, S., Ispir, M., Jain, V., Koc, L., Koo, C.Y., Lew, L., Mewald, C., Modi, A.N., Polyzotis, N., Ramesh, S., Roy, S., Whang, S.E., Wicke, M., Wilkiewicz, J., Zhang, X., Zinkevich, M.: TFX: a tensorflow-based production-scale machine learning platform. In: KDD, pp. 1387–1395 (2017)
14. Bellamy, R.K.E., Dey, K., Hind, M., et al.: AI fairness 360: an extensible toolkit for detecting and mitigating algorithmic bias. IBM J. Res. Dev. **63**, 4:1-4:15 (2019)
15. Berk, R., Heidari, H., Jabbari, S., Kearns, M., Roth, A.: Fairness in criminal justice risk assessments: the state of the art (2017)
16. Berthelot, D., Carlini, N., Cubuk, E.D., Kurakin, A., Sohn, K., Zhang, H., Raffel, C.: Remixmatch: semi-supervised learning with distribution matching and augmentation anchoring. In: ICLR (2020)
17. Berthelot, D., Carlini, N., Goodfellow, I.J., Papernot, N., Oliver, A., Raffel, C.: Mixmatch: a holistic approach to semi-supervised learning. In: NeurIPS, pp. 5050–5060 (2019)
18. Biessmann, F., Golebiowski, J., Rukat, T., Lange, D., Schmidt, P.: Automated data validation in machine learning systems. IEEE Data Eng. Bull. **44**(1), 51–65 (2021)
19. Biggio, B., Corona, I., Maiorca, D., Nelson, B., Srndic, N., Laskov, P., Giacinto, G., Roli, F.: Evasion attacks against machine learning at test time. In: ECML PKDD, pp. 387–402. Springer (2013)
20. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: COLT, pp. 92–100. ACM, New York (1998)
21. Boehm, M., Antonov, I., Baunsgaard, S., Dokter, M., Ginthör, R., Innerebner, K., Klezin, F., Lindstaedt, S.N., Phani, A., Rath, B., Reinwald, B., Siddiqui, S., Wrede, S.B.: Systemds: a declarative machine learning system for the end-to-end data science lifecycle. In: CIDR (2020)
22. Breck, E., Zinkevich, M., Polyzotis, N., Whang, S., Roy, S.: Data validation for machine learning. In: MLSys (2019)
23. Brickley, D., Burgess, M., Noy, N.F.: Google dataset search: building a search engine for datasets in an open web ecosystem. In: WWW, pp. 1365–1375 (2019)
24. CrowdFlower Data Science Report. https://visit.figure-eight.com/rs/416-ZBE-142/images/CrowdFlower_DataScienceReport_2016.pdf
25. Cafarella, M.J., Halevy, A.Y., Lee, H., Madhavan, J., Cong, Y., Wang, D.Z., Wu, E.: Ten years of webtables. PVLDB **11**(12), 2140–2149 (2018)
26. Cambronero, J., Feser, J.K., Smith, M.J., Madden, S.: Query optimization for dynamic imputation. Proc. VLDB Endow. **10**(11), 1310–1321 (2017)
27. Chakraborty, A., Alam, M., Dey, V., Chattopadhyay, A., Mukhopadhyay, D.: Adversarial attacks and defences: a survey. CoRR arXiv:1810.00069 (2018)
28. Chang, H.-S., Learned-Miller, E.G., McCallum., A.: Active bias: training more accurate neural networks by emphasizing high variance samples. In: NeurIPS, pp. 1002–1012 (2017)
29. Che, Z., Purushotham, S., Cho, K., Sontag, D., Liu, Y.: Recurrent neural networks for multivariate time series with missing values. Nat. Sci. Rep. **8**(1), 6085 (2018)
30. Chen, A., Chow, A., Davidson, A., DCunha, A., Ghodsi, A., Hong, S.A., Konwinski, A., Mewald, C., Murching, S., Nykodym, T., Ogilvie, P., Parkhe, M., Singh, A., Xie, F., Zaharia, M., Zang, R., Zheng, J., Zumar, C.: Developments in mlflow: a system to accelerate the machine learning lifecycle. In: DEEM@SIGMOD, pp. 5:1–5:4 (2020)
31. Chen, I.Y., Johansson, F.D., Sontag, D.A.: Why is my classifier discriminatory? In: NeurIPS, pp. 3543–3554 (2018)
