Automated Data Preparation for Machine Learning: a Survey

Sasa Mladenovic[1], Marius Lindauer[2], Carola Doerr[1]

[1]Sorbonne Université, CNRS, LIP6, Paris, France
[2]Leibniz University Hannover, L3S Research Center, Hannover, Germany

08/2025

Abstract

Data preparation is essential for effective machine learning (ML), yet typically remains a manual, time-consuming process. While automated machine learning (AutoML) has successfully addressed modeling aspects of the ML workflow, data preparation has largely been overlooked, leading to challenges with real-world, imperfect data. Conversely, a rising paradigm in the world of artificial intelligence (AI) and ML is that of data-centric AI, shifting focus from just refining models, to enhancing data in order to advance performance boundaries. This survey motivates the need for automated solutions regarding data preparation, offering a fundamental understanding of the benefits of data transformations and establishing the complexity of data pipeline optimization, while highlighting the importance of data quality. We provide a comprehensive overview and categorization of existing automation approaches, both in AutoML and as standalone fully or semi-automated systems. We discuss underlying methodologies, their advantages, and limitations. Our work explores the prospects of expanding automation to cover a broader data preparation process, aiming to bridge the gap between data-centric AI and AutoML. It paves the way to a wholly automated pipeline from raw real-world data to quality model predictions, and outlines future research directions towards that goal.

1 Introduction

Motivation. The field of artificial intelligence (AI), more specifically machine learning (ML), has undergone a period of explosive growth in recent years. This phenomenon has been fueled by multiple factors, including increasing availability of large amounts of data, advancements in computational power driven by hardware improvements and cloud computing, as well as significant algorithmic innovations such as novel neural network architectures, improved training strategies, and more efficient optimization techniques. One of the challenges that has been raised among these innovations is that of *automating data science* [DBDRH+22].

Data science (DS) allows us to extract insights about the real world trough data-driven approaches, which often draw upon ML techniques. Machine learning heavily relies on human expertise, namely knowledge of the domain its data stems from, as well as technical skills in mathematics, programming, and ML itself. Nonetheless, a significant part of its workflow can also be delegated to automation. This has been the focus of the domain of automated machine learning (AutoML) [HKV19], which leverages automation techniques with the objectives of widening the machine learning audience by making ML more accessible to non-experts, improving the efficiency of the ML process via optimized design choices, and accelerating research in the field through faster experimentation.

The machine learning pipeline can be subdivided into two overarching components: (1) data preparation and (2) modeling. *Data preparation* designates the application of a sequence of transformations to raw data. Its goal is to ensure usability of the data by resolving any incompatibilities with a given ML model, and optimize result quality once it is input into that model. *Modeling*, which represents the core of the ML process, encompasses selecting the most adequate machine learning algorithm to apply to the data, and tuning its hyperparameters so as to obtain the best possible outcomes. These two components are closely intertwined: data preparation can be tailored to a chosen model, while at the same time the state of the data influences the best choice of model [OMB+24].

The field of data science incorporates a deeper data preparation segment that reaches beyond typical data treatment in ML. While the latter is largely focused on model compatibility and performance, broader data preparation, as seen by the database (DB) community, extends to processes aimed at organizing data and raising its quality [CWL+23]. These aspects serve to improve the efficiency, effectiveness, and reliability of any downstream analytics and systems, including machine learning pipelines.

In this work, we consider a holistic data preparation definition integrating the understanding of the term by both the DB and ML communities, as both eventually have a significant impact on ML outcomes. Figure 1 illustrates the structure and relationships between the different entities and processes composing a DS workflow that enfolds an ML pipeline. The figure highlights the data preparation process, and outlines its interaction mechanisms within the larger ML and DS workflows.

Both the data preparation and modeling steps of the machine learning pipeline are essential for good results [Law17, ALPS22, OMB+24], in terms of model quality as well as time and memory performance. In some conditions, it may be possible to overcome certain types of data imperfections with robust models instead of resorting to data preparation [PDTL20, CAV22, NCA+22]. However, this is not the general case. In practice, it is common for data preparation to occupy the majority of a data scientist or ML practitioner's time, with literature estimating shares of 50-90%, and multiple claims of around 80% [Mun12, BAAW16, SK19, WL20, KDS+24]. The customary manual treatment of both data and model optimization is in fact quite time-consuming, typically warranting

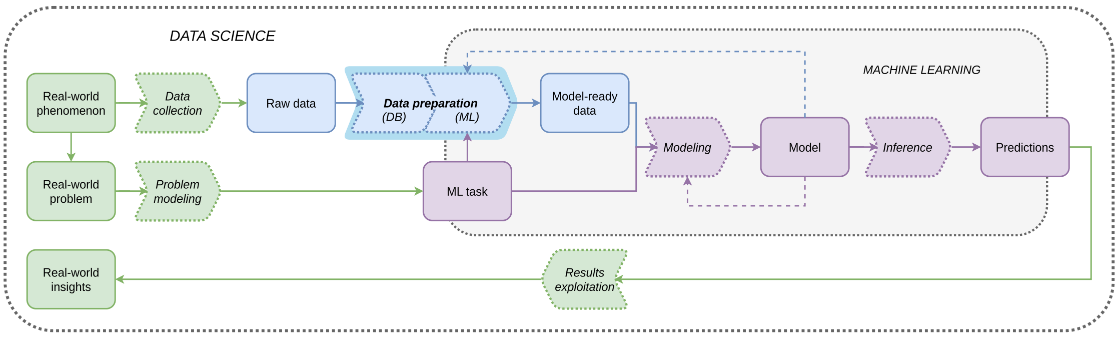

Figure 1: Data preparation within the ML and DS workflows

an iterative trial and error approach by an expert in ML or data science—hence the appeal of AutoML, which reduces the human effort and bias involved in these slow and repetitive tasks, while simultaneously optimizing the performance of the ML system.

Historically, the machine learning field has been dominated by model-centric approaches, concentrated on achieving performance by optimizing algorithms, models, and hyperparameters. While the database community has long focused on improving data systems, relatively little of that work has actually been adopted in ML workflows. Yet, data quality often presents a bottleneck in ML [Sin23, MP24], as models are ultimately only as effective as the data that shapes them [JPN$^+$20, WRSL23, ZBL$^+$23]. This is particularly true in real-world ML applications, where data is typically imperfect in many ways (e.g., inconsistent, noisy, incomplete), thus termed *dirty data* [HS98, KCH$^+$03]. Recognizing not only this setback, but also the potential performance upside of optimizing data [BBP$^+$24, MP24], the rising data-centric AI (DCAI) [JMG23, ZBL$^+$23, JVK$^+$24, KDS$^+$24, ZBL$^+$25] paradigm shifts the focus from refining the model to enhancing the data. By prioritizing data quality, data-centric machine learning research (DMLR) [OMB$^+$24] aims to bring the added effectiveness, efficiency, and reliability of data-centric approaches to ML processes.

Taking a closer look at AutoML research, we observe strong tendencies favoring the modeling phase of the ML pipeline, while the data preparation aspect is mostly overlooked, essentially following the same trend as traditional machine learning [OMB$^+$24]. This imbalance is witnessed by scarce publications and the notable absence of surveys or reviews centered around automating data preparation, in addition to little attention to the topic in AutoML surveys [KSHS$^+$21, HZC21, ZH21, BWL$^+$24, SIS$^+$24, SZW$^+$24]. In fact, leading openly available AutoML systems often necessitate already preprocessed input data or provide minimal data preparation [TWG$^+$19, KPZS20, Kra20, BAI$^+$22], whilst focusing on modeling, which can significantly limit their usability on real-world data [RRK$^+$22].

Moreover, there appears to be a disconnect between claims that AutoML and resulting industry solutions have solved the problem of automating the entirety of the ML workflow, and the reality of remaining data issues and lack of meaningful benchmarks for real applications [Kum21]—thus raising the question of *automating data preparation*, and prompting inquiry into its feasibility, usefulness, and necessity [Pat19].

Objective. In this article, we survey the existing body of knowledge on automating the data preparation process in the context of machine learning, seeking to offer insights regarding scope, relevance, challenges, and methodologies. We lay the groundwork for research on the holistic integration of data preparation into automated machine learning, in an effort to facilitate extending the reach of automation to the complete data science workflow, and bridge the gap between AutoML and data-centric AI.

Scope. We delve into the specifics of data preparation, covering the entire data treatment pipeline between raw source data and fully processed data as it is used for ML model input. While many different terms are frequently employed in literature to designate parts of this process—data wrangling, data engineering, data cleaning, data preprocessing, feature engineering, and several others—their meanings often vary or overlap, hence our use of the term *data preparation* to unify them.

In an overview of underlying data transformations, we direct our attention to the different categories of operations and their value to the overall process. We strive to provide a well-rounded and structured approach with representative examples. We further expand upon the challenges in designing and evaluating data pipelines.

Even though there are multiple data modalities that can be handled by machine learning algorithms, such as text, image, audio, or video data, our study does not consider them all in-depth. We primarily focus on tabular data, as the most common format in data science and classical (Auto)ML. We do, however, note that some recent advances in AutoML incorporate techniques to deal with multimodal data [TFZ$^+$24].

As the core of our work, we survey developments around the topic of automating the data preparation process. In particular, we examine the presence and role of data transformations in existing AutoML systems. Additionally, we explore the numerous and varied approaches to automating data preparation. This includes approaches covering the complete process or specific data aspects, operating in a fully automated manner or aided by human input.

Contributions. To the best of our knowledge, this survey is the first of its kind to focus on data preparation for machine learning as a whole, and specifically on automating it. Our main contributions lie in establishing a thorough background for the complete data preparation process, proposing a taxonomy of the data transformations within, exploring challenges, and providing a review of the current state of research on automating the process—thus creating a connecting thread from fundamentals of the field, through current automation progress, to prospects for further advancement. To be more precise:

1. We initially present a comprehensive background of data preparation in ML and the

relevant context. We highlight the different categories of data transformations and their most common operations, and provide a meaningful structure according to their roles. We assess the challenges in data preparation pipeline design, as well as in their evaluation.

2. To begin investigating automation aspects, after compiling a list of pertinent AutoML systems, we group them them by data pipeline optimization architecture. We deliver an analysis of the automated data preparation elements integrated into each system, and extract common tendencies.

3. We then examine semi-automated data preparation frameworks, taking into account scope, features, and objectives, and categorizing them with regard to their varied approaches. We further derive insights on the feasibility and challenges of automating the whole process.

4. Following that, we provide an overview of fully automated data preparation methodologies, with example systems belonging to each. We review their merits and shortcomings, and present a comparative analysis along relevant criteria for the choice of quality solutions.

5. Lastly, we summarize and discuss our findings, and consider future research directions.

ML Terminology. In order for the paper to be self-contained, we provide definitions of ML-related terms relevant to the context. They can be found in Table 4 in Appendix Section A.

2 Data Preparation: Transformations

In the context of machine learning, data preparation transformations are operations applied to data, with the end goal of enhancing results when combined with a model [Agg15, GLH15, SZ19, Bro20, HN20, FKK$^+$23]. The pool of possible transformations is large, and the relevance of each transformation depends on the data itself, but potentially also the ML task to be performed, and the model employed for the task. To facilitate navigation among these possibilities, we extract a structure based on the aspects of data preparation that each transformation addresses.

2.1 Taxonomy Structure

We propose a taxonomy of data preparation transformations with three levels of categorization: transformation (1) purpose, (2) function, and (3) category. A visual representation of this structure is depicted in Figure 2.

On the highest level, transformations can be characterized by what they aim to achieve, i.e., their *purposes*: data organization, data quality, and model performance [NWC$^+$20].

Rather than always having precise cutoffs, transformation purposes form a spectrum such that transformations can serve multiple purposes to varying degrees. For example, aligning data contents is an organizational task that also results in higher-quality data, which eventually translates to better model predictions. In the interest of clear structuring, we attribute transformations to primary purposes. We also note that each purpose is generally conducive to the next one: good organization is reflected in data quality, and quality tends to boost the results achieved by models.

We further differentiate transformations by their *functions*, the functional facets through which they advance their purposes: data integration, data cleaning, data preprocessing, and feature engineering. Similarly to purposes, there is some overlap between adjacent transformation functions.

Finally, we assign transformations to multiple *categories*, by grouping together different transformation methods that address a common data preparation element—whether resolving a common problem with the data (e.g., different ways to handle missing values), or improving a common aspect of the data (e.g., different ways to generate data points).

Below are descriptions of data preparation purposes, functions, and the categories of transformations enveloped within, along with transformation method examples of each type. We note that this is not a complete catalog of all possible data transformations, particularly if we consider the prospect of creating custom operations optimized for each individual dataset. In practice, transformation ordering often tends to align with the presented order, however this is not a mandatory pattern.

