Automated Data Validation in Machine Learning Systems

Felix Biessmann, Jacek Golebiowski, Tammo Rukat, Dustin Lange, Philipp Schmidt

{biessman,jacekgo,tammruka,langed,phschmid}@amazon.com

Amazon

Abstract

Machine Learning (ML) algorithms are a standard component of modern software systems. The validation of data ingested and produced by ML components has become a central challenge in the deployment and maintenance of ML systems. Subtle changes in the input data can result in unpredictable behavior of an ML algorithm that can lead to unreliable or unfair ML predictions. Responsible usage of ML components thus requires well calibrated and scalable data validation systems. Here, we highlight some challenges associated with data validation in ML systems. We review some of the solutions developed to validate data at the various stages of a data pipeline in modern ML systems, discuss their strengths and weaknesses and assess to what extent these solutions are being used in practice. The research reviewed indicates that the increasing need for data validation in ML systems has driven enormous progress in an emerging community of ML and Data Base Management Systems (DBMS) researchers. While this research has led to a number of technical solutions we find that many ML systems deployed in industrial applications are not leveraging the full potential of data validation in practice. The reasons for this are not only technical challenges, but there are also cultural, ethical and legal aspects that need to be taken into account when building data validation solutions for ML systems. We identify the lack of automation in data validation as one of the key factors slowing down adoption of validation solutions and translation of research into useful and robust ML applications. We conclude with an outlook on research directions at the intersection of ML and DBMS research to improve the development, deployment and maintenance of ML systems.

1 Introduction

Machine Learning (ML) technology has become a standard component in modern software systems. Many decisions are increasingly being automated with ML and the predictions of ML models are being exposed in data products or consumed by other downstream software components. This trend gives rise to new research challenges at the intersection between Data Base Management Systems (DBMS) community and the ML community. Many of these challenges are related to data validation. In contrast to standard software systems, for which a large arsenal of testing concepts and utilities exists, testing of ML systems is difficult. Depending on the ingested data, ML systems can behave very different, and often subtle changes in the input data, that are hard

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

to detect by humans, can lead to very different ML predictions [4]. This data dependency in the transformations induced by ML models is their very strength: It allows these systems to adapt to any problem setting by learning from rules defined implicitly by a data set. But this flexibility comes at a cost: The more complex and powerful ML systems become, the more difficult it becomes to validate their predictions.

For decades ML scientists have been considering data validation rather an engineering challenge than a research problem. Recently however with more ML systems being deployed in customer facing applications newspaper headlines remind us that without proper validation of ML components we will see more racist chatbots[1] or fatal car crashes[2]. Scientists in the field are beginning to take this problem very seriously. There is an emergent field of research around data management tailored to ML workflows [36]. Testing, monitoring and validation of ML systems and the data ingested and produced by ML models has become a major focus of research with dedicated workshops at major conferences and a rapidly growing research community, as summarized in section 3. Also the recent trend in ML to render models and their predictions more transparent [52] can be regarded as an attempt to validate ML systems and training or prediction data [67]. However to the best of our knowledge there is no commonly agreed upon data validation solution for ML systems that has reached broad adoption. Here, we argue that one of the most important factors is that *automating* validation of ML systems is difficult. Especially when systems learn autonomously and continuously, it can be challenging to ensure that the performance of an ML system is not shifting due to accidental or adversarial changes in the data. Given the above examples for how ML systems can fail in practice, it is obvious that ML systems require scalable, robust and automated data validation solutions. This constraint does not apply to academic research, and thus the lack of automation in ML system validation can be considered as a major blocker slowing down the transfer of the often rapidly evolving advances in ML research into robust and trustworthy customer facing products.

But why is this so difficult to automate? Researchers in the DBMS community have been investigating data profiling for decades [3] and many of these solutions are just right for some aspects of data validation also in the context of ML systems. However some of the challenges in ML system monitoring go beyond that. Many data validation aspects in ML systems *depend on the state of the trained ML model*, such as the performance of a ML model under data set shift, or the differential privacy of a model [16, 2, 5]. Other aspects such as fairness of a ML model require domain expertise that ML engineers often do not have. As illustrated in Figure 1 the competencies required to validate data in the various stages of an ML system require competencies that are usually distributed across several experts in a team or even separate teams.

In the following chapters we will first review the challenges associated with data validation in ML systems and highlighting some of the practical implications. Afterwards we review some of the data validation solutions, with a special focus on the practical applicability of these approaches. Finally we will conclude with an outlook of technical and non-technical challenges associated with ML system validation in order to ensure more responsible usage of ML systems in production software systems.

2 Data Validation Dimensions

As ML systems learn from data, validation of the data ingested during training and prediction is a fundamental prerequisite for well functioning and robust ML systems [58, 10]. Relevant data validation dimensions for ML systems can be broadly categorized into those notions of data quality commonly considered in classical data base management systems (DBMS) and ML model dependent dimensions. In the following, we highlight some of the dimensions and provide examples for each category.