32. Chen, T., Guestrin, C.: Xgboost: a scalable tree boosting system. In: KDD, pp. 785–794 (2016)
33. Cheng, Y., Diakonikolas, I., Ge, R.: High-dimensional robust mean estimation in nearly-linear time. In: SIAM, pp. 2755–2771 (2019)
34. Choi, K., Grover, A., Singh, T., Shu, R., Ermon, S.: Fair generative modeling via weak supervision. In: ICML, pp. 1887–1898 (2020)
35. Chouldechova, A.: Fair prediction with disparate impact: a study of bias in recidivism prediction instruments. Big Data **5**(2), 153–163 (2017)
36. Chouldechova, A., Roth, A.: A snapshot of the frontiers of fairness in machine learning. Commun. ACM **63**(5), 82–89 (2020)
37. Chzhen, E., Denis, C., Hebiri, M., Oneto, L., Pontil, M.: Leveraging labeled and unlabeled data for consistent fair binary classification. In: NeurIPS, pp. 12739–12750 (2019)
38. Cotter, A., Jiang, H., Sridharan, K.: Two-player games for efficient non-convex constrained optimization. In: ALT, pp. 300–332 (2019)
39. Cretu, G.F., Stavrou, A., Locasto, M.E., Stolfo, S.J., Keromytis, A.D.: Casting out demons: sanitizing training data for anomaly sensors. In: IEEE S&P, pp. 81–95 (2008)
40. Cubuk, E.D., Zoph, B., Mané, D., Vasudevan, V., Le, Q.V.: Autoaugment: learning augmentation strategies from data. In: CVPR, pp. 113–123 (2019)
41. Data age 2025. https://www.seagate.com/our-story/data-age-2025/
42. Data-centric AI resource hub. https://datacentricai.org/
43. Data prep still dominates data scientists' time, survey finds. https://www.datanami.com/2020/07/06/data-prep-still-dominates-data-scientists-time-survey-finds/
44. Diakonikolas, I., Kamath, G., Kane, D., Li, J., Moitra, A., Stewart, A.: Robust estimators in high-dimensions without the computational intractability. SIAM J. Comput. **48**(2), 742–864 (2019)
45. Dieterich, W., Mendoza, C., Brennan, T.: Compas risk scales: demonstrating accuracy equity and predictive parity. Technical report, Northpoint Inc (2016)
46. Doan, A., Halevy, A.Y., Ives, Z.G.: Principles of Data Integration. Morgan Kaufmann, Burlington (2012)
47. Dolatshah, M., Teoh, M., Wang, J., Pei, J.: Cleaning crowdsourced labels using oracles for statistical classification. PVLDB **12**(4), 376–389 (2018)
48. Dong, X.L., Rekatsinas, T.: Data integration and machine learning: a natural synergy. In: KDD, pp. 3193–3194 (2019)
49. Dreves, M., Huang, G., Peng, Z., Polyzotis, N., Rosen, E., Paul Suganthan, G.C.: Validating data and models in continuous ML pipelines. IEEE Data Eng. Bull. **44**(1), 42–50 (2021)
50. Dua, D., Graff, C.: UCI machine learning repository (2017)
51. Dwork, C., Hardt, M., Pitassi, T., Reingold, O., Zemel, R.S.: Fairness through awareness. In: ITCS, pp. 214–226 (2012)
52. Facets—visualization for ML datasets. https://pair-code.github.io/facets/. Accessed 13 July 2022

53. Feldman, M., Friedler, S.A., Moeller, J., Scheidegger, C., Venkatasubramanian, S.: Certifying and removing disparate impact. In: KDD, pp. 259–268 (2015)
54. Fernandez, R.C., Abedjan, Z., Koko, F., Yuan, G., Madden, S., Stonebraker, M.: Aurum: a data discovery system. In: ICDE, pp. 1001–1012 (2018)
55. Foster, D.P., Stine, R.A.: Alpha-investing: a procedure for sequential control of expected false discoveries. J. R. Stat. Soc. Ser. B (Stat. Methodol.) **70**(2), 429–444 (2008)
56. GCP AI platform data labeling service. https://cloud.google.com/ai-platform/data-labeling/docs. Accessed 13 July 2022