2.2 Data Organization

Organizing data implies shaping it into a format that is well-suited for analytics and further exploitation. For tabular data, this is commonly a single dataset with data points as rows and features as columns. In the case of supervised learning, a target variable column is also included.

2.2.1 Data Integration

Data integration [Len02, ZD07, Doa12], also known as ETL (Extract, Transform, Load), consists in putting together data from different sources and in different formats, into a single coherent dataset in a format that can be input into an ML model and processed by it. Integration concerns both outward formatting and content organization.

Parsing refers to extracting data from a variety of sources and formats and loading their contents into structured datasets.

Examples: importing data from databases, spreadsheets, files in varied formats, into datasets.

Merging designates combining the contents of multiple datasets, with possibly different

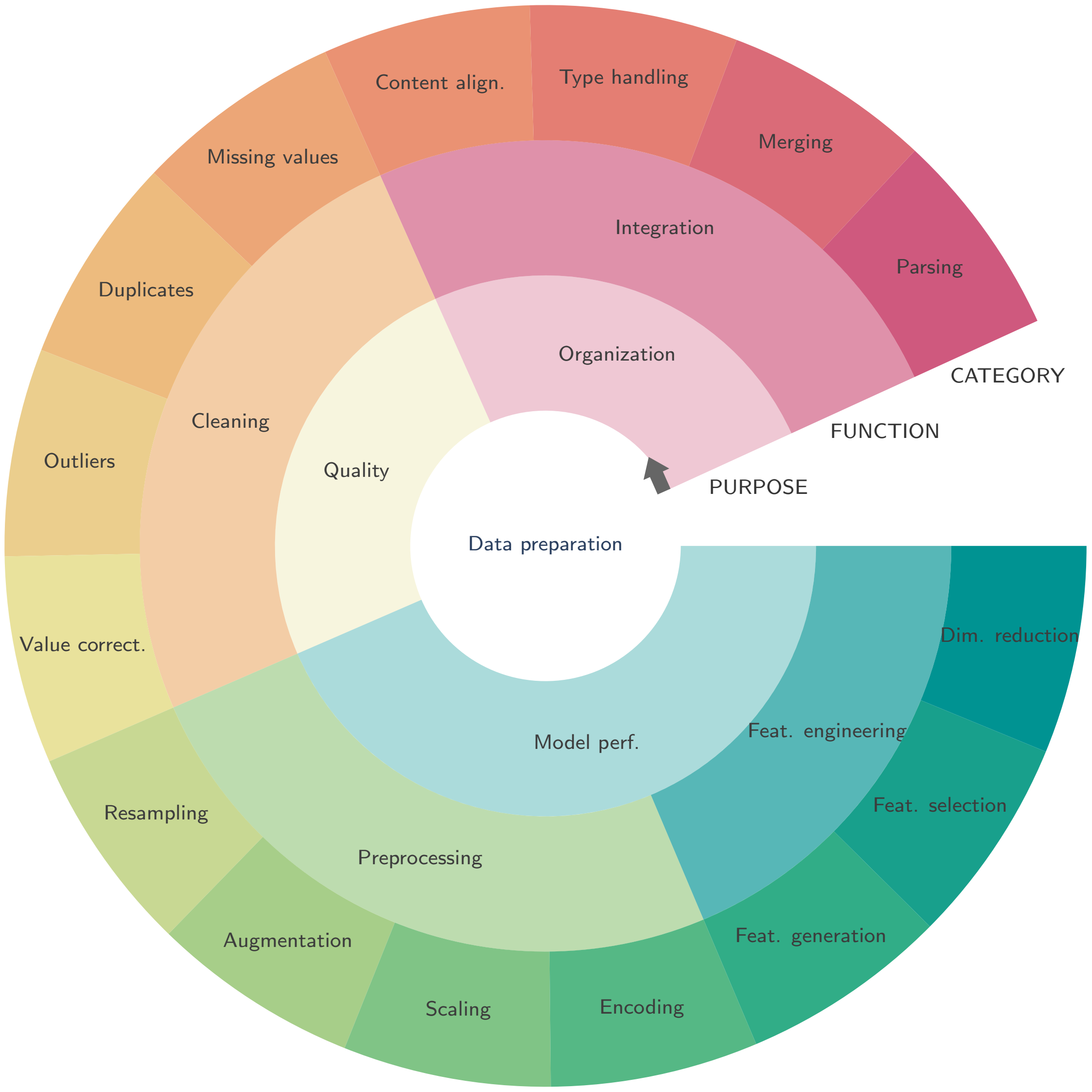

Figure 2: Taxonomy of data preparation transformations

structures, into a single dataset, while recognizing shared features (schema alignment).

Examples: vertical, horizontal, or diagonal concatenation; inner, outer, left, or right join.

Type handling stands for converting and manipulating data types to make sure they are supported and interpretable by data preprocessing operations and ML models.

Examples: casting booleans to categorical or numeric types, casting complex types to more interpretable ones (e.g., dates in string format to date/time or numeric formats), simplifying composite types (e.g., exploding lists of values into separate data points).

Content alignment focuses on handling cases where the same information appears in multiple datasets, columns, or data points, but with inconsistencies or different formatting. Content alignment includes entity resolution [GM12, CEP+20, ZZT+20] (e.g., recognizing "John Smith", "J. Smith", "Smith, John" as the same entity), semantics (e.g., unifying "birth_date" and "DOB" features), and units (e.g., converting km, m, cm to the same unit).

Examples: record linkage [Win14] (linking entities based on shared attributes), fuzzy matching [Nav01] (approximate string matching), unit standardization (e.g., converting all distances to meters).

2.3 Data Quality

High-quality data is crucial for trustworthy analysis, and provides a strong foundation for inference [MBF+25]. Independent of the chosen model, data quality can be improved by making sure the data is complete, consistent (no conflicting information), representative, and interpretable. Data quality sets an upper bound on model performance with regard to reality [LRB+21, WRSL23].

2.3.1 Data Cleaning

Data cleaning [RD00, Das04, CIKW16] targets imperfections in the data contents, such as errors, inconsistencies, or the presence of unnecessary elements; ensuring the data can be used by a model, and raising its general quality by repairing or removing these imperfections.

Missing values refer to completing data by detecting and handling null values in the dataset.

Examples: removing missing values, imputing [LT19] (filling) them using different methods (e.g., forward/backward filling i.e., using the value of the next or previous data point, interpolation, the k-nearest neighbor [Pet09] algorithm), encoding "missing-ness" as a new feature.

Duplicates designate detecting and handling duplicated data points in a dataset, so as to avoid incorrect biases.

Examples: removing duplicates, encoding them as a new feature representing the number of appearances of each data point.

Outliers address detecting and possibly removing outliers, i.e., data points that significantly differ from the rest of the data or their surroundings, and could represent errors or noise.

Examples: detection using standard deviation or IQR (interquartile range) [HA04].

Value correction deals with detecting and repairing incorrect or inconsistent values in the data. It concerns feature value repair [Ber19b] (e.g., spelling errors, out-of-range measurements, or examples such as a mismatched city and postal code), often referred to as record repair in a database context, as well as label correction [FV14, SKP+23] when the error concerns the target variable (wrongly labeled data points).

Examples: rules and constraints (e.g., age must be ≥ 0), outlier value detection (c.f. *Outliers* above) combined with statistical (e.g., mean, mode) or ML-based (e.g., classification, regression) imputation.

2.4 Model Performance

Model performance optimization refers to transforming data in order to improve the outcomes of a particular ML model. The main performance criterion is usually a chosen effectiveness metric for that model (its loss function), but there can also be considerations for efficiency, prompting additional optimization for execution time and resource usage [NAR13, MSK21].

2.4.1 Data Preprocessing

Data preprocessing addresses the distribution and contents of data points in order to make the data more robust and facilitate correct interpretation by a model, raising its performance. Preprocessing makes it possible to leverage the superior efficiency of many ML models when it comes to working with numeric data. Moreover, changing the number of data points can, in addition to balancing out data distribution, also affect time and memory usage due to differences in the amount of data to pass through the model.

Resampling [CPB25, KSA+21, BL21] mitigates issues relating to imbalances in data distribution, typically for classification tasks, via oversampling or undersampling, i.e., strategically adding data points to underrepresented classes (i.e., minority classes) or removing data points from overrepresented classes (i.e., majority classes).

Examples: Oversampling methods: random oversampling (duplicating random data points in minority classes), SMOTE—Synthetic Minority Oversampling Technique [CBHK02] (generating new points via interpolation between minority class data points and their nearest neighbors), ADASYN—Adaptive Synthetic algorithm [HBGL08] (an extension of SMOTE specifically targeting the lowest density areas of minority class data points); Undersampling methods: random undersampling (removing random data points in majority classes), ENN—Edited Nearest Neighbors [Wil72] (removing majority class points located in a neighborhood of predominantly minority class points), Tomek's Links [Tom76] (removing majority class points whose nearest neighbor is a minority class point).

Data augmentation [MM22] corresponds to synthetically generating new data points, by altering or combining feature values from existing ones.

Examples: for tabular data, augmentation can coincide with resampling methods that generate synthetic data points, such as SMOTE and ADASYN mentioned above. There can be more diverse transformations for other kinds of data, e.g., for images: flipping, rotating, cropping; or for text: replacing words with synonyms, deleting some words.

Scaling designates the scaling of numeric features by adjusting their range or distribution, without distorting the differences between values. It brings consistency, ensuring that no feature dominates others due to its magnitude. Feature scaling enhances performance for many kinds of models, and is particularly useful when feature values are interpreted as distances between data points, such as in nearest-neighbor algorithms.

Examples: normalization (proportionally scaling to a fixed range, usually [0, 1]), standardization (centering data so that the mean becomes 0 and the standard deviation 1), robust scaling (similar to standardization, but using the mean and interquartile range) [dCC23].

Encoding [CLVZ11] stands for the encoding of features (e.g., categorical features in the form of strings) to numeric representations (scalars or vectors), so that they can be used by models accepting only numeric features, which are quite common.

Examples: hashing (applying a hash function to create a mapping between categorical and numeric values), one-hot encoding (converting features to binary vectors of their possible values, where 1 indicates the presence of the value, and 0 its absence) [KNH+24].

2.4.2 Feature Engineering

Feature engineering designates the manipulation of features in order to optimize model performance. Curating the selection of features and the information within them can help avoid overfitting the model and improve time performance, while mostly preserving data quality and information. It is also possible to derive added meaning from existing features, improving model results.

Feature generation focuses on generating new features by extracting or transforming information from existing ones.

Examples: Applying numeric operations (e.g., logarithm, square root), discretization [DKS95] (transforming continuous features to discrete ones, often with binning techniques: e.g., uniform, quantile, k-means clustering [Mac67]), extracting temporal data (e.g., the day of the week from a date).

Feature selection refers to reducing the number of features by making a selection of those most relevant to the target variable.

Examples: Forward feature selection (incrementally selecting one at a time), coefficient of variation (ratio of variance to mean) threshold, RFE (recursive feature elimination) decision trees, univariate correlation to the target [LCW+17].

Dimensionality reduction [vdMPv09, SVM14, JSLH22] means reducing the number of features through transforming the data into a lower-dimensional representation, by combining features while retaining essential information.

Examples: PCA—Principal Component Analysis [Pea01, AW10] (a linear technique of projection to a coordinate system where the greatest variance lies along the first axis or principal component, the second greatest variance along the second axis, and so on), t-SNE—t-distributed Stochastic Neighbor Embedding [vdMH08] (a non-linear technique preserving local relationships between data points), UMAP—Uniform Manifold Approximation and Projection [MHSG18] (a non-linear technique preserving both local and global structures).

The transformation categories described above represent a broad overview of data preparation elements that generally support transformations for all data modalities, with our examples mainly focusing on tabular data. Nevertheless, there are also transformation categories specific to certain other data types, which could be considered in a wider context. Examples include image manipulation such as resizing, filtering, or segmentation [SHB93], or NLP (natural language processing) techniques for textual data, such as tokenization, vectorization, or translation methods [PBGN23].

2.5 Data Usability

One additional concept to consider is that of data *usability*. In the context of machine learning, data being usable means that it can be processed by a given ML model. Usability criteria can differ for different models. For instance, some models cannot run on incomplete data, while others can handle missing values; some models can only process numeric data, meanwhile others support a wider variety of types. While not a goal in itself, usability is a requirement for an ML system to function. It is achieved by applying data transformations that address specific incompatibilities—in the previous examples, handling missing values, or encoding certain data types.

2.6 Caveats

While transformations ensuring base compatibility between dataset and model may be indispensable, the contributions of others can be more situational. The varying effectiveness of data transformations in different cases has been pointed out in analyses of some specific transformations. Examples include a study on the diminishing returns of missing value imputation [MV25], or another that shows that balancing data with the SMOTE algorithm is more pertinent for weaker classification models, and less so in the opposite case [EA22].