[1] https://blogs.microsoft.com/blog/2016/03/25/learning-tays-introduction/

[2] https://www.nytimes.com/2016/09/15/business/fatal-tesla-crash-in-china-involved-autopilot-government-tv-says.html

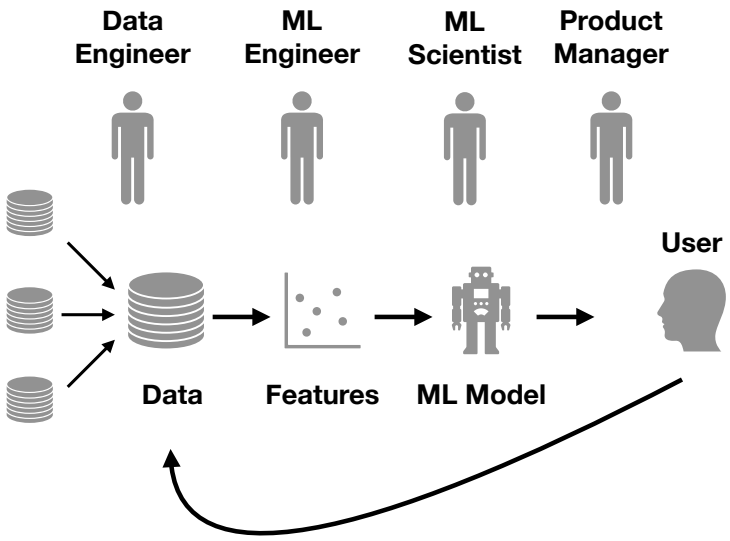

Figure 1: Responsibilities for a machine learning (ML) production software system are often distributed across different roles and teams [1]. Data from various sources is integrated, features are extracted and ML models transform these into predictions, which are then served to users whose behavioural signals are fed back to the system. Monitoring and validating these data streams at scale is only feasible with automated tests. But validating the data in each of these stages requires different competencies and often domain expertise. This imposes novel technical and conceptual challenges on engineering teams when automating ML production systems.

2.1 Classical DBMS dimensions

Independent of ML applications there are many data validation dimensions that have been investigated in the DBMS community, more specifically in the data profiling literature[3]:

Data Correctness: This dimension refers to schema violations, such as string values in numeric columns, or generally syntactically wrong data. Defects in this dimension can often break ML processing pipelines. Some of these violations can be detected easily for instance by type checks, and corrupted data instances can be filtered out.

Data Consistency refers to defects that are not captured by syntactic correctness, including duplicate entries or invalid values. Detecting these cases can be difficult and computationally challenging, but there exist efficient approaches to de-duplication and detection of semantic inconsistencies. Violations of semantic inconsistencies can for instance be detected by validation of functional or approximate functional dependencies [46, 35].

Completeness of a data source reflects the ratio of missing values in a data set. In principle this dimension is simple to probe, however only under the assumption that the missing value symbol is known. Unfortunately this assumption is often violated. As missing values often cause problems in data pipelines feeding into ML systems, a common strategy of avoid such problems is to replace missing values with syntactically valid values. These transformations often make the missing values pass validations for correctness and consistency.

Statistical Properties: This dimension includes basic descriptive statistics of an attribute, such as mean of a numeric or mode of a categorical variable, and more complex parametric or non-parametric statistical aggregates

of a data set. A popular validation strategy in this data dimension is anomaly detection.

2.2 ML model dependent dimensions

Validating the above data quality dimensions can be challenging, but these tests of the input data are independent of the downstream ML model. This implies that testing only for these dimensions can lead to both false negatives, when a data corruption is not detected that has a negative impact on the downstream ML model, as well as false positives, when a data corruption raises an alarm but does not have any effect on the downstream ML model. Examples for false negatives of ML model independent validations include adversarial attacks, such as for instance avoiding filters for adult language by *leet speak*[3] strings obfuscated by misspellings and replacing letters with numeric characters. Examples for false positive alarms are ML models employing strong L_1 or sparsity-enforcing regularization constraints. In these models, a large number of input features is often not affecting the ML model output, hence distributional shifts in these input features would not be reflected in shifts of the ML model outputs and validation of the input data independent of the model outputs will raise false alarms.

Data validation in the context of ML production systems thus has to take into account the current state of the downstream ML model to ensure robust and reliable performance. The state of the ML model and and the impact of data corruptions on ML performance can be monitored by a number of different validation dimensions of the ML model *output* data. These have to be validated along with the usual criteria common to all production software systems, such as requirements on the latency and availability of the system. SageMaker Model Monitor[4] is a solution that enables monitoring of models deployed in production. For example, alarms are triggered when data distributions shifts are detected.

Predictive performance metrics are usually the most important metrics optimized in production ML systems. These metrics include accuracy, precision or F1-score of a classifier, the mean-squared error, mean absolute error or r^2-scores for regression models or various ranking losses, evaluated via cross-validation on a test set that was not used for training the respective ML model. When the data processed by trained and deployed ML model is drawn from the same distribution as the training and test data, then these cross-validated metrics reflect the predictive performance reliably. But in production ML systems, this assumption is often violated. Examples include shifts in the input data (covariate shifts) or shifts in the target data distribution (label shift).

Robustness The shifts induced by corruptions in the data that are a) not caught by upstream classical data validation techniques and that have b) a negative impact on the predictive performance of a production ML system can be difficult to detect, as there is no ground truth label information available for the data. In the absence of ground truth data needed to compute predictive performance metrics, one can resort to classical data validation techniques applied to the outputs of ML models. But this approach can fail to detect certain covariate shifts induced by outliers, adversarial attacks or other corruptions.