57. Google apologises for Photos app's racist blunder. https://www.bbc.com/news/technology-33347866. Accessed 13 July 2022
58. Goel, K., Albert, G., Li, Y., Ré, C.: Model patching: closing the subgroup performance gap with data augmentation. In: ICLR (2021)
59. Goodfellow, I.J.: NIPS 2016 tutorial: generative adversarial networks. CoRR arXiv:1701.00160 (2017)
60. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A.C., Bengio, Y.: Generative adversarial nets. In: NIPS, pp. 2672–2680 (2014)
61. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. In: ICLR (2015)
62. Gordon, J.: Introducing tensorflow hub: a library for reusable machine learning modules in tensorflow (2018)
63. Grafberger, S., Stoyanovich, J., Schelter, S.: Lightweight inspection of data preprocessing in native machine learning pipelines. In: CIDR (2021)
64. Halevy, A.Y., Korn, F., Noy, N.F., Olston, C., Polyzotis, N., Roy, S., Whang, S.E.: Goods: organizing Google's datasets. In: SIGMOD, pp. 795–806 (2016)
65. Han, B., Yao, Q., Yu, X., Niu, G., Xu, M., Hu, W., Tsang, I.W., M. Sugiyama. Co-teaching: robust training of deep neural networks with extremely noisy labels. In: NeurIPS, pp. 8536–8546 (2018)
66. Hardt, M., Price, E., Srebro, N.: Equality of opportunity in supervised learning. In: NIPS, pp. 3315–3323 (2016)
67. Hashimoto, T.B., Srivastava, M., Namkoong, H., Liang, P.: Fairness without demographics in repeated loss minimization. In: Dy, J.G., Krause, A. (eds.) ICML, vol. 80, pp. 1934–1943. PMLR (2018)
68. Hazelwood, K.M., Bird, S., Brooks, D.M., Chintala, S., Diril, U., Dzhulgakov, D., Fawzy, M., Jia, B., Jia, Y., Kalro, A., Law, J., Lee, K., Lu, J., Noordhuis, P., Smelyanskiy, M., Xiong, L., Wang, X.: Applied machine learning at Facebook: a datacenter infrastructure perspective. In: HPCA, pp. 620–629 (2018)
69. Hendrycks, D., Mu, N., Cubuk, E.D., Zoph, B., Gilmer, J., Lakshminarayanan, B.: Augmix: a simple data processing method to improve robustness and uncertainty. In: ICLR (2020)
70. Heo, G., Roh, Y., Hwang, S., Lee, D., Whang, S.E.: Inspector gadget: a data programming-based labeling system for industrial images. In: PVLDB (2021)
71. Hermann, J.M., Baso, D.: Meet michelangelo: Uber's machine learning platform (2017)
72. Hodge, V.J., Austin, J.: A survey of outlier detection methodologies. Artif. Intell. Rev. **22**(2), 85–126 (2004)
73. Huber, P.J.: Robust estimation of a location parameter. In: Kotz, S., Johnson, N.L. (eds.) Breakthroughs in Statistics, pp. 492–518. Springer, Berlin (1992)
74. Ilyas, I.F., Chu, X.: Data Cleaning. ACM, New York (2019)
75. Ilyas, I.F., Rekatsinas, T.: Machine learning and data cleaning: Which serves the other? J. Data Inf. Qual. (2021). Just Accepted
76. Iosifidis, V., Ntoutsi, E.: Adafair: cumulative fairness adaptive boosting. In: CIKM, pp. 781–790 (2019)
77. Jiang, H., Nachum, O.: Identifying and correcting label bias in machine learning. In: AISTATS, pp. 702–712 (2020)
78. Jiang, L., Zhou, Z., Leung, T., Li, L.-J., Fei-Fei, L.: Mentornet: learning data-driven curriculum for very deep neural networks on corrupted labels. In: ICML, pp. 2309–2318 (2018)
79. Kaggle. https://www.kaggle.com
80. Kamiran, F., Calders, T.: Data preprocessing techniques for classification without discrimination. Knowl. Inf. Syst. **33**(1), 1–33 (2011)