Another aspect to consider when selecting transformations is their potential effect on information content [KEVDS19]. Information loss most commonly stems from the removal of parts of the data, such as missing out on some fringe cases when removing outliers, or slightly degraded predictions when omitting less relevant features. Altering information content can have deeper effects on data analytics and ML result interpretation, as exemplified in a fairness study [BR21] showing that certain data preparation elements can introduce or enhance biases in the data, and suggesting alternatives to mitigate this effect.

Additionally, explainability can also be affected by data preparation [GZ19, SK24]. Explainability is tightly connected to the transparency of selected data transformations. It can be enhanced by some integration and cleaning operations, but also reduced by non-invertible transformations that render the data more obscure, such as the PCA algorithm or various aggregation functions. Result explainability also depends on the selected ML model [Rud19].

3 Data Preparation: Pipelines

The data preparation process consists in constructing a pipeline of transformations to be applied to the data, as part of the larger machine learning pipeline wherein it accompanies modeling. In this section, we describe how data pipelines are designed and evaluated.

3.1 Pipeline Design

The design of data preparation pipelines is aimed at elevating the quality of the data, in addition to ensuring compatibility and optimizing synergy with a model, all in service of enhancing results. These goals can be achieved by optimizing a data transformation sequence on datasets adequately set up for the task.

3.1.1 Optimization Problem

Some experiments have suggested that, for certain datasets, optimal pipeline design can be model-independent [Que19]. Nevertheless, there is no single best data preparation pipeline that is the same for every possible problem, nor can we truly affirm that one pipeline is *a priori* better than another in isolation. This question has notably been addressed for data integration [Sd04], as well as for learning on imbalanced data [MM21]. It has also been studied for entire pipelines [GBA22].

In actuality, what constitutes the best data preparation pipeline for a given problem is highly dependent upon the data format and contents, the machine learning task to perform, and the model selected for the task. For a given ML algorithm, the main considerations in designing this pipeline are:

1. *Transformation selection*—choosing data transformations to apply and sometimes which features to apply them to;

2. *Transformation ordering*—architecturing the sequence of transformations;

3. *Hyperparameter optimization*—configuring transformation hyperparameter values where applicable.

The pipeline optimization process is illustrated in Figure 3. Handling the three aforementioned objectives allows for much flexibility. A data pipeline can contain multiple instances of the same data transformation, possibly applied to different sets of features and with different hyperparameter configurations. Though data pipelines are commonly structured as linear sequences of transformations (as depicted in the figure), they can also adopt acyclic graph structures [OM16], where some transformations are executed in parallel. In this case, we can talk of a sequence of transformation steps, where each step can consist of one or more simultaneous transformations.

Since the impact of data pipeline choices cannot be assessed without empirical evaluation, we can regard pipeline design as a *black-box optimization problem* [AH17] whose search space lies in the possible combinations along the three enumerated dimensions.

We can formally define the problem as follows. Let:

- D be a dataset (we deliberately consider an open dataset description, as modern ML best practices include multiple options going beyond the basic train-test split, namely when it comes to validation strategies. This is discussed in Section 3.1.3);

- A be a machine learning algorithm with a fixed hyperparameter configuration;

- $\mathcal{T}$ be the set of all possible data transformations, with $T \in \mathcal{T}$ a transformation instance;

- Λ_T be the hyperparameter configuration space of a transformation T, with $\lambda_T \in \Lambda_T$

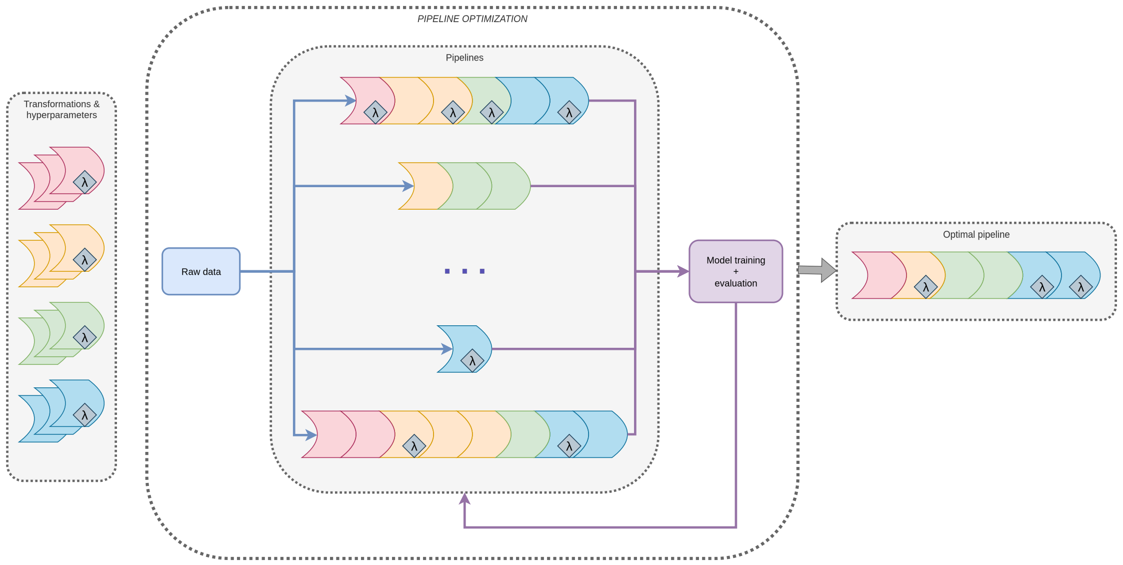
PIPELINE OPTIMIZATION
Transformations & hyperparameters
Pipelines
Raw data
Model training + evaluation
Optimal pipeline

Figure 3: Data preparation pipeline optimization

one specific hyperparameter configuration of T;

- $\mathcal{P}$ be a data preparation pipeline configuration space, with elements $P \in \mathcal{P}$. In a linear pipeline structure, elements are in the form of $P = \big((T_1,\, \lambda_{T_1}),\, (T_2,\, \lambda_{T_2}),\, \dots,\, (T_n,\, \lambda_{T_n})\big)$, $n \in \mathbb{N}$;
- $\mathcal{L}(P, A, D)$ be the loss achieved by a model trained with algorithm A, on dataset D transformed by a pipeline P.

With $\mathcal{L}$ serving as an estimate of the model generalization loss, the optimal solution to our data pipeline optimization problem lies in solving Equation (1):

$$P^* \in \arg\min_{P \in \mathcal{P}} \; \mathcal{L}(P, A, D) \tag{1}$$

Data pipeline optimization can also be considered in combination with the *CASH* (Combined Algorithm Selection and Hyperparameter Optimization) [THHL13] problem, which is central to AutoML. The search space is then extended to combinations of data pipelines, ML algorithms, and algorithm hyperparameters. In addition to the previous problem setup, let:

- $\bar{\mathcal{A}}$ be the set of possible ML algorithms, with $\bar{A} \in \bar{\mathcal{A}}$ one (unconfigured) algorithm;
- $\Lambda_{\bar{A}}$ be the hyperparameter configuration space of algorithm $\bar{A}$, with $\lambda_{\bar{A}} \in \Lambda_{\bar{A}}$ one specific hyperparameter configuration of $\bar{A}$;
- $\mathcal{A}$ be the set of possible ML algorithms and implicitly their possible configurations,

where each element $A \in \mathcal{A}$ has the form of $A = (\bar{A}, \lambda_{\bar{A}})$

To include ML algorithm selection and associated hyperparameter optimization, Equation (1) is expanded into Equation (2):

$$(P,\, A)^* \in \arg \min_{P \in \mathcal{P},\, A \in \mathcal{A}} \mathcal{L}(P, A, D) \tag{2}$$

3.1.2 Experience and domain insights

Beyond exploring the data pipeline search space, data preparation can be facilitated by learning from experience with different datasets and pipelines, as well as by leveraging domain knowledge, which implies knowledge about the data's meaning, structure, and context. When available, these two kinds of insights allow for informed decision-making rather than operating in a purely black-box environment, leading to more efficient search space navigation and faster convergence towards an optimal solution.

3.1.3 Dataset, Training, and Testing Considerations

As with the modeling process, where an ML model is trained on a training dataset and evaluated on a test dataset, data pipeline optimization is also done with the help of the training set. The same sequence of transformations is later applied to the test set before evaluating the model on it, with a few exceptions—transformations that risk skewing the evaluation process, such as for instance resampling transformations, which can distort the data distribution by adding or removing data points, are not transferred to the test set. Additionally, the inclusion of some transformations in the pipeline requires that their inverses be applied before evaluation. For example, if the target variable is encoded during preprocessing, comparisons with (unencoded) ground truth will fail—the variable therefore needs to be decoded. The process for any other non-training data (such as a validation set or newly incoming data) is the same as that of the test set.

Another point of importance with regard to the train-test mechanism is the process of separating a dataset into training and test data. In a controlled environment, the two sets should be disjoint, however in real-world conditions some overlap can occur. In order to enable robust learning, so that patterns learned on the training set can be relevant to a previously unseen test set, the two datasets should be sampled from the same underlying data distribution in a representative way. Despite the most frequent dataset splitting method being the random split—randomly assigning data points to the two sets while only considering their proportional sizes—distribution similarity between datasets can be promoted through the use of strategic sampling methods, such as k-fold cross-validation [Sto74, RTL09]. A well-stratified and robust data split increases the odds of finding an optimal data preparation pipeline in terms of generalization.

In addition to stratified sampling, ML processes often implement strategies to mitigate the phenomenon of overfitting [Die95]. In the context of data pipelines, overfitting occurs if the pipeline design or its inner operations fit too closely to training data specifics, for instance capturing noise, random fluctuations, or artifacts—which leads to poor generalization to new, unseen data. Therefore, a common ML practice is to split the data into three datasets: (1) the training set is used to fit the pipeline (e.g., hyperparameters), (2) the validation set is used to provide feedback to the training process with regard to pipeline generalization performance (e.g., using AutoML), and, since repeated evaluation on the validation data can also lead to overfitting of the pipeline design, (3) the test set is used on the final optimized data pipeline to obtain a real, unbiased, estimate of its generalization capabilities to new data. For both levels of splits (i.e., training-validation and validation-test), different approaches can be employed, such as hold-out splits, cross-validation, or other (nested) resampling strategies, typically depending on the chance of overfitting and the cost of doing evaluations.

3.2 Pipeline Evaluation

Evaluating data preparation pipelines in a meaningful and objective way is quite challenging [Kum21, ALPS22]. As of yet, there are no standard benchmarks, methods, or metrics specific to it. Instead, impact is usually quantified indirectly, through assessments of data quality and model performance.

3.2.1 Evaluation through Data Quality

Data quality is of paramount importance when it comes to achieving reliable ML results that adequately represent reality [MBF+25]. Evaluating data pipelines through the lens of data quality implies comparing data quality before and after the application of a data preparation pipeline. However, defining and measuring data quality is in itself a topic of discussion [PLW02, BCFM09, Law17, JPN+20, SCvdS22]. Identifying quality facets to consider and ways in which to quantify them are non-trivial problems.

In addition, data quality issues can be domain-specific [FFR22], which adds a layer of complexity to the treatment of such data. For instance, in the case of outliers, without domain knowledge it can be hard to tell whether an outlying point is an error, or actually constitutes a relevant data point.

Moreover, not every data quality aspect or data transformation necessarily influences the performance of every model in the same way. For example, in classification over unevenly distributed data, a neural network classifier can significantly improve after resampling the data to balance out its distribution [CPB25], while random forest models tend to be more robust and may not need this kind of preprocessing to achieve similar results [KGH07, DKN15].

These kinds of inconsistencies with regard to evaluating data across different models call for consideration of the selected model in the evaluation process of data preparation pipelines.

3.2.2 Evaluation through Model Performance

The most common way to evaluate data preparation is through downstream model performance, i.e., assessing model loss after training on data transformed by a given pipeline. While this is certainly a pertinent indicator, it is in fact a measure of how well the model fits the data, rather than how well the model fits reality. Equating strong model performance with good data preparation can be risky due to questions of data quality and the possibility of underlying biases.

In order for results to be pertinent, it is also particularly important for the model loss metric to be carefully selected so as to be adequate for the problem. Even then, model performance may not be entirely reliable, since its score can increase from overfitting. In actuality, this makes the model worse, though the effect can be mitigated through validation strategies, e.g., cross-validation [Sto74, RTL09], or robustness techniques such as *early stopping* or *regularization* [SHK+14, Yin19].