Privacy metrics have become increasingly important since ML models are used to process personal data such as health care data, financial transactions, movement patterns or really almost every aspect of our lives. Popular concepts include *differential privacy* [17] and *k-anonymity* [61]. Note that k-anonymization [61] is not model dependent, it is a property of a data base where user data is considered private if information about each user cannot be distinguished from at least k-1 other users whose information is in the dataset. In the field of ML, differential privacy is more useful than k-anonymity. Two main reasons for that are a) k-anonymity is defined for a fixed data set, if new rows are added, one might need to convert the data again in order to achieve k-anonymity,

[3] `https://en.wikipedia.org/wiki/Leet`

[4] `https://docs.aws.amazon.com/sagemaker/latest/dg/model-monitor.html`

and b) if a k-anonymous data set is combined with other information, it is often possible to de-anonymize the data. A large body of literature in the ML community builds upon the definition of ϵ-differential privacy [18] which deals with the problem of learning little to nothing about individual data-points while learning a lot about the population by ensuring the same conclusions can be drawn whether any single data-point is included in the dataset or not. Most implementations of differentially private ML models add noise to the data, gradients or the model coefficients [2, 59, 47, 42]. This can be an effective countermeasure to model inversion attacks [20] or membership inference attacks [59]. The amount of noise added determines the amount of privacy preserved – but it also bounds the predictive performance or *utility* of the ML model [30].

Robustness to adversarial data Recent developments have shown that ML models, placed at the centre of security-critical systems such as self-driving cars, can be targeted by dedicated adversarial examples - inputs that are very difficult to distinguish from regular data but are misclassified by the model [22]. Those inputs, created by applying a small but very significant perturbation to regular examples can be used to manipulate the model to predict outputs desired by an adversary [62, 9, 44]. The robustness to adversarial attacks of the model can be measured by (1) generating a set of adversarial examples based on the test set [62, 37, 28, 43, 22] and (2) computing what is the change in models accuracy given a fixed average distortion, or what is the minimum average distortion needed to reach 0% accuracy on the test set.

Fairness Another dimension that is highly relevant when putting ML systems in production is that of fairness or more generally ethical aspects of ML predictions [40]. There are a number of examples when ML technology used as assistive technology for instance in border control[5] or policing[6] led to racist or otherwise biased decisions. It has been widely recognized in the ML research community that when building ML applications validating the ML predictions with respect to fairness is a fundamental prerequisite. Evaluating this dimension is however amongst the most difficult of all validation challenges: there are different fairness definitions, and even if one could agree on a small set of definitions, validating these requires insight into the respective grouping variables, which are, for ethical and legal reasons, typically not available. The relevant grouping information is thus often reflected in a data set through other features, which are considered not ethically problematic, such as zip code, but which are often correlated with ethically sensitive features. Other implicit biases of data sets can emerge from data preprocessing, for instance how missing values are dealt with when they are distributed not equally across ethnical groups [65]. Detecting these biases requires not only domain expertise, but also often information that is not contained in the data set itself.

3 Current Solutions

Validating data at each of the stages in a ML workflow as sketched in Figure 1 requires a wide range of competencies from data engineering over ML algorithms to user interface design. These competencies are often distributed across different roles in a team, and especially in small (5 to 10 members) teams, it is not unlikely that only one person has the competency to dive deep into a root cause analysis of the various data validation dimensions sketched in section 2. Solving problems in an on-call situation or when scaling up an ML system requires reliable automation for the data validation. As many of these validations have to be data dependent, such automation often amounts to *using ML to validate data in ML workflows* which has, for instance, been done for outlier removal [68], data cleaning [34] or missing value imputation [8]. There are, however, simpler approaches that do not leverage ML models to validate data in ML systems, such as Tensorflow Extended (TFX) [6] or DEEQU [54]. In the following we will highlight some of the state of the art solutions to automated data validation in the context of ML systems.

[5] `https://www.bbc.com/news/technology-53650758`

[6] `https://www.nytimes.com/2019/07/10/opinion/facial-recognition-race.html`

3.1 Schema Validation

Validating syntactic correctness, also referred to as schema validation, is – at first glance – relatively simple and often integral part of data ingestion. Usually data violating data type constraints will immediately raise errors and break a data pipeline. Yet, these validations are often counteracted by humans. In practice it is not uncommon to use generic data types such as strings for data that is actually numeric, ordinal or categorical. While this will ensure that schema violations are minimized, as any data type can be cast to string, it imposes additional work on any downstream validation for data correctness and consistency. This problem is worsened by the fact that some software libraries popular in the ML community often cast string data that contains numeric data automatically to numeric data types. This can have effects on entire scientific communities. A recent study reports that casting errors induced by excel in biomedical data occur in approximately one fifth of the papers in leading genomics journals [70].

While simple validation such the data type can be trivially automated, it is unlikely that this will prevent engineers from having to deal with data type errors. There will always be on-call engineers who need to fix problems as fast as possible and the cost of having stale or broken data pipelines in a production system is often far bigger then the difficulty to quantify cost of accumulating hidden technical debt by specifying generic data types like strings and thereby circumventing data type checks. This is why this simple task of data type detection is actually an interesting research challenge. Often simple heuristics will do the job and the popular libraries use these heuristics to determine the data type of columns. The simplest heuristic is probably to try and cast a value to a certain data type and try other data types in case an error is raised[7]. Such heuristics often work well in practice but can be insufficient, especially when dealing with heterogeneous data, such as mixed data types in one column. Also, for the seemingly simple task of automating data type inference, there is interesting recent work on using ML models to infer a probabilistic notion of the most likely data type given the symbols used in a data base [12]. Approaches combining scalable heuristics with more sophisticated ML based data type inference techniques are a promising alternative to the current situation in which restrictive data type checks and broken data pipelines often lead data engineers to opt for too generic data types.