81. Kamishima, T., Akaho, S., Asoh, H., Sakuma, J.: Fairness-aware classifier with prejudice remover regularizer. In: ECML PKDD, pp. 35–50 (2012)
82. Karlas, B., Li, P., Wu, R., Gürel, N.M., Chu, X., Wu, W., Zhang, C.: Nearest neighbor classifiers over incomplete information: from certain answers to certain predictions. Proc. VLDB Endow. **14**(3), 255–267 (2020)
83. Khademi, A., Lee, S., Foley, D., Honavar, V.: Fairness in algorithmic decision making: an excursion through the lens of causality. In: WWW, pp. 2907–2914 (2019)
84. Khani, F., Liang, P.: Removing spurious features can hurt accuracy and affect groups disproportionately. In: FAccT, pp. 196–205. ACM (2021)
85. Kilbertus, N., Rojas-Carulla, M., Parascandolo, G., Hardt, M., Janzing, D., Schölkopf, B.: Avoiding discrimination through causal reasoning. In: NeurIPS, pp. 656–666 (2017)
86. Kim, H., Lee, K., Hwang, G., Suh, C.: Crash to not crash: learn to identify dangerous vehicles using a simulator. In: AAAI, pp. 978–985 (2019)
87. Koh, P.W., Steinhardt, J., Liang, P.: Stronger data poisoning attacks break data sanitization defenses. CoRR arXiv:1811.00741 (2018)
88. Krishnan, S., Wang, J., Eugene, W., Franklin, M.J., Goldberg, K.: Activeclean: interactive data cleaning for statistical modeling. PVLDB **9**(12), 948–959 (2016)
89. Kurach, K., Lucic, M., Zhai, X., Michalski, M., Gelly, S.: The GAN landscape: losses, architectures, regularization, and normalization. CoRR arXiv:1807.04720 (2018)
90. Kusner, M.J., Loftus, J., Russell, C., Silva, R.: Counterfactual fairness. In: NeurIPS, pp. 4066–4076 (2017)
91. Lahoti, P., Beutel, A., Chen, J., Lee, K., Prost, F., Thain, N., Wang, X., Chi, E.: Fairness without demographics through adversarially reweighted learning. In: NeurIPS (2020)
92. Lamy, A.L., Zhong, Z.: Noise-tolerant fair classification. In: Wallach, H.M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E.B., Garnett, R. (eds.) NeurIPS, pp. 294–305 (2019)
93. Lee, D.J.L., Parameswaran, A.G.: The case for a visual discovery assistant: a holistic solution for accelerating visual data exploration. IEEE Data Eng. Bull. **41**(3), 3–14 (2018)
94. Lee, J.-G., Roh, Y., Song, H., Whang, S.E.: Machine learning robustness, fairness, and their convergence. In: KDD, pp. 4046–4047 (2021)
95. Li, J., Socher, R., Hoi, S.C.H.: Dividemix: learning with noisy labels as semi-supervised learning. In: ICLR (2020)
96. Li, P., Rao, X., Blase, J., Zhang, Y., Chu, X., Zhang, C.: CleanML: a benchmark for joint data cleaning and machine learning [experiments and analysis]. In: ICDE (2021)
97. Liu, Z., Park, J.H., Rekatsinas, T., Tzamos, C.: On robust mean estimation under coordinate-level corruption. In: ICML, pp. 6914–6924. PMLR (2021)
98. Liu, Z., Park, J., Rekatsinas, T., Tzamos, C.: On robust mean estimation under coordinate-level corruption. In: ICML, pp. 6914–6924 (2021)
99. Malach, E., Shalev-Shwartz, S.: Decoupling "when to update" from "how to update". In: NIPS, pp. 960–970 (2017)
100. Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., Galstyan, A.: A survey on bias and fairness in machine learning. CoRR arXiv:1908.09635 (2019)

101. Melgar, L.A., Dao, D., Gan, S., Gürel, N.M., Hollenstein, N., Jiang, J., Karlas, B., Lemmin, T., Li, T., Li, Y., Rao, X., Rausch, J., Renggli, C., Rimanic, L., Weber, M., Zhang, S., Zhao, Z., Schawinski, K., Wu, W., Zhang, C.: Ease.ml: a lifecycle management system for machine learning. In: CIDR (2021)