This method of evaluating data preparation by measuring model performance is therefore directly dependent on data quality and the robustness of the model evaluation. It is also quite costly, since it requires a model to be trained on every pipeline. However, this remains the most accessible and dependable evaluation method to date.

3.2.3 Transformation Interactions

If we consider individual or groups of transformations within a pipeline, evaluation is additionally complicated by interactions between data transformations: applying one transformation can affect the behavior of a subsequent one. For example, removing outliers can change the span of values of a feature, and consequently the effectiveness of normalization. This makes it very difficult to measure the impact of a single transformation in isolation, or to identify which parts of the pipeline contribute to performance improvement or degradation, though there are some promising beginnings in this direction [GZ19, SK24].

3.2.4 Benchmarks, Tools, other Evaluation Methods

Setting aside methodology, we also encounter a lack of standardized benchmarks for data pipeline evaluation [OMB+24]. Unlike certain ML subfields that benefit from well-known datasets conventionally used for their model-centric benchmarks, such as *ImageNet* [DDS+09], *CIFAR-10/100* [LBBH98] and *MNIST* [Kri09] for image classification, or *GLUE* [WSM+19] for natural language processing, the domain of data preparation, or more broadly that of data-centric ML, has yet to establish a consensus on benchmarking criteria and datasets [OMB+24].

Nonetheless, some recent works have begun to tackle various aspects of evaluating data preparation pipelines specifically. One proposition is an automated *provenance-based screening* [SGG24] method, modeling data preparation pipelines into *dataflows* used to detect certain types of mistakes and generate metadata information. The *ML Data Prep Zoo* [SK19] presents a repository of data preparation tasks, labeled benchmark datasets, and pre-trained ML models, providing the tools to create and evaluate automated solutions to various data preparation tasks. The GouDa [RBCS22] tool automatically generates flawed datasets that, along with their ground truth, facilitate the analysis and evaluation of data preparation pipelines. Dcbench [EKR+22] is a benchmark for the evaluation of select components of data-centric AI systems. DataPerf [MBY+23] provides another collection of data-centric algorithm benchmarks for multiple ML tasks, as well as a community platform allowing new benchmarks to be added. Additional benchmarking datasets can be found on the *OpenML* [VvRBT14, BCD+25] platform. Such methods, tools, and collections pave the way towards a better understanding and a more structured approach to pipeline design and evaluation.

4 Automated Data Preparation in AutoML

In order to assess the state of data preparation in automated machine learning (AutoML), we explore the data transformation categories implemented by different AutoML solutions, as well as the decision processes in charge of handling them.

4.1 Overview of Relevant AutoML Systems

We compile a collection of AutoML systems with support for tabular data relevant to this survey according to the following criteria:

(1) Providing end-to-end automation. This excludes approaches that are not fully automated and require human interaction. Notable systems eliminated by this criterion include: Alpine meadow [Kra18, SZB+19], AutoML-DSGE [ALRM20], Autostacker [CWM+18], Auto_ViML [fea19], Auto tune models (ATM) [SDC+17], dabl [dab16], Ludwig [MDM19], MOSAIC [RSS19], Neural Network Intelligence (NNI) [Mic18b], REsilient ClassifIcation Pipeline Evolution (RECIPE) [dPOP17], TransmogrifAI [Sal17].

(2) Covering the whole ML pipeline or at minimum the whole modeling phase. This excludes approaches addressing only inner ML components or parts of the pipeline that do not interact with data preparation directly, e.g., neural network search (NAS) systems such as NASLib [RZS+20], Katib [ZVP+19], Hypernets [YLW20].

(3) Being open source and providing public documentation with enough information for our study. This ensures that our information about the considered systems is reliable and relatively easily verifiable. This selection notably excludes closed-source commercial systems from some of the biggest industry players in AI and AutoML: Google Cloud AutoML [Goo18], DataRobot AI Cloud [Dat15], H2O.ai Driverless AI [H2O20], Microsoft Azure Machine Learning AutoML [Mic18a], Amazon SageMaker Autopilot [Ama19], Oracle AutoML [YMM+20]. It is, however, worth noting that these typically do contain

data preparation elements. Published research solutions lacking supporting code, such as Alternating Direction Method of Multipliers (ADMM) AutoML [LRV$^+$20] and Auto-Compete [TK15], are also excluded.

(4) Being reasonably usable (projects that are still maintained, or left in a stable state). This eliminates deprecated, outdated or unmaintained solutions.

Below are the AutoML systems that we retain for consideration. We categorize them according to the way in which they handle data preparation: separating those that perform preset or rule-based data treatment—systems with static data preparation, and those whose optimization algorithm also covers data preparation pipelines—systems with optimized data pipelines. We do not differentiate systems by the ML tasks they target or other modeling factors, keeping our focus on data preparation aspects instead.

4.1.1 Static Data Preparation Systems

Oboe (& TensorOboe) [YAKU19, YFWU20] is a meta-learning-based [Sch87, HAMS21] AutoML system using collaborative filtering [GNOT92, SKKR01] to predict model performance under resource constraints; TensorOboe extends the method with tensor completion [LMWY13] and adds more robustness with regard to data quality.

AutoML-Zero [RLSL20] is a research prototype exploring the evolution [BBBMM14] of complete machine learning algorithms from a set of basic mathematical operations, i.e., "from zero".

FLAML (Fast and Lightweight AutoML) [WWWZ21] is an AutoML library designed for efficiency, using lightweight learners and resource-aware tuning without heavy search. It supports integration with the *Fabric* data platform, as well as the MLOps platform *MLflow*.

AutoKeras [JCSH23] is an AutoML system based on the Keras [Cho15] deep learning library, which performs neural architecture search and model tuning on image, text, and multimodal data. As of today, tabular data is no longer included in the official module, though it is available through an extension [DB17].

H2O AutoML [LP20] is an enterprise-ready, scalable AutoML solution using ensembling [Die00], and offering model interpretability and integration with the *H2O* analytics platform.

BlueCast [Tho23] is a fast and lightweight AutoML library mainly featuring XG-Boost [CG16] models. It also includes optional user customization and a data toolkit for more advanced tasks, as well as explainability features.

Auto_ml [Par16] is a simplified AutoML package designed for production. It allows automatic training of multiple models of the same kind based on a chosen data split (e.g., a separate model for every country from worldwide data).

MLJAR-supervised [MLJ18] is an AutoML tool with different modes for fast prototyping, explainability, and deployment-ready outputs.

TabPFN [HMEH22, HMP+25] is a foundation model [BHA+22] for small datasets that near-instantly produces highly accurate predictions, without the need for hyperparameter tuning.

AutoGluon [EMS+20, STE+23, TFZ+24] is a versatile AutoML solution supporting tabular (AutoGluon-Tabular), time series (AutoGluon-TimeSeries), multimodal with image and text (AutoMM) data, and performing automatic ensembling and fine-tuning of foundation models.

Naïve AutoML [MW22] is a naïve AutoML tool useful for baseline comparisons, that combines basic preprocessing with random search. Its efficiency lies in optimizing each pipeline component in isolation, thus significantly reducing its search space.

MLBox [de 17] is an AutoML library focused on data drift detection [AFR+20], preprocessing, and model optimization.

LightAutoML [VRS+22] is an AutoML library optimized for performance, interpretability, and production deployment.

ML.NET AutoML [AAB+19] is a framework for .NET developers, which provides seamless integration with .NET applications. It supports tabular, text, and image data.

4.1.2 Optimized Data Pipeline Systems

Auto-WEKA (& 2.0) [THHL13, KTH+17] is an extension of the WEKA [HDW94] data platform that automates algorithm and hyperparameter selection using Bayesian optimization [SLA12, Gar22]; v2.0 features multi-objective search.

HyperOPT-sklearn [KBE14] is a wrapper for scikit-learn [PPV+11] models with Hyperopt-based [BYC13] Bayesian optimization for algorithm and hyperparameter optimization.

TPOT (The Tree-based Pipeline Optimisation Tool) [OM16, GVO17, PNJK19] is an AutoML system that uses genetic programming [BBBMM14] to evolve ML pipelines; Layered-TPOT and TEAPOT-SH variants improve efficiency via hierarchical search (evaluating pipelines on increasingly large data subsets) and successive halving [JT16].

EDCA (Evolutionary Data-Centric AutoML) [SC25] is an efficient AutoML solution that optimizes the entire ML pipeline. It performs data analysis, followed by the selection and optimization of data preparation steps and a model, using a genetic algorithm.

ML-Plan (& ML2-Plan) [MWH18, WMTH19] is a hierarchical planning-based [EHN94] AutoML system for constructing ML pipelines; ML2-Plan adapts the method for multi-label data.

DeepLine [HVKR20] is an AutoML tool that relies on meta-learning, deep reinforcement learning (RL) [KLM96, Sut98] and *hierarchical actions filtering* to generate machine learning pipelines.

AlphaD3M [LLR+23] is an AutoML library that automates data pipeline and model optimization by combining deep reinforcement learning and meta-learning. It is part of a broader project (D3M) aiming to automate data science.

Auto-sklearn (& 2.0) [FKE+15, FEF+22] is a scikit-learn-based AutoML system using Bayesian optimization via SMAC [LEF+22] and meta-learning; v2.0 improves efficiency with early stopping via successive halving and with task-specific portfolios (predefined pipelines).

GAMA (General Automated Machine Learning Assistant) [GV21] is an AutoML tool using genetic algorithms to evolve and ensemble machine learning pipelines asynchronously.

Auto-PyTorch [ZLH21] is an AutoML toolkit for deep learning using PyTorch [PGM+19], automating neural architecture and hyperparameter search for tabular, time series, and image data, by using Bayesian optimization, multi-fidelity [FSK07, KFB+17, FKH18], meta-learning, and ensemble construction.

FEDOT [NVS+22] is an AutoML framework employing meta-heuristic evolutionary methods for learning and optimization on graph-based pipelines. FEDOT can handle tabular and multimodal data, with text, image, and time series.

4.2 Analysis

We proceed to an analysis of data preparation in the selected AutoML systems, supported by a comparative table. Alternative comparisons of a similar kind can be found in some earlier AutoML studies [TWG+19, Kra20, BAI+22], but covering different (smaller) sets of transformation categories and AutoML systems.

4.2.1 Setup

Table 1 presents a breakdown of automated data preparation elements that are implemented in our list of AutoML systems, organized according to transformation categories, functions, and purposes, as described in our taxonomy in Section 2. In the last column, titled *DP Optimization*, we differentiate between systems with data treatments that are fixed or arbitrarily decided, and ones that optimize a data preparation pipeline along with the model. We order the systems based on (1) whether they perform optimized data preparation, (2) the number of transformation categories they cover, and (3) their publication date.

We consider only explicit data preparation operations, as opposed to ones implicitly performed by certain models in the model selection pool of an AutoML system (e.g., some tree models inwardly doing their own feature selection). The information in the table reflects the state of each AutoML system to the best of our knowledge, according to the contents of their corresponding research papers, documentation, and code.

4.2.2 Findings

Table 1 demonstrates that, barring a few exceptions, automated data preparation in AutoML systems today is by and large focused on model performance optimization, in line with the traditional model-centric trend in ML and AutoML in general.

The most frequently present transformations are feature engineering ones (feature generation, feature selection, and dimensionality reduction) and feature-oriented preprocessing ones (scaling and encoding), emphasizing the prioritization of model performance, via features in particular. The only other consistently represented transformation categories are the cleaning of missing values, which can completely obstruct the function of many models, and sometimes data type handling. These indicate added consideration for the most common obstacles to a functioning model.

There is a notable lack of transformations centered around data quality, excepting the cleaning of missing values. Likewise, more complex data integration (dataset merging, content alignment) going beyond the minimum of loading and interpreting data types, is almost entirely absent. Preprocessing transformations dealing with data points as opposed to features are also underrepresented. This reveals a tendency to rely on the model's capabilities to overcome potential data issues such as noise, inconsistencies, or imbalance, while only treating data as far as making it viable for use.

From a methodology point of view, the table exhibits a mix of AutoML systems that include data preparation in their optimization process, and those that apply preset transformations or follow a rule-based logic. Research-oriented solutions appear to lean towards optimization more than their production-oriented counterparts, potentially leading to questions about the theoretical versus practical efficiency and effectiveness of the two approaches.

5 Semi-Automated Data Preparation

Semi-automated data preparation encompasses approaches that partially automate the data pipeline or fully automate a particular part of it. While some aspects of data preparation can be automated, it has been suggested that others require human intervention [DBDRH$^+$22]. Beyond expanding upon the set of transformation categories that can be automated, as seen in Section 4.2, we further seek to gain a better understanding of the importance of human contributions in the data preparation process, and whether the human component presents an obstacle to fully automating the pipeline.