3.2 Data Correctness and Data Consistency

Validation of data consistency and correctness are one of the areas where classical DBMS research has made significant contributions, for instance in the data profiling literature [3]. Popular approaches to validate correctness and consistency are rule based systems such as *NADEEF* [13]. There are commercial versions of simple rule based systems, a popular example is trifacta[8]. And there are open source systems for semi-automated validation of consistency, such as Open-Refine[9]. Other approaches primarily investigated in an academic context include attempts to learn *(approximate/relaxed) functional dependencies* between columns of a table [46]. These functional dependencies between columns include for example a dependency between a column containing zip codes and another column containing city names. These dependencies can be used to automatically learn validation checks related to consistency and correctness [50]. Such approaches are well established and there exist efficient methods for large scale functional dependency discovery [35]. Yet, in the context of data pipelines for ML systems none of these approaches have reached broad adoption. We discuss some potential reasons in section 4.

In addition to the solutions of the DBMS community there are a number of approaches developed at the intersection of DBMS and ML communities, tailored to ML specific workflows [11, 54]. The research is accompanied by open source libraries[10][11] that implement schema validation by checking for correct data types, but they also offer validation options for or other basic data properties, for instance whether the values are contained

[7] https://github.com/pandas-dev/pandas/blob/v1.1.4/pandas/core/dtypes/common.py#L138
[8] https://www.trifacta.com/
[9] https://openrefine.org/
[10] https://www.tensorflow.org/tfx/guide/tfdv
[11] https://github.com/awslabs/deequ

in a predefined set. Prominent examples for a data validation solution for ML systems are the ones integrated in SageMaker Data Wrangler[12] and TFX [11]. Their schema validation conveniently allows users to define integrity checks on the serving data in an ML system, e.g. for data type or value ranges. Schema validation parameters can also be derived from the training data, which allows to automate many of these checks.

3.3 Validation of Statistical Properties

All data sets are subject to noise, be it because humans enter wrong values into a form or because human software developers write code that leads to faulty data ingestion. Dealing with this noise is difficult for exact validation rules such as functional dependencies. Statistical learning techniques can help to deal with noisy data.

Error-Detection Solutions Taking a probabilistic stance on DBMSs is not a new idea [48]. Recently, the idea of using statistical techniques for data validation tasks has become a major focus of research. One example are statistical learning approaches to error detection in data bases [50, 25, 29, 39]. These approaches have the potential to automatically validate data correctness and to automatically clean data sets. Some of these systems can be automated to a large extent [32, 34], others rely on a semi-supervised approach [33].

ML Frameworks for Statistical Data Validation Also data validation solutions like TFX data validation [11] or DEEQU [54] address data validation including statistical rules, such as deviation of values around the mean of a numeric column. These validation solutions can be very helpful, if one knows what to test for. But data sources can easily have billions of rows and hundreds of columns. For these cases it can infeasible to manually create, validate and maintain data quality checks. To alleviate the burden of manually creating constraints, the authors of DEEQU [54] propose to utilize historic data to generate column profiles and generate data quality constraints from these. These quality constraints can also make use of the temporal structure of data quality metrics collected over time using time series models or other anomaly detection methods.

Anomaly Detection Methods Instead of applying them to metrics computed on a data set, anomaly detection methods are also often used to detect anomalies in the data tuples directly. There are many methods available in easy to use software libraries, see for instance [68], and there are commercial products that allow to automate anomaly detection in large scale industrial settings[13]. While the goal of these anomaly detection approaches is similar to the above error detection approaches originating in the DBMS community, many anomaly detection solutions emerged from the ML community. One of the most important differences is that anomaly detection approaches, as most ML methods, usually expect the data to come in matrix form, with all columns being numeric. Most methods from the DBMS community expect the data to be high cardinality categorical data, some also are applicable to numeric values, but that is not very common in research on functional dependencies for instance [46]. So applying anomaly detection methods to heterogeneous data sets with non-numeric (categorical, ordinal or even free text data) requires to apply *feature extraction* techniques to bring the data into a numeric format. This is a standard preprocessing step in ML pipelines, but popular software libraries for anomaly detection, such as [68], do not include this important preprocessing step. This makes these libraries difficult to apply for automated data validation. It is possible to integrate standard anomaly detection methods as statistical data validation steps in ML systems, but this imposes two additional challenges onto the engineering team. For one, this integration requires to write 'glue code' for the feature extraction – which is often one of the major sources for accumulating *technical debt* in a software system. And secondly this requires to have a good evaluation

[12] `https://aws.amazon.com/sagemaker/data-wrangler/`

[13] `https://aws.amazon.com/de/blogs/big-data/real-time-clickstream-anomaly-detection-with-amazon-kinesis-analytics/`

metric for the anomaly detection. Which is, in contrast to standard supervised learning scenarios, often difficult to define and get ground truth data for.

Model Robustness and Generalization Performance Another central problem with all of the above approaches to statistical data validation in a ML context is that most of these methods are blind to the impact of data errors on downstream ML components. This is however one of the most important aspects for ML systems. Often it does not matter for the ML model, whether a feature is affected by a data shift, for instance when regularization of an ML model forces the ML model to ignore a feature. And in other cases tiny changes in the input, which human observers would not be able to detect, can have devastating consequences on the ML model output [4]. Recent work in the ML community has shown that especially novel deep learning methods can suffer from severe stability problems [14]. Some aspects of this can be mitigated by employing standard ML concepts such as k-fold cross-validation (CV). This approach has unfortunately lost popularity due to the sheer compute demand of modern deep learning methods. Most deep learning papers usually use just a single train/validation/test split. Standard nested k-fold CV can have decisive advantages when it comes to measuring robustness of a ML model. However, these techniques only work when there is ground truth data available. In a production setting, this is often not the case. There exist however also other methods to measure the robustness of ML models when there is no ground truth data available. For instance in [55] the authors leverage a set of declaratively defined data errors applied to data for which ground truth is available and measure the predictive performance of a ML model under these perturbations. This allows to train a meta model that can be used to predict the predictive performance on new unseen data with high precision. Such an approach can be useful in production ML systems to automatically validate data errors with respect to their impact on downstream ML model performance.