102. Meng, D., Chen, H.: Magnet: a two-pronged defense against adversarial examples. In: Thuraisingham, B.M., Evans, D., Malkin, T., Xu, D. (eds.) ACM SIGSAC, pp. 135–147 (2017)
103. Metzen, J.H., Genewein, T., Fischer, V., Bischoff, B.: On detecting adversarial perturbations. In: ICLR (2017)
104. Miller, R.J., Nargesian, F., Zhu, E., Christodoulakis, C., Pu, K.Q., Andritsos, P.: Making open data transparent: data discovery on open data. IEEE Data Eng. Bull. **41**(2), 59–70 (2018)
105. Mintz, M., Bills, S., Snow, R., Jurafsky, D.: Distant supervision for relation extraction without labeled data. In: Su, K.-Y., Su, J., Wiebe, J. (eds.) ACL, pp. 1003–1011 (2009)
106. Nabi, R., Shpitser, I.: Fair inference on outcomes. In: AAAI, pp. 1931–1940 (2018)
107. Neutatz, F., Chen, B., Abedjan, Z., Eugene, W.: From cleaning before ML to cleaning for ML. IEEE Data Eng. Bull. **44**(1), 24–41 (2021)
108. Noroozi, M., Favaro, P.: Unsupervised learning of visual representations by solving jigsaw puzzles. In: ECCV, pp. 69–84 (2016)
109. Principles for AI ethics. https://research.samsung.com/artificial-intelligence. Accessed 13 July 2022
110. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE TKDE **22**(10), 1345–1359 (2010)
111. Papernot, N., McDaniel, P.D., Wu, X., Jha, S., Swami, A.: Distillation as a defense to adversarial perturbations against deep neural networks. In: IEEE SP, pp. 582–597 (2016)
112. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: an imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d' Alché-Buc, F., Fox, E., Garnett, R. (eds.) NeurIPS, pp. 8024–8035. Curran Associates, Inc. (2019)
113. Patrini, G., Rozza, A., Menon, A.K., Nock, R., Qu, L.: Making deep neural networks robust to label noise: a loss correction approach. In: CVPR, pp. 2233–2241 (2017)
114. Paudice, A., Muñoz-González, L., György, A., Lupu, E.C.: Detection of adversarial training examples in poisoning attacks through anomaly detection. CoRR arXiv:1802.03041 (2018)
115. Pelekis, N., Ntrigkogias, C., Tampakis, P., Sideridis, S., Theodoridis, Y.: Hermoupolis: a trajectory generator for simulating generalized mobility patterns. In: ECML PKDD, pp. 659–662 (2013)
116. Pleiss, G., Raghavan, M., Wu, F., Kleinberg, J.M., Weinberger, K.Q.: On fairness and calibration. In: NIPS, pp. 5680–5689 (2017)
117. Polyzotis, N., Roy, S., Whang, S.E., Zinkevich, M.: Data management challenges in production machine learning. In: SIGMOD, pp. 1723–1726 (2017)
118. Polyzotis, N., Roy, S., Whang, S.E., Zinkevich, M.: Data lifecycle challenges in production machine learning: a survey. SIGMOD Rec. **47**(2), 17–28 (2018)
119. Qayyum, A., Qadir, J., Bilal, M., Al-Fuqaha, A.: Secure and robust machine learning for healthcare: a survey. IEEE Rev. Biomed. Eng. **14**, 156–180 (2020)
120. Responsible AI practices. https://ai.google/responsibilities/responsible-ai-practices. Accessed 13 July 2022
121. Responsible AI principles from Microsoft. https://www.microsoft.com/en-us/ai/responsible-ai. Accessed 13 July 2022
122. Ratner, A., Bach, S.H., Ehrenberg, H., Fries, J., Sen, W., Ré, C.: Snorkel: rapid training data creation with weak supervision. PVLDB **11**(3), 269–282 (2017)
123. Ratner, A., Bach, S.H., Ehrenberg, H.R., Fries, J.A., Sen, W., Ré, C.: Snorkel: rapid training data creation with weak supervision. VLDB J. **29**(2–3), 709–730 (2020)