Transformation category: / *System:*	Parsing	Merging	Type handling	Content align.	Missing values	Duplicates	Outliers	Value correct.	Resampling	Augmentation	Scaling	Encoding	Feat. generation	Feat. selection	Dim. reduction	**DP Optimization**
Oboe	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
AutoML-Zero	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
FLAML	-	-	-	-	✓	-	-	-	-	-	✓	✓	-	-	-	-
AutoKeras	-	-	-	-	-	-	-	-	-	✓	✓	-	-	-	✓	-
H2O AutoML	-	-	✓	-	✓	-	-	-	-	-	✓	✓	-	-	-	-
BlueCast	-	-	✓	-	✓	-	-	-	-	-	-	✓	-	✓	-	-
Auto_ml	-	-	✓	-	-	-	-	-	-	-	✓	✓	✓	✓	-	-
MLJAR-supervised	-	-	-	-	✓	-	-	-	-	-	✓	✓	✓	✓	-	-
TabPFN	-	-	-	-	✓	-	-	-	-	-	✓	✓	✓	✓	-	-
AutoGluon	✓	-	✓	-	✓	-	-	-	-	✓	-	✓	✓	-	-	-
Naïve AutoML	-	-	-	-	✓	-	-	-	-	-	✓	✓	✓	✓	✓	-
MLBox	✓	-	✓	-	✓	✓	-	-	-	-	-	✓	✓	✓	-	-
LightAutoML	-	-	✓	-	✓	-	-	-	-	-	✓	✓	✓	✓	✓	-
ML.NET AutoML	✓	-	✓	-	✓	-	-	-	-	-	✓	✓	✓	✓	✓	-
Auto-WEKA	-	-	-	-	-	-	-	-	-	-	-	-	-	✓	-	✓
HyperOPT-sklearn	-	-	-	-	-	-	-	-	-	-	✓	✓	✓	-	✓	✓
TPOT	-	-	-	-	-	-	-	-	-	-	✓	-	✓	✓	✓	✓
EDCA	-	-	-	-	✓	-	-	-	✓	✓	✓	✓	-	✓	-	✓
ML-Plan	-	-	-	-	✓	-	-	-	-	-	✓	✓	✓	✓	✓	✓
DeepLine	-	-	-	-	✓	-	-	-	-	-	✓	✓	✓	✓	✓	✓
AlphaD3M	✓	-	-	-	✓	-	-	-	-	-	✓	✓	✓	✓	✓	✓
Auto-sklearn	-	-	-	-	✓	-	-	-	✓	-	✓	✓	✓	✓	✓	✓
GAMA	✓	-	✓	-	✓	-	-	-	-	-	✓	✓	✓	✓	✓	✓
Auto-PyTorch	-	-	✓	-	✓	-	-	-	-	✓	✓	✓	✓	✓	✓	✓
FEDOT	✓	✓	✓	-	✓	-	✓	-	✓	-	✓	✓	✓	✓	✓	✓
Function:	Integration				Cleaning				Preprocessing				Feat. engi.			
Purpose:	Organization				Quality				Model perf.							

Table 1: Automated data preparation in AutoML systems,
ordered by data preparation optimization, transformation categories, and publication date

In the following subsections, we provide an overview of various semi-automated methodologies, structured according to the different ways in which they integrate automated components, and the role of the human within their workflows. Though not all of these approaches are aimed specifically at machine learning applications, they can nonetheless deliver significant value in an ML context.

As a great many semi-automated tools and approaches exist in literature, it is quite possible that our collection may not be exhaustive. However, it provides a comprehensive overview, from well-known frameworks to newer methods, with a variety of semi-automation approaches which, put together, cover virtually all the data preparation categories presented in Section 2.

5.1 Full Automation of Parts of the Pipeline

The tools and frameworks below are specialized in automatically treating select parts of the data preparation pipeline.

ColNet [CJHS19] automatically annotates table columns with semantic types by predicting them based on column contents and a knowledge base, using convolutional neural networks (CNN) [LBH15] along with semantic embeddings [CP18].

UniDM (a Unified framework for Data Manipulation) [QHZ+24, LDZ25] leverages large language models (LLM) [VSP+17, MMN+25] for various data preparation tasks. It breaks automated data manipulation down into three steps with prompts specifically designed for them: context retrieval to capture knowledge from data lakes, context data parsing to reshape the knowledge into natural language, and target prompt construction to design and effective LLM prompt for the data preparation task at hand.

Skrub [skr18] is a Python library that aims to "directly connect database tables to machine learning estimators". It offers advanced automated tools that smartly handle multiple complex data transformation categories, going beyond the basic transformations found in more typical data preparation solutions. The library is still under development, with additions of new submodules for a wider range of functionalities.

ExploreKit [KSS16] performs automatic feature generation. It generates a large number of features by combining existing ones, then uses its ML-based feature selection process to predict the usefulness of the new features and to select the best ones.

AutoLearn [KMP17] is a regression-based feature learning algorithm for automated feature generation and selection. It analyzes pairwise feature relationships, then generates new features based on discovered insights, and finally makes a selection by finding the best predictors via regression.

TSFresh [CBNK18] is a Python library for automatic time series feature extraction and selection. It generates features using pre-defined time series characterization methods, then makes a selection via hypothesis testing, using p-value calculation with multiple testing.

FeatLLM [HYAP24] is a LLM-based system that performs automated feature engineering. It makes use of LLM capabilities to generates prompts for feature generation augmented by domain knowledge, as well as for informed feature selection.

5.2 Human-in-the-Loop Approaches

Human-in-the-loop approaches refer to ones that include both automated processes and the involvement of a human user. The methods in this category can be subdivided into single-interaction systems, where the user only intervenes at the beginning of the process while the rest of it is fully automated, and systems that operate in an interactive loop of human and automated elements, constructing their pipeline in an iterative process.

5.2.1 Single Interaction

The methods below enable automated data preparation with the help of a single human intervention that provides insights into the data or examples of data treatment to learn from. As such, they can also be classified as human-guided automation methods.

FlashExtract [LG14] automatically extracts raw data from text files, webpages, spreadsheets. Using human-generated examples, it synthesizes extraction scripts in a domain-specific language.

The *Deep Feature Synthesis* [KV15] algorithm and the supporting FeatureTools tool are geared towards automated feature creation. They extract feature tables from relational database-type structures.

SampleClean [KWF+15] proposes a method consisting in manually cleaning only a sample of database data, then using it to automatically estimate aggregate query results on dirty data. It is accompanied by ActiveClean [KFG+16], which extends the concept to some ML models, by selecting data to prioritize cleaning for during training. BoostClean [KFGW17] takes this a step further, identifying both the data to clean and a cleaning method for it, using statistical boosting [Sch90, Sch03] and Word2Vec word embeddings [MCCD13]. Lastly, AlphaClean [KW19] fully automates the design and parameter configuration of entire data cleaning pipelines.

Data programming [RDSW+16] presents a paradigm for creating and modeling labeled training datasets. For an unlabeled or partially labeled dataset, users encode weak supervision by way of *labeling functions*, which each provide a label for a subset of the data. The collection of labeling functions forms a generative model, allowing the dataset to be denoised through making decisions on overlapping labels, by learning the accuracies and correlation structure of the labeling functions.

HoloClean [RCIR17] is a data cleaning framework for value correction using probabilistic inference [Nea93, Box11]. It relies on human input in the form of constraints and metadata, and can also accept feedback on its data repairs.

CPClean [KLW+20] performs data cleaning with the aid of a human who provides ground truth information. Its algorithm is rooted in the concept of *Certain Predictions*, aiming to correct the data in such a way that any predictor would give the same prediction results.

Raha [MACF+19] and Baran [MA20] are tools for value error detection and correction, respectively. The detection system is configuration-free and relies on the generation of feature vectors to cover various types of data errors. The correction system uses transfer learning for multiple error correction, with the aid of context-aware data representation. The systems are highly automated, requiring only a few initial user-provided examples.

HAIPipe [CTF+23] is a system combining human-generated pipelines (HI-pipelines), designed with domain knowledge and experience, and machine-generated ones (AI-pipelines), which are search-based and optimized, into a best of both worlds solution. It uses an *enumeration-sampling strategy* to find the best combined pipeline: enumerating possible pipelines and using active learning [CAL94] to select the best-performing one.

5.2.2 Interactive Loop: Human-Guided Automation

An interactive loop implies back-and-forth exchanges between human and automation in the pipeline optimization process.

Interactive approaches also present multiple examples of human-guided automation, where the design of the data pipeline is mostly automated, but with the guidance of a human user. The user can contribute in different ways depending on the system, such as analyzing the data, highlighting areas that require attention, or providing feedback on results.

Feedback-driven improvement of data preparation pipelines [KP20] is an approach where the user guides an otherwise automatic data preparation process, by providing feedback regarding resulting data after transformations are applied. The provided feedback represents correctness criteria.

Rain [WFWW20] is a complaint-driven training data debugging system. It uses human feedback on dataset queries to resolve issues and improve dataset contents through heuristic approaches based on influence functions [KL17].

Cleaning for ML rather than before ML [NCA+21] is an interactive architecture proposal for cleaning data during the training process using model feedback, instead of cleaning independently before training. While the cleaning workflow is automatically generated and evaluated over time, the user is integrated into the loop to oversee the process and provide signals aiding decision-making, such as annotating errors or certain properties.

AlpineMeadow [Kra18, SZB$^+$19] is an interactive AutoML tool providing a user-guided way of curating ML pipelines in an effort to simulate a data scientist's process. The system combines query optimization ideas and pipeline optimization methods such as Multi-Armed Bandits [Rob52, Sli19], Bayesian optimization [SLA12, Gar22], and meta-learning [Sch87, HAMS21], with human input in the form of feedback at every step.

The Automatic Statistician [SSJ$^+$19] presents a vision for automated data science frameworks with minimal human intervention. Beginning at raw datasets, the concept covers data preparation, modeling, evaluation and insights. It also includes considerations for explainability and resource budget. Interactions with users in natural language are envisioned through automatically generated reports. The user guides the system by indicating certain preferences such as model and evaluation methods, areas of interest in the data, or resource constraints. Every other aspect of the process is automated.

Learn2Clean+HIL (Learn2Clean with Human-In-The-Loop) [Ber20] is an extension of the automated reinforcement-learning-based [KLM96, Sut98] data preparation system Learn2Clean [Ber19a]. In this variant, a human user is integrated into the process in order to actively give regular feedback to the system through a manual mechanism for RL rewards.

5.2.3 Interactive Loop: Human Assisted by Automation

Another kind of approach interactively integrating human and automation is one where the roles are reversed: a user constructs a data preparation pipeline with the assistance of an automated system. In this format, the automation usually includes a preliminary data analysis or an optimization process, in order to provide recommendations to the user that help build or improve the data pipeline.

Data Diff [SHGC18] performs automated data wrangling on data that is periodically resampled, by detecting certain types of differences (inconsistencies/corruptions) and *patching* (fixing) them using transformations in a domain-specific language. If an automatic patch is not feasible, it informs the user.

Auto-Prep [BAI$^+$22] is a partially automated data preparation solution that detects the data problem (ML task), generates visualizations, and makes data cleaning and preparation recommendations. The user then performs actions based on those insights.

Predictive Interaction [HHK15] is a framework design that provides data visualizations, lets the user choose features of interest from them, and then uses predictive methods to suggest data transformations accordingly. This process is repeated in a loop where the system provides guidance while the user makes decisions.

ChatPipe [CLJ$^+$24] allows a user to design a data preparation pipeline through interactions with the *ChatGPT* [RNSS18, BMR$^+$20, OAA$^+$24] LLM. The system creates ChatGPT prompts based on the problem and wishes expressed by the user, who in turn receives recommendations for subsequent data preparation transformations, as well as automatically generated code.

Another LLM-based approach is that of interactively improving data preparation code by automatically generating *shadow pipelines* [GGS24]. It consists in generating variants of an original pipeline that detect issues, try improvement modifications, and suggest and explain them to the user, using LLMs. The user can then choose to integrate these changes.

5.3 Approach Summary and Pipeline Coverage

Similarly to the setup in the previous section, Table 2 reflects the data preparation coverage of the semi-automated systems above, relative to the transformation categories, functions, and purposes described in the taxonomy in Section 2. Three additional columns are included to illustrate the variety in approaches, outlining which tasks are undertaken by humans and which by automation, and how the two interact. The systems are ordered by (1) semi-automation approach, (2) transformation categories, and (3) publication date.

The *Guidance* column indicates whether the human or the automation (or both/neither) provides guidance to their counterpart within the data preparation process. This assistance can come in different forms, such as offering examples, data insights, or transformation recommendations.

The *Action-taking* column indicates whether the human or the automation (or either) has the final say in terms of making decisions and performing transformations on the data.