3.4 Fairness Validation

Fairness is one of the most prominent examples of how ML systems can fail and severely influence the public opinion about a company or an entire scientific community. It is thus of utmost importance to ensure that this dimension of data validation in the context of ML systems is not neglected. Yet validating this dimension is especially difficult, for a number of reasons. First and foremost it is difficult to define fairness. An excellent overview over the last decades of fairness research with a focus on ML systems can be found in [40]. The main insight here is that fairness validation it is not only a technical challenge. Instead, it is imperative to include multiple scientific disciplines in this research, in particular also researchers from sociology, psychology and law. Setting the stage for such transdisciplinary research is a challenge in itself, for instance finding a common terminology is not trivial. But we have seen that the research community has made progress by fostering transdisciplinary discussions at newly emerging conferences[14]. The joint efforts of different communities have helped to identify many ways in which biases leading to unfair ML systems can enter a workflow. Many of these biases arise in the data generating process. Enabling scientists and engineers to identify such biases should be part of the research agenda of the data management and ML community [60]. One example in this direction is *FairBench* [65], an open source library that helps to trace changes in data distributions and visualize distortions with respect to protected group memberships throughout the pipeline. Another example is SageMaker Clarify[15], an explainability feature for Amazon SageMaker that provides insights into data and ML models by identifying biases and explaining predictions. It is deeply integrated into Amazon SageMaker, a fully managed service that enables data scientists and developers to build, train, and deploy ML models at any scale. Clarify supports bias detection and feature importance computation across the ML lifecycle, during data preparation, model evaluation, and post-deployment monitoring. Libraries like these are a prerequisite to better automation of fairness

[14] See for instance `https://dl.acm.org/conference/fat`

[15] https://aws.amazon.com/sagemaker/clarify/

validation. However, another more fundamental problem of fairness validation remains even if technical solutions for visualizing and detecting biases are available: Group membership variables are required for most technical solutions to fairness. Storing these variables can be more challenging, from an operational stance, than storing other non-confidential data.

3.5 Privacy Validation

Similar to fairness, also privacy is a difficult to define validation dimension. However in contrast to fairness, the research community has converged to a few concepts that are relatively straightforward in terms of their definitions. Popular concepts include *differential privacy* [17] and *k-anonymity* [61], see also section 2. Most ML related work on privacy focuses on differential privacy, where noise is added the data, gradients or the model coefficients. Validating and trading off privacy against predictive performance or *utility* of the ML model can be challenging [30]. Empirically, evaluating privacy is often done using *membership inference attacks* [59], which has also been adopted for unsupervised ML models [24]. One limitation of these approaches is that privacy validation is always dependent on a specific model and data set. General statements about privacy and utility independent of models and data is hence difficult.

3.6 Validation of Robustness against adversarial attacks

Privacy validation is aiming at defending the ML system against a certain type of adversarial attack, where for instance the identity of data points used for training the ML system is revealed. There are however other types of adversarial attacks, for instance when artificial examples are generated to result in ML predictions with a certain outcome. Validation of robustness against such types of attacks can be achieved by perturbations around the data manifold [45, 38]. This can be achieved by extracting latent representations of the input data [26] or of the predictions [7, 19]. Alternative methods rely on training an additional classifier used to decide whether an example is adversarial or not [21, 23, 41]. Complementary to the work on validating adversarial robustness, a lot of work has been devoted to making ML models more robust to adversarial attacks by augmenting the training datasets with adversarial examples [22, 38, 66, 69].

3.7 Human in the loop evaluation

Most of the above data quality dimensions are easy for humans to assess. This is why human audits are still one of the most direct and most robust options for data validation in ML systems. Expert auditors, such as researchers developing a new service, often can quickly identify errors and their root causes by simply inspecting input and outputs of a ML system. Among the most important disadvantages with this approach is that these validations are expensive and do not scale well. Sometimes human-in-the-loop validations can be scaled up using crowd-sourcing platforms such as Yandex' Toloka or Amazon Mechanical Turk. Increasing the quality of crowd-sourced validations is an active topic of ML research [64]. For instance there are attempts to render audits more efficient by including transparency of ML model predictions [56] or by providing more inciting incentives [27, 63]. Still, this approach can be difficult to automate and is generally used as an andon cord in a modelling pipeline rather than an integrated quality test. This is not only due to the fact that it is difficult to integrate human audits in build processes. Focussing on human judgements only can lead to biased validations, especially when using transparent ML methods [57].

4 Conclusion

Validation of input and output data in ML production systems has many facets that require competencies often distributed across a heterogeneous team of engineers and scientists, as illustrated in Figure 1. While some of the

data validation challenges, such as schema validation or data consistency, can be tackled with traditional data profiling methods established the DBMS community, other validation dimensions are specific to ML systems. These ML model dependent validation challenges include aspects like accuracy and robustness under data shifts, fairness of ML model predictions and privacy concerns.

In section 3 we highlight a number of solutions to validate single aspects. Many of these approaches are typically tailored to specific use cases and often require considerable integration efforts in production ML systems. A good example are the various solutions from both the ML community as well as the DMBS community for checking simple data properties, such as the data types, and also more complex dimensions like data consistency. Many of these approached allow for automating the generation of validation checks. Yet in practice it is not trivial to automate the generation of validations for a new ML system that ingests and produces millions of rows and columns. For instance, there are many cases when simple validation checks on input data will lead to false alarms when shifts or errors in a single feature do not impact the output of a downstream ML model – maybe because that feature was neglected by the ML model, when strong regularization during the ML model training phase taught the model to ignore that feature.