124. Ratner, A.J., Ehrenberg, H.R., Hussain, Z., Dunnmon, J., Ré, C.: Learning to compose domain-specific transformations for data augmentation. In: NIPS, pp. 3239–3249 (2017)
125. Redyuk, S., Kaoudi, Z., Markl, V., Schelter, S.: Automating data quality validation for dynamic data ingestion. In: EDBT, pp. 61–72 (2021)
126. Reed, S.E., Lee, H., Anguelov, D., Szegedy, C., Erhan, D., Rabinovich, A.: Training deep neural networks on noisy labels with bootstrapping. In: ICLR (2015)
127. Rekatsinas, T., Chu, X., Ilyas, I.F., Ré, C.: Holoclean: holistic data repairs with probabilistic inference. PVLDB **10**(11), 1190–1201 (2017)
128. Renggli, C., Rimanic, L., Gürel, N.M., Karlas, B., Wu, W., Zhang, C.: A data quality-driven view of mlops. IEEE Data Eng. Bull. **44**(1), 11–23 (2021)
129. Ricci, F., Rokach, L., Shapira, B. (eds.): Recommender Systems Handbook. Springer, Berlin (2015)
130. Roh, Y., Heo, G., Whang, S.E.: A survey on data collection for machine learning: a big data—AI integration perspective. In: IEEE TKDE (2019)
131. Roh, Y., Lee, K., Whang, S.E., Suh, C.: FR-Train: a mutual information-based approach to fair and robust training. In: ICML (2020)
132. Roh, Y., Lee, K., Whang, S.E., Suh, C.: Fairbatch: batch selection for model fairness. In: ICLR. OpenReview.net (2021)
133. Roh, Y., Lee, K., Whang, S.E., Suh, C.: Sample selection for fair and robust training. In: NeurIPS (2021)
134. Software 2.0. https://medium.com/@karpathy/software-2-0-a64152b37c35
135. South Korean AI chatbot pulled from Facebook after hate speech towards minorities. https://www.theguardian.com/world/2021/jan/14/time-to-properly-socialise-hate-speech-ai-chatbot-pulled-from-facebook. Accessed 13 July 2022
136. Salimi, B., Rodriguez, L., Howe, B., Suciu, D.: Interventional fairness: causal database repair for algorithmic fairness. In: SIGMOD, pp. 793–810 (2019)
137. Schelter, S., Böse, J.-H., Kirschnick, J., Klein, T., Seufert, S.: Automatically tracking metadata and provenance of machine learning experiments. In: Workshop on ML Systems at NIPS (2017)
138. Schelter, S., Grafberger, S., Schmidt, P., Rukat, T., Kießling, M., Taptunov, A., Bießmann, F., Lange, D.: Differential data quality verification on partitioned data. In: ICDE, pp. 1940–1945 (2019)
139. Schelter, S., Lange, D., Schmidt, P., Celikel, M., Bießmann, F., Grafberger, A.: Automating large-scale data quality verification. Proc. VLDB Endow. **11**(12), 1781–1794 (2018)
140. Schelter, S., Rukat, T., Biessmann, F.: JENGA: a framework to study the impact of data errors on the predictions of machine learning models. In: EDBT, pp. 529–534 (2021)
141. Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J.-F., Dennison, D.: Hidden technical debt in machine learning systems. In: NIPS, pp. 2503–2511 (2015)
142. Settles, B.: Active learning. In: Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers (2012)
143. Shafahi, A., Huang, W.R., Najibi, M., Suciu, O., Studer, C., Dumitras, T., Goldstein, T.: Poison frogs! Targeted clean-label poisoning attacks on neural networks. In: NeurIPS, pp. 6106–6116 (2018)
144. Shang, L.: Denoising natural images based on a modified sparse coding algorithm. Appl. Math. Comput. **205**(2), 883–889 (2008)

145. Shen, Z., Liu, J., He, Y., Zhang, X., Xu, R., Yu, H., Cui, P.: Towards out-of-distribution generalization: a survey. arXiv:2108.13624 (2021)