The *Interactive loop* column indicates whether the system involves an iterative back-and-forth process between human and automation while constructing a data preparation pipeline—as opposed to a fully automated approach, or a one-shot interactive approach where one actor (human or automation) participates at the beginning of the process, and the other actor then takes over without further interactions between the two.

5.3.1 Pipeline Coverage Observations

Regarding explicit pipeline coverage in terms of data transformation categories, we note that, with a few exceptions, the systems that actually implement transformations tend to concentrate on a single transformation function. On the one hand, we have tools from the database community centered around data integration or cleaning (mostly value correction), and on the other hand, tools focusing on feature engineering (feature generation and selection) from more ML-oriented works.

Looking back at the transformation categories covered by AutoML solutions (c.f. Section 4.2), where we saw that data preparation in AutoML was majorly centered around feature transformations, it is interesting to find that many semi-automated systems that

Transformation category: *System:*	Parsing	Merging	Type handling	Content align.	Missing values	Duplicates	Outliers	Value correct.	Resampling	Augmentation	Scaling	Encoding	Feat. generation	Feat. selection	Dim. reduction	**Guidance**	**Action-taking**	**Interactive loop**
ColNet	-	-	-	✓	-	-	-	-	-	-	-	-	-	-	-	-	🤖	-
UniDM	-	-	✓	✓	✓	-	-	✓	-	-	-	-	-	-	-	-	🤖	-
Skrub	-	✓	✓	✓	✓	-	-	✓	-	✓	-	✓	-	-	-	-	🤖	-
ExploreKit	-	-	-	-	-	-	-	-	-	-	-	-	✓	✓	-	-	🤖	-
AutoLearn	-	-	-	-	-	-	-	-	-	-	-	-	✓	✓	-	-	🤖	-
TSFresh	-	-	-	-	-	-	-	-	-	-	-	-	✓	✓	-	-	🤖	-
FeatLLM	-	-	-	-	-	-	-	-	-	-	-	-	✓	✓	-	-	🤖	-
FlashExtract	✓	-	-	-	-	-	-	-	-	-	-	-	-	-	-	👤	🤖	-
FeatureTools	✓	-	-	-	-	-	-	-	-	-	-	-	-	-	-	(👤)	🤖	-
(Sample-)AlphaClean	-	-	-	-	-	-	-	✓	-	-	-	-	-	-	-	👤	🤖	-
Data Programming	-	-	-	-	-	-	-	✓	-	-	-	-	-	-	-	👤	🤖	-
HoloClean	-	-	-	-	-	-	-	✓	-	-	-	-	-	-	-	👤	🤖	-
CPClean	-	-	-	-	-	-	-	✓	-	-	-	-	-	-	-	👤	🤖	-
Raha & Baran	-	-	-	-	-	-	✓	✓	-	-	-	-	-	-	-	👤	🤖	-
HAIPipe	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	👤🤖	🤖	-
Feedback-driven improvement	-	-	-	-	-	-	-	✓	-	-	-	-	-	-	-	👤	🤖	✓
Rain	-	-	-	-	-	-	-	✓	-	-	-	-	-	-	-	👤	🤖	✓
Cleaning for ML	-	-	-	-	✓	-	✓	✓	-	-	-	-	-	-	-	👤	🤖	✓
AlpineMeadow	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	👤	🤖	✓
The Automatic Statistician	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	👤	🤖	✓
Learn2Clean+HIL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	👤	🤖	✓
Data Diff	-	-	-	-	-	-	-	✓	-	-	-	-	-	-	-	(🤖)	👤🤖	✓
Auto-Prep	-	-	✓	-	✓	-	-	-	-	-	✓	✓	✓	✓	-	🤖	👤	✓
Predictive Interaction	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	🤖	👤	✓
ChatPipe	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	🤖	👤	✓
Shadow pipelines	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	🤖	👤	✓
Function:	Integration				Cleaning				Preprocessing				Feat. engi.					
Purpose:	Organization				Quality				Model perf.									

Table 2: Semi-automated data preparation approaches,
ordered by semi-automation approach, transformation categories, and publication date
(✓: *not implemented but the method allows it*)

do not include a human component likewise focus on feature manipulation. Conversely, semi-automated approaches show more inclination towards data integration and cleaning than their AutoML counterparts, especially in the areas of parsing, content alignment, and value correction. These categories are plausibly where human insight is most needed, and may thus be the hardest to automate.

While fully automated approaches and some partly automated ones offer concrete implementations of both their methods and data transformations, others are more conceptual in nature—particularly when it comes to data transformations. Table 2 reflects this, as several frameworks propose methods with the potential to integrate the entire data preparation pipeline, without necessarily addressing the specifics of individual transformations. Instead, these systems primarily focus on pipeline design methodology.

5.3.2 Further Automation Perspectives

The semi-automated solutions examined in this section present a variety of ways to combine human and automated components. Some necessitate regular human participation, yet others require it only once. Some rely on human data insights and automate pipeline construction, while in others these responsibilities are reversed. The main value of human input when it comes to data preparation is generally considered to be their domain knowledge and technical expertise. The design choices in semi-automated systems indicate that this input still holds value. Yet, the differing degrees of human intervention and possibility or role reversal with automation would suggest that any aspect of the data preparation process could selectively be automated—sustaining the prospect of automating all of them in a unified process. The question remains whether this is feasible without compromising results, i.e., whether automation alone can adequately compensate for human contributions.

6 Automated Data Preparation

Having explored the extent of data preparation in automated machine learning and examined semi-automated data preparation approaches, we now turn to fully automated data preparation. For this category of approaches, we take into consideration systems that are specialized in automating data pipeline optimization, as well as AutoML systems whose optimization process covers the data pipeline along with the model (c.f. Section 4.1.2, Section 4.2).

6.1 Methodologies and Examples

Underlying methodologies for the optimization of data pipelines range from traditional optimization techniques, such as heuristic search and Bayesian optimization, to emerging ML paradigms such as foundation models. Some also integrate techniques for context-awareness [CTFL23] through the use of language models (word embeddings, LLMs), or learning from experience [GBA22] via transfer or meta-learning, in order to make the process more effective and efficient. We review these different approaches with the aim of extracting a structured overview of relevant methods along with their advantages, limitations, and particularities. A comparative summary of our findings is available in Table 3.

6.1.1 Sequential Model-Based Optimization

SMBO (Sequential Model-Based Optimization) [JSW98, BLP05, HHL11] is one type of method that be used to automatically optimize data preparation pipelines. An SMBO method navigates its search space with the help of a *surrogate model* that approximates the objective function, i.e., in our context, a loss function. The surrogate model is built by fitting data pipelines to models, training them, and evaluating them. It iteratively selects new points to evaluate by optimizing an *acquisition function*, which balances exploration and exploitation. The surrogate model is updated at every step based on the newly evaluated point. The most commonly used SMBO method is Bayesian optimization [SLA12, Gar22], where the surrogate model is probabilistic, such as a Gaussian process [Ras04].

SMBO methods allow for efficient optimization while focusing on the most informative or promising areas of the search space. Depending on the underlying surrogate model, they may become inefficient at greater scale, though there have been advances to mitigate that effect [EPG+19]. They are explicitly designed to support budget restrictions in the form of limited evaluations. Through the choice of interpretable surrogate models, they may in some cases provide explainability.

SMBO techniques have proven capable of automatically optimizing data pipelines with different surrogate functions—SVMs, neural networks, random forests, decision trees [HFT01]—in a fixed-architecture *DPSO* (Data Pipeline Selection and Optimization) [Que19] setup.

There are also multiple examples of Bayesian optimization approaches among AutoML solutions that include data pipeline optimization, such as Auto-WEKA [THHL13, KTH+17] and HyperOPT-sklearn [KBE14]. Others, namely Auto-PyTorch [ZLH21] and Auto-sklearn [FKE+15, FEF+22], rely on a Bayesian optimization base algorithm augmented by combinations with other approaches.

6.1.2 Evolutionary Algorithms

Evolutionary algorithms [BBBMM14] are optimization methods inspired by biological evolution and natural selection processes. They can iteratively optimize data preparation pipelines by evolving a population of *candidate solutions* in the form of data pipelines, over successive *generations*. At each iteration, the pipelines are evaluated according to a *fitness function*, corresponding to the downstream model's loss function. The best-performing candidates are retained. Those pipelines evolve into a next generation through mechanisms such as *mutation* via the addition, removal, swapping, or hyperparameter reconfiguration of data transformations, or *crossover* combinations with other pipelines.

Evolutionary approaches promote exploration, and are well-suited for complex black-box optimization problems. They can implement budget restrictions by limiting the population size and the number of generations, and offer some explainability by tracing back the evolution process. Depending on the optimization landscape and algorithm settings, their main limitation is that they may be slow to converge.

Evolutionary algorithms are the optimization method of choice for several AutoML systems with data pipeline optimization, namely FEDOT [NVS$^+$22], GAMA [GV21], TPOT [OM16], and EDCA [SC25].

6.1.3 Hierarchical Planning

Hierarchical planning [EHN94] draws on AI planning concepts to break down complex tasks into simpler subtasks. This can be done recursively, resulting in multiple levels of subtasks. Decisions are made at each level using search, typically best-first search [Pea84, DP85] or anytime search [HZ07], and evaluation. The principle can be applied to build ML or data pipeline construction in a top-down manner, by treating the construction process as a hierarchical task decomposition problem [EHN94]. The highest level task would be the goal itself: pipeline optimization. It can then be decomposed into subtasks at multiple levels. For instance, one subtask could be transformation category selection, then at the next level transformation operation selection within a category, and then hyperparameter configuration.

Hierarchical planning provides a structured search approach, which offers transparency and interpretability. It evaluates partial pipelines and prunes weak performers, making it quite efficient. In addition, it can provide a solution at any time (but also further refine it), allowing for a form of budget control. Its drawbacks are its dependency on the quality of underlying heuristics, overhead that can result from recursive decomposition and search, and its sequential nature, which limits parallelism.

The ML-PLAN [MWH18, WMTH19] AutoML system uses hierarchical planning to build ML pipelines. Using a hierarchical task network (HTN) process, it decomposes the task into feature preprocessing and modeling decisions at multiple levels.

6.1.4 Gradient Descent

Gradient descent [HFT01, Boy06] algorithms, very common in ML, optimize model parameters by iteratively calculating the gradient of the loss function and adjusting parameter values so as to minimize the loss. Though they are typically used for the downstream ML model's parameters rather than data preparation pipeline optimization, gradient descent algorithms can also be employed for the latter, as long as the pipeline construction process is modeled to be differentiable. The main benefit of using gradient descent for this problem is that it can optimize the data pipeline along with the model as a bi-level optimization problem, thus training the model only once—unlike most other methods that require repeated training with different pipelines.

Gradient descent is a very efficient optimization approach, but only works with differentiable downstream ML models. It also bears the risk of the gradient getting stuck in local minima, thus a proper tuning of hyperparameters, in particular the learning rate, is important. Gradient descent additionally allows for limited budget considerations through techniques such as early stopping.

DiffML [HHR+23] is an approach that consists in making whole ML pipelines, including data preparation pipelines, differentiable. It achieves this by expressing different pipeline possibilities as a so-called *mixture* of pipeline alternatives, whose weights can be optimized during the training process. By making the complete machine learning pipeline differentiable, this approach allows for combined data pipeline and model optimization with gradient methods.

DiffPrep [LCCR23] for tabular data is a method for data preparation pipeline searching along with model optimization, for differentiable ML models. It treats the problem as a bi-level optimization problem, minimizing training loss for the model training, and validation loss for the data pipeline design process. By relaxing the discrete non-differentiable pipeline search space into a continuous differentiable one [LSY19], this approach allows for pipeline optimization via gradient descent, jointly with model optimization.

6.1.5 Reinforcement Learning

Reinforcement learning [KLM96, Sut98] systems learn by performing *actions* in a modeled stateful *environment*. They receive environment feedback for those actions in the form of *rewards* following changes of state, and adjust their future behavior accordingly—thus iteratively shaping their decision *policy*. In the case of data preparation pipeline construction, actions correspond to data transformation decisions, while rewards represent downstream model performance. The developed decision policy is what guides pipeline construction, and the result is an optimized data pipeline.

RL approaches are typically good at modeling sequential tasks and learning long-term consequences of actions. However, they often require a large number of training loops, and may not converge in a stable manner.

Learn2Clean [Ber19a] relies on the Q-learning [Wat89, WD92] reinforcement learning technique to perform data preparation on Web data, which usually implies large amounts of dirty data. The system takes a dataset, ML model, and an evaluation metric as input. The choice of Q-learning allows for a relatively lightweight RL system despite the large search space due to its *model-free* nature, as well as a flexible one thanks to frequent reward updates for performed actions. Q-learning is typically effective on small search spaces, but may not always scale well.