Despite the rapid progress in recent years to automate and monitor ML systems: to the best of our knowledge there exists no data validation system that has reached broad adoption and which takes into account all of the data validation dimensions sketched in section 2. One reason for this is the difficulty of combining the multitude of validation strategies listed in section 3 into one unified framework. Considering the rapid pace of research at the intersection of ML and DBMS, see for instance [15], it is fair to assume that it is merely a matter of a few years until one framework or some open standard for data validation in the context of ML systems will have reached broad adoption.

There are many data validation challenges in ML systems that go beyond technical aspects. Many of them are due to the complex nature of the data transformations induced by ML models. For instance identifying unfair biases often requires domain knowledge or access to grouping variables, which are often not available. And even if those are available, it is not always clear how fairness in the context of ML systems can be defined [67]. A conceptual challenge related to privacy is for instance the trade-off between utility and differential privacy of a ML system [30]: how much predictive accuracy should be sacrificed to ensure privacy? Sacrificing accuracy against privacy in domains like health care or jurisdiction is a difficult question for which ethical and legal dimensions are more important than technical aspects. Next to these ethical and legal aspects, there is one key factor hindering adoption of comprehensive data validation in ML systems and that more related to cultural aspects. Many scientists like to build new models and tools, but writing tests, integrating monitoring and validation stages in an ML system are not exactly the most popular tasks amongst researchers. But often the competencies of the scientists who built a model is required to build well functioning monitoring and validation solutions in ML systems.

Based on these observations we derive some suggestions for how to drive innovation and adoption of data validation in the context of ML systems. First, we hope that the current trend for research at the intersection of ML and DBMS communities will continue to grow and identify more synergies leveraging and complementing each others expertise. We have seen some great examples of constructive but vivid discussion between the two communities, for instance that sparked by Kraska and colleagues around their work on learning index structures [31]. This work is unrelated to data validation and mentioned merely as an example of transdisciplinary research debates. Second, when building ML systems there is a broad spectrum of operational challenges and seamless integration with cloud infrastructure is key to reaching broad adoption. We conclude that establishing data validation in ML systems will require a stronger focus on usability and simple APIs. Third we believe that data validation in ML systems will reach broad adoption once the research community will have found better ways of automating the validation workflow, most importantly the generation of checks for each of the data validation dimensions listed in section 2.

In the past years we have seen great examples of automated tooling for tracking ML experiments [53], experimentation in ML production systems [10], input data validation [54, 11] and validation strategies for

predictions of ML systems [49, 55, 14]. One example of how some of these data validation techniques could be integrated into an automated workflow would be that presented in [51], where the authors propose to iterate through a sequence of data validation [54], data cleaning[8] and quantification of downstream impact on ML predictive performance [55] to achieve an automated ML workflow. We believe that increasing the level of usability through automation in data validation will enable researchers to focus on more important questions like the conceptual, ethical and legal questions and ultimately lead to more responsible usage of ML systems in production systems.

References

[1] *Data Lifecycle Challenges in Production Machine Learning: A Survey*, number (Vol. 47, No. 2). SIGMOD Record, 2018.

[2] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pages 308–318, New York, NY, USA, 2016. Association for Computing Machinery.

[3] Ziawasch Abedjan, Lukasz Golab, Felix Naumann, and Thorsten Papenbrock. Data Profiling. *Synthesis Lectures on Data Management*, 10(4):1–154, nov 2018.

[4] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. volume 80 of *Proceedings of Machine Learning Research*, pages 284–293, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[5] Eugene Bagdasaryan, Omid Poursaeed, and Vitaly Shmatikov. Differential privacy has disparate impact on model accuracy. In H. Wallach, H. Larochelle, A. Beygelzimer, F. Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, pages 15479–15488. Curran Associates, Inc., 2019.

[6] Denis Baylor, Eric Breck, Heng Tze Cheng, Noah Fiedel, Chuan Yu Foo, Zakaria Haque, Salem Haykal, Mustafa Ispir, Vihan Jain, Levent Koc, Chiu Yuen Koo, Lukasz Lew, Clemens Mewald, Akshay Naresh Modi, Neoklis Polyzotis, Sukriti Ramesh, Sudip Roy, Steven Euijong Whang, Martin Wicke, Jarek Wilkiewicz, Xin Zhang, and Martin Zinkevich. TFX: A TensorFlow-based production-scale machine learning platform. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, volume Part F129685, pages 1387–1395, New York, NY, USA, aug 2017. Association for Computing Machinery.

[7] Arjun Nitin Bhagoji, Daniel Cullina, Chawin Sitawarin, and Prateek Mittal. Enhancing robustness of machine learning systems via data transformations. *2018 52nd Annual Conference on Information Sciences and Systems (CISS)*, Mar 2018.

[8] Felix Biessmann, David Salinas, Sebastian Schelter, Philipp Schmidt, and Dustin Lange. "Deep" Learning for Missing Value Imputationin Tables with Non-Numerical Data. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management - CIKM '18*, pages 2017–2025, New York, New York, USA, 2018. ACM Press.

[9] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. *Lecture Notes in Computer Science*, pages 387–402, 2013.

[10] Joos Hendrik Böse, Valentin Flunkert, Jan Gasthaus, Tim Januschowski, Dustin Lange, David Salinas, Sebastian Schelter, Matthias Seeger, and Yuyang Wang. Probabilistic demand forecasting at scale. *Proceedings of the VLDB Endowment*, 10(12):1694–1705, 2017.