146. Sheng, V.S., Provost, F.J., Ipeirotis, P.G.: Get another label? improving data quality and data mining using multiple, noisy labelers. In: KDD, pp. 614–622 (2008)
147. Sinha, A., Namkoong, H., Duchi, J.C.: Certifying some distributional robustness with principled adversarial training. In: ICLR (2018)
148. Solans, D., Biggio, B., Castillo, C.: Poisoning attacks on algorithmic fairness. In: Hutter, F., Kersting, K., Lijffijt, J., Valera, I. (eds.) ECML PKDD, vol. 12457, pp. 162–177. Springer (2020)
149. Song, H., Kim, M., Lee, J.-G.: SELFIE: refurbishing unclean samples for robust deep learning. In: ICML, pp. 5907–5915 (2019)
150. Song, H., Kim, M., Park, D., Lee, J.-G.: Learning from noisy labels with deep neural networks: a survey. CoRR arXiv:2007.08199 (2020)
151. Song, H., Kim, M., Park, D., Shin, Y., Lee, J.-G.: Robust learning by self-transition for handling noisy labels. In: KDD, pp. 1490–1500 (2021)
152. Stonebraker, M., Ilyas, I.F.: Data integration: the current status and the way forward. IEEE Data Eng. Bull. **41**(2), 3–9 (2018)
153. Stonebraker, M., Rezig, E.K.: Machine learning and big data: what is important? IEEE Data Eng. Bull. **42**, 3–7 (2019)
154. Trusting AI. https://www.research.ibm.com/artificial-intelligence/trusted-ai/. Accessed 13 July 2022
155. Tae, K.H., Whang, S.E.: Slice tuner: a selective data acquisition framework for accurate and fair machine learning models. In: SIGMOD, pp. 1771–1783. ACM (2021)
156. Tarvainen, A., Valpola, H.: Mean teachers are better role models: weight-averaged consistency targets improve semi-supervised deep learning results. In: NIPS, pp. 1195–1204 (2017)
157. Terrizzano, I.G., Schwarz, P.M., Roth, M., Colino, J.E.: Data wrangling: the challenging Yourney from the wild to the lake. In: CIDR (2015)
158. Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., Abbeel, P.: Domain randomization for transferring deep neural networks from simulation to the real world. In: IROS, pp. 23–30 (2017)
159. Tremblay, J., Prakash, A., Acuna, D., Brophy, M., Jampani, V., Anil, C., To, T., Cameracci, E., Boochoon, S., Birchfield, S.: Training deep networks with synthetic data: bridging the reality gap by domain randomization. In: CVPR Workshops, pp. 969–977 (2018)
160. Triguero, I., García, S., Herrera, F.: Self-labeled techniques for semi-supervised learning: taxonomy, software and empirical study. Knowl. Inf. Syst. **42**(2), 245–284 (2015)
161. Tukey, J.W.: A survey of sampling from contaminated distributions. In: Contributions to Probability and Statistics, pp. 448–485 (1960)
162. van Buuren, S., Groothuis-Oudshoorn, K.: mice: multivariate imputation by chained equations in r. J. Stat. Softw. **45**(3), 1–67 (2011)
163. Varma, P., Ré, C.: Snuba: automating weak supervision to label training data. Proc. VLDB Endow. **12**(3), 223–236 (2018)
164. Vartak, M., Rahman, S., Madden, S., Parameswaran, A.G., Polyzotis, N.: SEEDB: efficient data-driven visualization recommendations to support visual analytics. PVLDB **8**(13), 2182–2193 (2015)
165. Venkatasubramanian, S.: Algorithmic fairness: measures, methods and representations. In: PODS, p. 481 (2019)
166. Wang, H., Liu, B., Li, C., Yang, Y., Li, T.: Learning with noisy labels for sentence-level sentiment classification. In: EMNLP (2019)