HAIPipe [CTF$^+$23] is a semi-automated approach to data preparation, that includes automated data pipelines called *AI-Pipes* as one of its components—the other being a human-generated pipeline to combine with it. For our purposes, we focus on how this approach generates AI-Pipes. It proposes a method for the iterative design of pipelines based on DQN (Deep Q-Network) [MKS$^+$15] reinforcement learning. The DQN technique offers better scalability to large state spaces than tabular Q-learning, but is more computationally expensive.

CTXPipe [GCD$^+$24] is a system for context-aware automated data pipeline design. It is able to integrate additional context into the process by using pre-trained semantic embedding models [CP18]. This approach provides the system with domain expertise by allowing it to augment its training data through the extraction of semantic insights from it. With this added information, the system constructs data preparation pipelines using DQN reinforcement learning as its optimization method.

We can also find AutoML systems that optimize the data pipeline with reinforcement learning, such as DeepLine [HVKR20], which uses other techniques in addition RL to further enhance its optimization process.

6.1.6 Meta-Learning

Meta-learning [Sch87, HAMS21] is a process of *learning to learn* through prior experience. In the case of data preparation, by training many different data pipelines on many different datasets, meta-learning methods can capture knowledge about the pipeline optimization process by extracting common patterns. This provides them with strong generalization capabilities and fast convergence. Instead of learning how to optimize a pipeline for each dataset from scratch, meta-learning systems can leverage their gathered knowledge to efficiently construct data pipelines for previously unseen datasets. Nonetheless, the overhead in training a meta-learning model can be quite heavy.

One solution for automated data preparation via meta-learning [BAAW16] employs meta-learning concepts to evaluate positive or negative effects of different data preparation transformation applications within a ML pipeline. It then provides a *transformation ranking*, i.e., a set of recommended transformations to apply to the data.

MetaPrep [ZSDSS$^+$21] is an automated data preparation system based on meta-learning. Trained on a collection of example datasets and data pipelines, the system learns by extracting meta-knowledge in the form of data characteristics, model evaluations after applying data pipelines, and dataset similarity indicators. It can subsequently apply that knowledge to create quality data preparation pipelines for new datasets. MetaPrep returns the five best pipelines among those generated.

6.1.7 Foundation Models

Foundation models [BHA$^+$22], such as large language models (LLM) [VSP$^+$17, MMN$^+$25], are large pre-trained models that are trained on massive amounts of data. They offer general-purpose utility for a wide range of downstream tasks. Using vast corpora of language data from a great diversity of domains, LLMs learn broad language patterns that allow them to restore their knowledge by generating natural language, as well as programming code. For the purpose of building data preparation pipelines, LLMs can recommend data transformations and provide corresponding code. They can also optimize pipelines through repeated interaction to further improve their output.

LLMs offer unique upside in their ability to access domain knowledge outside of the user-provided data. However, LLMs also come with some known downsides: they are initially *very* expensive to train, their inference process is costly, and their generated output is not always reliable—they sometimes produce inconsistent responses, hallucinations, or inefficient code [HTQ$^+$23, LAB$^+$24, MEM$^+$24, KWB$^+$25]. A typical observation is that LLMs can generate good solutions with far fewer trials than classical optimization methods, but do not reach the same final performance.

CAAFE (Context-Aware Automated Feature Engineering) [HMH23] presents an LLM-based approach to automated data preparation that includes data context awareness. The system takes a dataset accompanied by a natural language problem specification by a user as input. It then iteratively generates a data preparation pipeline along with corresponding code, applies it to the data and trains a model, then re-generates data preparation code based on the model's performance results, and continues repeating the process.

6.2 Combined Methodologies

Combining multiple optimization methods is a convenient way to take advantage of their complementarity in order to enhance results. It is also possible to break down the optimization process and select a method most appropriate for each part.

Saga [SKB23] is a data preparation framework that builds data pipelines in two phases. It starts by finding the most promising pipelines using an evolutionary algorithm, which optimizes pipelines by adding or removing data transformations from them, reordering them, or mixing parts of two pipelines. The data transformations in this step are all configured with default hyperparameters. In the second phase, transformation hyperparameters are tuned using successive halving with Hyperband [JT16, LJD$^+$18]. Finally, the system ranks the pipelines and returns the top-k ones. Saga is designed to support parallelization. It also optionally includes user input in the form of custom constraints or data transformations.

AutoML systems also present a variety of combinations of optimization algorithms.

AUTO-PYTORCH enhances its Bayesian optimization approach with multi-fidelity optimization [FSK07, KFB$^+$17, FKH18] to strategically allocate its evaluation budget, and with meta-learning in order to warm-start the Bayesian optimizer.

Similarly, AUTO-SKLEARN, based on Bayesian optimization, uses meta-learning to initialize its Bayesian optimizer in order to start evaluations from quality points, and employs successive halving for better budget allocation.

Two members of the TPOT family extend the base approach with different additions: LAYERED-TPOT [GVO17] introduces a hierarchical search method that creates a layered structure to reduce the number of evaluations by favoring promising pipelines. TPOT-SH [PNJK19] uses successive halving to balance its budget, allocating less time to weak pipelines, and more to promising ones.

DEEPLINE constructs machine learning pipelines using reinforcement learning, augmented by meta-learning to guide actions, and hierarchical action filtering to structure and prune the search space, leading to faster convergence.

These combined approaches demonstrate the benefits of using complementary optimization techniques to enhance or overcome certain shortcomings of the base optimization algorithm. The most frequent choices are the use of meta-learning to warm-start the search process, thus speeding it up and increasing the likelihood of a quality solution; hierarchical methods to structure the search process, making it more efficient; multi-fidelity optimization and in particular successive halving, to improve search budget allocation based on the pertinence of the explored solution. Naturally, the addition of these techniques also comes with a cost, however, it is worth considering whether the extra cost is offset by their contributions.

6.3 Comparative Overview

Table 3 presents a comparative overview of pipeline optimization methods for automated data preparation. It summarizes their main advantages and limitations, and rates them according to relevant criteria when selecting an approach: *effectiveness* in terms of final solution quality; *efficiency*, both sampling and computational; having *low pre-requirements* such as additional resources and overhead; *context-awareness*; time, resource, or evaluation *budget control*; and *explainability*. It also categorizes the individual approaches discussed in the subsections above based on the broader optimization methodology they belong to. AutoML solutions (that also optimize the model, as opposed to just the data pipeline) are marked with an asterisk (*). Solutions that combine multiple optimization methods are marked with (+). It is worth noting that the mentioned advantages and limitations are of a rather general nature, and that individual approaches may implement additional techniques to overcome certain shortcomings.

Despite the lesser prominence of data preparation in AutoML when compared to modeling, the contributions reviewed in this section nonetheless show significant progress in

Opt. method	Advantages	Limitations	Effectiveness	Efficiency	Low pre-req.	Context-aware.	Budget control	Explainability	Example systems

SMBO (e.g., Bayesian optimization)	• Good at black-box optimization
	• Sample-efficient
	• Surrogates can capture uncertainty
		• Poorer scaling
		• Relatively large overhead
			✓	✓	~	-	✓	~	DPSO
									Auto-WEKA(*)
									HyperOPT-sklearn(*)
									Auto-PyTorch(*)(+)
									Auto-sklearn(*)(+)

Evolutionary algorithms	• Good at black-box optimization
	• Strong global exploration
		• Slow convergence	✓	~	✓	-	✓	~	FEDOT(*)
									GAMA(*)
									TPOT(*)
									EDCA(*)
									Saga(+)

[Opt. +] Hierarchical planning/search	• Structured and interpretable search
	• Efficient pruning
	• Anytime optimization
		• Heuristic dependency
		• Planning overhead
			✓	✓	~	-	~	✓	ML-PLAN(*)
									Layered-TPOT(*)(+)

Gradient descent	• Fast • Efficient	• Directly usable on differentiable problems only • Can get stuck in local optima	~	✓✓	~	-	~	-	DiffML DiffPrep

Reinforcement learning	• Good at sequential tasks
	• Learns long-term effects of actions
		• Sample-inefficient
		• Unstable convergence
			✓	~	~	-	-	-	Learn2Clean
									HAIPipe
									CTXPipe(+)
									DeepLine(*)(+)

[Opt. +] Meta-learning	• Generalization across datasets and pipelines
	• Few-shot adaptation
	• Warm-starting optimizers
		• Expensive training
		• Requires many prior datasets and pipelines
			✓	✓✓	✗	-	-	-	*Transformation ranking*
									MetaPrep
									Auto-PyTorch(*)(+)
									Auto-sklearn(*)(+)

Foundation models (e.g., LLMs)	• Provides domain knowledge • Query-efficient	• Unreliable • Requires pre-trained foundation models • Sub-optimal solutions	~	✗	✗✗	✓✓	-	-	CAAFE

Opt. + Multi-fidelity (e.g., Successive halving)	• Budget allocation balancing	• Requires reliable fidelity estimates	✓	✓✓	~	-	✓	-	Saga(+)
									Auto-PyTorch(*)(+)
									Auto-sklearn(*)(+)
									TPOT-SH(*)(+)

Opt. + Semantic embeddings	• Provides domain knowledge	• Requires pre-trained embedding models	✓	~	✗	✓	-	-	CTXPipe(+)

Table 3: Automated data preparation approaches
((*): *AutoML system*; (+): *combined methods*)

automating data pipeline optimization. They offer a varied selection of approaches depending on needs and available resources, as can be seen through comparison along some meaningful criteria in Table 3. These approaches propose different compromises in terms of efficiency, pre-requirements, as well budget or explainability concerns. The context-aware nature of certain methods is of particular interest, since it adds a new dimension to the automated process by providing a degree of domain expertise. Usually reserved to humans, this aspect represents one of the main challenges in effectively automating data preparation, next to achieving an efficient pipeline optimization process.

7 Conclusions and Outlook

Summary and Discussion. Data preparation is essential to making raw data usable and effective for machine learning tasks. Our structured overview of data transformations according to their purposes, functions, and category types illustrates the plethora of possible configurations in data pipeline optimization. In practice, selecting transformations, ordering them, and optimizing their hyperparameters is a time- and labor-intensive procedure. Handling it with automation allows to streamline the decision process, and substantially alleviates the required effort.

Examining data preparation in existing AutoML solutions has shown that they typically rely on their input data being ML-ready beforehand. We found that those systems, relatively light on data treatment, are primarily focused on basic usability and feature engineering, overlooking other important aspects such as data quality. Furthermore, only a few AutoML solutions include a pipeline optimization architecture that could support the complete data preparation process. Even so, none of them fully implement such a workflow.

Among semi-automated approaches, the human-guided-automation and automation-assisted-human method alternatives for pipeline construction highlight the persistent value of human analytics and domain expertise. The different roles of human and automation in these approaches allow for a view into the difficulties of automating certain data preparation elements, notably those that require domain knowledge, such as parsing and content alignment. At the same time, they present an opening to further automate the process.

In a review of fully automated data preparation approaches and their varied underlying optimization methods, we comparatively assessed their advantages and limitations. They namely consist in tradeoffs between effective, efficient optimization and resource requirements, in addition to considerations of budget restrictions and explainability. Moreover, the inclusion of natural language techniques to provide context-awareness to certain data pipeline optimization methods has shown promise in terms of replicating the benefits of domain expertise in an automated fashion.

Future Directions. Traditionally, performance optimization efforts in machine learning, as well as AutoML, have been concentrated on the modeling aspect of the workflow. The recent rise of the field of data-centric AI has raised awareness about the merits of optimizing data, in order to improve not only model loss, but also quality in terms of faithfully representing real-world phenomena. This entails a growing interest in optimizing data preparation ahead of, or in combination with, model optimization.

Though select areas of data treatment have successfully been automated by the database or machine learning communities, mainly in the areas of data organization and quality for the former, and model performance optimization for the latter, a separation between the two is quite evident. Beyond the search for novel techniques, connecting the contributions of both communities in a single pipeline architecture would allow to bridge that gap and implement a more holistic process that is automated end-to-end, ranging from raw data to ML model predictions.

The main challenge in automating data preparation for ML has consistently been the design of an optimization algorithm for the vast search space of data pipeline configurations that is both efficient and effective. A variety of optimization methods have been proposed to approach this problem, with some also offering perks such as context-awareness, budget control, or explainability. However, despite their merits, each of these approaches has certain shortcomings, leaving room for further refinement.

The emerging data-centricity paradigm in ML has brought to light the unrealized potential in optimizing data to expand the limits of machine learning processes. There has been significant progress in automating data preparation pipelines to date, yet substantial challenges still remain. Continued research in these directions, and the eventual complete automation of the ML workflow from raw data to inference, would constitute a profound advancement towards automating data science in an increasingly data-driven world.