[11] Eric Breck, Neoklis Polyzotis, Sudip Roy, Steven Euijong Whang, and Martin Zinkevich. DATA VALIDATION FOR MACHINE LEARNING. Technical report, 2019.

[12] Taha Ceritli, Christopher K.I. Williams, and James Geddes. ptype: probabilistic type inference. *Data Mining and Knowledge Discovery*, 34(3):870–904, may 2020.

[13] Michele Dallachiesat, Amr Ebaid, Ahmed Eldawy, Ahmed Elmagarmid, Ihab F. Ilyas, Mourad Ouzzani, and Nan Tang. NADEEF: A commodity data cleaning system. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 2013.

[14] Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D. Hoffman, Farhad Hormozdiari, Neil Houlsby, Shaobo Hou, Ghassen Jerfel, Alan Karthikesalingam, Mario Lucic, Yian Ma, Cory McLean, Diana Mincu, Akinori Mitani, Andrea Montanari, Zachary Nado, Vivek Natarajan, Christopher Nielson, Thomas F. Osborne, Rajiv Raman, Kim Ramasamy, Rory Sayres, Jessica Schrouff, Martin Seneviratne, Shannon Sequeira, Harini Suresh, Victor Veitch, Max Vladymyrov, Xuezhi Wang, Kellie Webster, Steve Yadlowsky, Taedong Yun, Xiaohua Zhai, and D. Sculley. Underspecification Presents Challenges for Credibility in Modern Machine Learning. 2020.

[15] Xin Luna Dong and Theodoros Rekatsinas. Data Integration and Machine Learning: A Natural Synergy. *Proceedings ofthe VLDB Endowment, Vol.*, 11(12):2094–2097, 2018.

[16] Cynthia Dwork. Differential privacy: A survey of results. In *In Theory and Applications of Models of Computation*, pages 1–19. Springer, 2008.

[17] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In Shai Halevi and Tal Rabin, editors, *Theory of Cryptography*, pages 265–284, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[18] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3–4):211–407, August 2014.

[19] Reuben Feinman, Ryan R. Curtin, Saurabh Shintre, and Andrew B. Gardner. Detecting adversarial samples from artifacts, 2017.

[20] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the ACM Conference on Computer and Communications Security*, 2015.

[21] Zhitao Gong, Wenlu Wang, and Wei-Shinn Ku. Adversarial and clean data are not twins, 2017.

[22] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples, 2014.

[23] Kathrin Grosse, Praveen Manoharan, Nicolas Papernot, Michael Backes, and Patrick McDaniel. On the (statistical) detection of adversarial examples, 2017.

[24] Jamie Hayes, Luca Melis, George Danezis, and Emiliano De Cristofaro. LOGAN: Membership Inference Attacks Against Generative Models. *Proceedings on Privacy Enhancing Technologies*, 2019(1):133–152, 2019.

[25] Alireza Heidari, Joshua McGrath, Ihab F Ilyas, and Theodoros Rekatsinas. HoloDetect: Few-Shot Learning for Error Detection. *ACM Reference Format*, 2019.

[26] Dan Hendrycks and Kevin Gimpel. Early methods for detecting adversarial images, 2016.

[27] Chien-Ju Ho, Aleksandrs Slivkins, Siddharth Suri, and Jennifer Wortman Vaughan. Incentivizing high quality crowdwork. *Proceedings of the 24th International Conference on World Wide Web - WWW '15*, 2015.

[28] H. Hosseini, S. Kannan, and R. Poovendran. Are odds really odd? bypassing statistical detection of adversarial examples. *ArXiv*, abs/1907.12138, 2019.

[29] Zhipeng Huang and Yeye He. Auto-Detect: Data-Driven Error Detection in Tables. *SIGMOD*, 2017.

[30] Bargav Jayaraman and David Evans. *Evaluating Differentially Private Machine Learning in Practice*. 28th USENIX Security Symposium, 2019.

[31] Tim Kraska, Alex Beutel, Ed H Chi, Jeffrey Dean, and Neoklis Polyzotis. The case for learned index structures. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 489–504, 2018.

[32] Sanjay Krishnan, Michael J Franklin, Ken Goldberg, and Eugene Wu. BoostClean: Automated Error Detection and Repair for Machine Learning.

[33] Sanjay Krishnan, Jiannan Wang, Eugene Wu, Michael J. Franklin, and K. Goldberg. ActiveClean: Interactive data cleaning for statistical modeling. *Proceedings of the VLDB Endowment*, 9(12):948–959, 2016.

[34] Sanjay Krishnan and Eugene Wu. AlphaClean: Automatic Generation of Data Cleaning Pipelines. apr 2019.

[35] Sebastian Kruse and Felix Naumann. Efficient Discovery of Approximate Dependencies. *PVLDB*, 11(7):759–772, 2018.

[36] Arun Kumar, Matthias Boehm, and Jun Yang. Data Management in Machine Learning: Challenges, Techniques, and Systems. In *SIGMOD*, 2017.

[37] A. Kurakin, Ian J. Goodfellow, and S. Bengio. Adversarial examples in the physical world. *ArXiv*, abs/1607.02533, 2017.

[38] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks, 2017.

[39] Mohammad TU Mahdavi Berlin mahdavilahijani, tu-berlinde Ziawasch Abedjan, Raul Castro Fernandez MIT, Samuel Madden MIT, Mourad Ouzzani, Michael Stonebraker MIT, Nan Tang, Mohammad Mahdavi, Ziawasch Abedjan, Raul Castro Fernandez, Samuel Madden, and Michael Stonebraker. Raha: A Configuration-Free Error Detection System. 18(19).