167. Wang, J., Liu, Y., Levy, C.: Fair classification with group-dependent label noise. In: Elish, M.C., Isaac, W., Zemel, R.S. (eds.) FAccT, pp. 526–536. ACM (2021)
168. Wang, S., Guo, W., Narasimhan, H., Cotter, A., Gupta, M.R., Jordan, M.I.: Robust optimization for fairness with noisy protected groups. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M.-F., Lin, H.-T. (eds.) NeurIPS (2020)
169. Whang, S.E., Lee, J.-G.: Data collection and quality challenges for deep learning. Proc. VLDB Endow. **13**(12), 3429–3432 (2020)
170. Xin, D., Petersohn, D., Tang, D., Yifan, W., Gonzalez, J.E., Hellerstein, J.M., Joseph, A.D., Parameswaran, A.G.: Enhancing the interactivity of dataframe queries by leveraging think time. IEEE Data Eng. Bull. **44**(1), 66–78 (2021)
171. Xu, D., Yuan, S., Zhang, L., Wu, X.: Fairgan: fairness-aware generative adversarial networks. In: IEEE Big Data, pp. 570–575 (2018)
172. Xu, H., Liu, X., Li, Y., Jain, A.K., Tang, J.: To be robust or to be fair: towards fairness in adversarial training. In: Meila, M., Zhang, T. (eds.) ICML, vol. 139, pp. 11492–11501. PMLR (2021)
173. Xu, W., Evans, D., Qi, Y.: Feature squeezing: detecting adversarial examples in deep neural networks. In: NDSS (2018)
174. Yarowsky, D.: Unsupervised word sense disambiguation rivaling supervised methods. In: ACL, pp. 189–196, Stroudsburg, PA, USA (1995). Association for Computational Linguistics
175. Yun, S., Han, D., Chun, S., Oh, S.J., Yoo, Y., Choe, J.: Cutmix: regularization strategy to train strong classifiers with localizable features. In: ICCV, pp. 6022–6031 (2019)
176. Zafar, M.B., Valera, I., Gomez-Rodriguez, M., Gummadi, K.P.: Fairness beyond disparate treatment & disparate impact: learning classification without disparate mistreatment. In: WWW, pp. 1171–1180. ACM (2017)
177. Zafar, M.B., Valera, I., Gomez-Rodriguez, M., Gummadi, K.P.: Fairness constraints: mechanisms for fair classification. In: AISTATS, pp. 962–970 (2017)
178. Zhang, B.H., Lemoine, B., Mitchell, M.: Mitigating unwanted biases with adversarial learning. In: AIES, pp. 335–340 (2018)
179. Zhang, H., Chu, X., Asudeh, A., Navathe, S.B.: Omnifair: a declarative system for model-agnostic group fairness in machine learning. In: SIGMOD, pp. 2076–2088 (2021)
180. Zhang, H., Davidson, I.: Facct. pp. 138–148. ACM (2021)
181. Zhang, H., Cissé, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: beyond empirical risk minimization. In: ICLR (2018)
182. Zhang, J., Bareinboim, E.: Fairness in decision-making: the causal explanation formula. In: AAAI (2018)
183. Zhang, Y., Ives, Z.G.: Finding related tables in data lakes for interactive data science. In: SIGMOD, pp. 1951–1966 (2020)
184. Zhao, Z., De Stefani, L., Zgraggen, E., Binnig, C., Upfal, E., Kraska, T.: Controlling false discoveries during interactive data exploration. In: SIGMOD, pp. 527–540 (2017)
185. Zhou, Y., Goldman, S.A.: Democratic co-learning. In: IEEE ICTAI, pp. 594–602 (2004)
186. Zhou, Z.-H., Li, M.: Tri-training: exploiting unlabeled data using three classifiers. IEEE TKDE **17**(11), 1529–1541 (2005)
187. Zhu, C., Ronny Huang, W., Li, H., Taylor, G., Studer, C., Goldstein, T.: Transferable clean-label poisoning attacks on deep neural nets. In: ICML, pp. 7614–7623 (2019)
188. Zhu, X.: Semi-supervised learning literature survey. Technical report, Computer Sciences, University of Wisconsin-Madison (2005)