[40] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A Survey on Bias and Fairness in Machine Learning. aug 2019.

[41] Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations, 2017.

[42] I. Mironov. Rényi differential privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pages 263–275, 2017.

[43] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and P. Frossard. Deepfool: A simple and accurate method to fool deep neural networks. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2574–2582, 2016.

[44] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2015.

[45] Tianyu Pang, Chao Du, Yinpeng Dong, and Jun Zhu. Towards robust detection of adversarial examples, 2017.

[46] Thorsten Papenbrock, Jens Ehrlich, Jannik Marten, Tommy Neubert, Jan-Peer Rudolph, Martin Schönberg, Jakob Zwiener, and Felix Naumann. Functional dependency discovery. *Proceedings of the VLDB Endowment*, 8(10):1082–1093, jun 2015.

[47] Nicolas Papernot, Martín Abadi, Úlfar Erlingsson, Ian Goodfellow, and Kunal Talwar. Semi-supervised knowledge transfer for deep learning from private training data, 2016.

[48] Michael Pittarelli. Roger Cavallo. *Proceedings of the Thirteenth International Conference on Very Large Data Bases*, (January):71–81, 1987.

[49] Stephan Rabanser, Stephan Günnemann, and Zachary C Lipton. Failing loudly: An empirical study of methods for detecting dataset shift, 2018.

[50] Theodoros Rekatsinas, Xu Chu, Ihab F Ilyas, and Christopher Ré. HoloClean: Holistic Data Repairs with Probabilistic Inference. *Proceedings ofthe VLDB Endowment*, 10(11), 2017.

[51] Tammo Rukat, Dustin Lange, Sebastian Schelter, and Felix Biessmann. Towards Automated ML Model Monitoring: Measure, Improve and Quantify Data Quality. Technical report.

[52] Wojciech Samek and Klaus Robert Müller. Towards Explainable Artificial Intelligence. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 11700 LNCS, pages 5–22. Springer Verlag, 2019.

[53] Sebastian Schelter, Joos-Hendrik Boese, Johannes Kirschnick, Thoralf Klein, and Stephan Seufert. Automatically tracking metadata and provenance of machine learning experiments. *Machine Learning Systems workshop at NeurIPS*, 2017.

[54] Sebastian Schelter, Dustin Lange, Philipp Schmidt, Meltem Celikel, Felix Biessmann, Andreas Grafberger, and Meltem Ce-Likel. Automating Large-Scale Data Quality Verification. *PVLDB*, 11(12):1781–1794, 2018.

[55] Sebastian Schelter, Rukat Tammo, and Felix Biessmann. Learning to Validate the Predictions of Black Box Classifiers on Unseen Data. *SIGMOD*, 2020.

[56] Philipp Schmidt and Felix Biessmann. Quantifying interpretability and trust in machine learning systems, 2019.

[57] Philipp Schmidt and Felix Biessmann. Calibrating human-ai collaboration: Impact of risk, ambiguity and transparency on algorithmic bias. In Andreas Holzinger, Peter Kieseberg, A Min Tjoa, and Edgar Weippl, editors, *Machine Learning and Knowledge Extraction*, pages 431–449, Cham, 2020. Springer International Publishing.

[58] D Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips, Dietmar Ebner, Vinay Chaudhary, Michael Young, and Dan Dennison. Hidden Technical Debt in Machine Learning Systems. *Nips*, pages 2494–2502, 2015.

[59] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership Inference Attacks Against Machine Learning Models. *Proceedings - IEEE Symposium on Security and Privacy*, pages 3–18, 2017.

[60] Julia Stoyanovich, Bill Howe, and H. V. Jagadish. Responsible data management. *Proceedings of the VLDB Endowment*, 13(12):3474–3488, 2020.

[61] Latanya Sweeney. K-anonymity: A model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5):557–570, October 2002.

[62] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks, 2013.

[63] H. Wang, S. Guo, J. Cao, and M. Guo. Melody: A long-term dynamic quality-aware incentive mechanism for crowdsourcing. *IEEE Transactions on Parallel and Distributed Systems*, 29(4):901–914, 2018.

[64] Jennifer Wortman Vaughan. Making Better Use of the Crowd: How Crowdsourcing Can Advance Machine Learning Research. Technical report, 2018.

[65] Ke Yang, Biao Huang, and Sebastian Schelter. Fairness-Aware Instrumentation of Preprocessing Pipelines for Machine Learning. 2020.

[66] Haichao Zhang and Jianyu Wang. Defense against adversarial attacks using feature scattering-based adversarial training, 2019.

[67] Jie M. Zhang, Mark Harman, Lei Ma, and Yang Liu. Machine Learning Testing: Survey, Landscapes and Horizons. *IEEE Transactions on Software Engineering*, pages 1–1, feb 2020.

[68] Yue Zhao, Zain Nasrullah, and Zheng Li. PyOD: A python toolbox for scalable outlier detection. *Journal of Machine Learning Research*, 2019.

[69] Dingyuan Zhu, Ziwei Zhang, Peng Cui, and Wenwu Zhu. Robust graph convolutional networks against adversarial attacks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '19, pages 1399–1407, New York, NY, USA, 2019. Association for Computing Machinery.

[70] Mark Ziemann, Yotam Eren, and Assam El-Osta. Gene name errors are widespread in the scientific literature. *Genome Biology*, 17(1):177, aug 2016.