Acknowledgments and Disclosure of Funding

This research is jointly funded by the French National Research Agency (ANR-23-CE23-0035) and the German Research Foundation (DFG; LI 2801/10-1), through project Opt4DAC.

References

[AAB$^+$19] Zeeshan Ahmed, Saeed Amizadeh, Mikhail Bilenko, Rogan Carr, Wei-Sheng Chin, Yael Dekel, Xavier Dupre, Vadim Eksarevskiy, Eric Erhardt, Costin Eseanu, Senja Filipi, Tom Finley, Abhishek Goswami, Monte Hoover, Scott Inglis, Matteo Interlandi, Shon Katzenberger, Najeeb Kazmi, Gleb Krivosheev, Pete Luferenko, Ivan Matantsev, Sergiy Matusevych, Shahab Moradi, Gani Nazirov, Justin Ormont, Gal Oshri, Artidoro Pagnoni, Jignesh Parmar, Prabhat Roy, Sarthak Shah, Mohammad Zeeshan Siddiqui, Markus Weimer, Shauheen Zahirazami, and Yiwen Zhu. Machine Learning at Microsoft with ML .NET. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2448–2458, July 2019.

[AFR$^+$20] Samuel Ackerman, Eitan Farchi, Orna Raz, Marcel Zalmanovici, and Parijat Dube. Detection of Data Drift and Outliers Affecting Machine Learning Model Performance Over Time. *JSM 2020 - Section on Nonparametric Statistics*, 2020.

[Agg15] Charu C. Aggarwal. *Data Mining.* Springer International Publishing AG, Cham, 2015.

[AH17] Charles Audet and Warren Hare. Introduction: Tools and Challenges in Derivative-Free and Blackbox Optimization. In Charles Audet and Warren Hare, editors, *Derivative-Free and Blackbox Optimization*, pages 3–14. Springer International Publishing, Cham, 2017.

[ALPS22] Lora Aroyo, Matthew Lease, Praveen Paritosh, and Mike Schaekermann. Data excellence for AI. *Interactions*, 29(2):66–69, February 2022.

[ALRM20] Filipe Assunção, Nuno Lourenço, Bernardete Ribeiro, and Penousal Machado. Evolution of Scikit-Learn Pipelines with Dynamic Structured Grammatical Evolution. In Pedro A. Castillo, Juan Luis Jiménez Laredo, and Francisco Fernández de Vega, editors, *Applications of Evolutionary Computation*, pages 530–545, Cham, 2020. Springer International Publishing.

[Ama19] Amazon. AutoML - Automated Machine Learning - Amazon Web Services. https://aws.amazon.com/sagemaker-ai/autopilot/, 2019.

[AW10] Hervé Abdi and Lynne J. Williams. Principal component analysis. *WIREs Computational Statistics*, 2(4):433–459, July 2010.

[BAAW16] Besim Bilalli, Alberto Abelló, Tomàs Aluja-Banet, and Robert Wrembel. Automated Data Pre-processing via Meta-learning. In Ladjel Bellatreche, Óscar Pastor, Jesús M. Almendros Jiménez, and Yamine Aït-Ameur, editors, *Model and Data Engineering*, pages 194–208, Cham, 2016. Springer International Publishing.

[BAI+22] Mehwish Bilal, Ghulam Ali, Muhammad Waseem Iqbal, Muhammad Anwar, Muhammad Sheraz Arshad Malik, and Rabiah Abdul Kadir. Auto-Prep: Efficient and Automated Data Preprocessing Pipeline. *IEEE Access*, 10:107764–107784, 2022.

[BBBMM14] Thomas Bartz-Beielstein, Jürgen Branke, Jörn Mehnen, and Olaf Mersmann. Evolutionary Algorithms. *WIREs Data Mining and Knowledge Discovery*, 4(3):178–195, April 2014.

[BBP+24] Nikita Bhatt, Nirav Bhatt, Purvi Prajapati, Vishal Sorathiya, Samah Alshathri, and Walid El-Shafai. A Data-Centric Approach to improve performance of deep learning models. *Scientific Reports*, 14(1):22329, September 2024.

[BCD+25] Bernd Bischl, Giuseppe Casalicchio, Taniya Das, Matthias Feurer, Sebastian Fischer, Pieter Gijsbers, Subhaditya Mukherjee, Andreas C. Müller, László Németh, Luis Oala, Lennart Purucker, Sahithya Ravi, Jan N. van Rijn, Prabhant Singh, Joaquin Vanschoren, Jos van der Velde, and Marcel Wever. OpenML: Insights from 10 years and more than a thousand papers. *Patterns*, 6(7), July 2025.

[BCFM09] Carlo Batini, Cinzia Cappiello, Chiara Francalanci, and Andrea Maurino. Methodologies for data quality assessment and improvement. *ACM Computing Surveys*, 41(3):1–52, July 2009.

[Ber19a] Laure Berti-Equille. Learn2Clean: Optimizing the Sequence of Tasks for Web Data Preparation. In *The World Wide Web Conference*, pages 2580–2586, San Francisco CA USA, May 2019. ACM.

[Ber19b] Leopoldo Bertossi. Database Repairs and Consistent Query Answering: Origins and Further Developments. In *Proceedings of the 38th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, PODS '19, pages 48–58, New York, NY, USA, June 2019. Association for Computing Machinery.

[Ber20] Laure Berti-Equille. Active Reinforcement Learning for Data Preparation: Learn2Clean with Human-In-The-Loop. In *Conference on Innovative Data Systems Research (CIDR 2020)*, Amsterdam (NETHERLANDS), Netherlands, January 2020.

[BHA+22] Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Ben Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, Julian Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Rob Reich, Hongyu Ren, Frieda Rong, Yusuf Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishnan Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. On the Opportunities and Risks of Foundation Models, July 2022.

[BL21] Sikha Bagui and Kunqi Li. Resampling imbalanced data for network intrusion detection datasets. *Journal of Big Data*, 8(1):6, January 2021.

[BLP05] T. Bartz-Beielstein, C.W.G. Lasarczyk, and M. Preuss. Sequential parameter optimization. In *2005 IEEE Congress on Evolutionary Computation*, volume 1, pages 773–780 Vol.1, September 2005.

[BMR+20] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language Models are Few-Shot Learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.

[Box11] George E. P. Box. *Bayesian Inference in Statistical Analysis.* Wiley &

Sons, Incorporated, John, 2011.

[Boy06] Stephen P. Boyd. *Convex Optimization.* Cambridge University Press, Cambridge, UK, 2006.

[BR21] Sumon Biswas and Hridesh Rajan. Fair preprocessing: Towards understanding compositional fairness of data transformers in machine learning pipeline. In *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2021, pages 981–993, New York, NY, USA, August 2021. Association for Computing Machinery.

[Bro20] Jason Brownlee. *Data Preparation for Machine Learning: Data Cleaning, Feature Selection, and Data Transforms in Python.* Machine Learning Mastery, June 2020.

[BWL$^+$24] Mitra Baratchi, Can Wang, Steffen Limmer, Jan N. van Rijn, Holger Hoos, Thomas Bäck, and Markus Olhofer. Automated machine learning: Past, present and future. *Artificial Intelligence Review*, 57(5):122, April 2024.

[BYC13] J Bergstra, D Yamins, and D D Cox. Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures. *Proceedings of Machine Learning Research*, 2013.

[CAL94] David Cohn, Les Atlas, and Richard Ladner. Improving generalization with active learning. *Machine Learning*, 15(2):201–221, May 1994.

[CAV22] Alexis Cvetkov-Iliev, Alexandre Allauzen, and Gaël Varoquaux. Analytics on Non-Normalized Data Sources: More Learning, Rather Than More Cleaning. *IEEE Access*, 10:42420–42431, 2022.

[CBHK02] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, 16:321–357, June 2002.

[CBNK18] Maximilian Christ, Nils Braun, Julius Neuffer, and Andreas W. Kempa-Liehr. Time Series FeatuRe Extraction on basis of Scalable Hypothesis tests (tsfresh – A Python package). *Neurocomputing*, 307:72–77, September 2018.

[CEP$^+$20] Vassilis Christophides, Vasilis Efthymiou, Themis Palpanas, George Papadakis, and Kostas Stefanidis. An Overview of End-to-End Entity Resolution for Big Data. *ACM Computing Surveys*, 53(6):1–42, December 2020.

[CG16] Tianqi Chen and Carlos Guestrin. XGBoost: A Scalable Tree Boosting

System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, August 2016. Association for Computing Machinery.

[Cho15] François Chollet. Keras-team/keras. Keras, 2015.

[CIKW16] Xu Chu, Ihab F. Ilyas, Sanjay Krishnan, and Jiannan Wang. Data Cleaning: Overview and Emerging Challenges. In *Proceedings of the 2016 International Conference on Management of Data*, SIGMOD '16, pages 2201–2206, New York, NY, USA, June 2016. Association for Computing Machinery.

[CJHS19] Jiaoyan Chen, Ernesto Jiménez-Ruiz, Ian Horrocks, and Charles Sutton. ColNet: Embedding the semantics of web tables for column type prediction. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, volume 33 of *AAAI'19/IAAI'19/EAAI'19*, pages 29–36, Honolulu, Hawaii, USA, January 2019. AAAI Press.

[CLJ$^+$24] Sibei Chen, Hanbing Liu, Waiting Jin, Xiangyu Sun, Xiaoyao Feng, Ju Fan, Xiaoyong Du, and Nan Tang. ChatPipe: Orchestrating Data Preparation Pipelines by Optimizing Human-ChatGPT Interactions. In *Companion of the 2024 International Conference on Management of Data*, SIGMOD/PODS '24, pages 484–487, New York, NY, USA, June 2024. Association for Computing Machinery.

[CLVZ11] Ken Chatfield, Victor Lempitsky, Andrea Vedaldi, and Andrew Zisserman. The devil is in the details: An evaluation of recent feature encoding methods. In *Procedings of the British Machine Vision Conference 2011*, pages 76.1–76.12, Dundee, 2011. British Machine Vision Association.

[CP18] Jose Camacho-Collados and Mohammad Taher Pilehvar. From Word To Sense Embeddings: A Survey on Vector Representations of Meaning. *Journal of Artificial Intelligence Research*, 63:743–788, December 2018.

[CPB25] Miguel Carvalho, Armando J. Pinho, and Susana Brás. Resampling approaches to handle class imbalance: A review from a data perspective. *Journal of Big Data*, 12(1):71, March 2025.

[CTF$^+$23] Sibei Chen, Nan Tang, Ju Fan, Xuemi Yan, Chengliang Chai, Guoliang Li, and Xiaoyong Du. HAIPipe: Combining Human-generated and Machine-generated Pipelines for Data Preparation. *Proceedings of the ACM on Management of Data*, 1(1):1–26, May 2023.

[CTFL23] Chengliang Chai, Nan Tang, Ju Fan, and Yuyu Luo. Demystifying Artificial Intelligence for Data Preparation. In *Companion of the 2023 International Conference on Management of Data*, SIGMOD '23, pages 13–20, New York, NY, USA, June 2023. Association for Computing Machinery.

[CWL+23] Chengliang Chai, Jiayi Wang, Yuyu Luo, Zeping Niu, and Guoliang Li. Data Management for Machine Learning: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, 35(5):4646–4667, May 2023.

[CWM+18] Boyuan Chen, Harvey Wu, Warren Mo, Ishanu Chattopadhyay, and Hod Lipson. Autostacker: A compositional evolutionary learning system. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '18, pages 402–409, New York, NY, USA, July 2018. Association for Computing Machinery.

[dab16] dabl. Dabl/dabl. dabl, 2016.

[Das04] Tamraparni Dasu. *Exploratory Data Mining and Data Cleaning.* John Wiley & Sons, Hoboken, NJ, 1 edition, 2004.

[Dat15] DataRobot. DataRobot | AI that makes business sense. https://www.datarobot.com/, 2015.

[DB17] Bolke De Bruin. Bolkedebruin/autokeras_tabular, 2017.

[DBDRH+22] Tijl De Bie, Luc De Raedt, José Hernández-Orallo, Holger H. Hoos, Padhraic Smyth, and Christopher K. I. Williams. Automating data science. *Communications of the ACM*, 65(3):76–87, February 2022.

[dCC23] Lucas B.V. de Amorim, George D.C. Cavalcanti, and Rafael M.O. Cruz. The choice of scaling technique matters for classification performance. *Applied Soft Computing*, 133:109924, January 2023.

[DDS+09] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, June 2009.

[de 17] Axel de Romblay. AxeldeRomblay/MLBox, 2017.

[Die95] Tom Dietterich. Overfitting and undercomputing in machine learning. *ACM Computing Surveys*, 27(3):326–327, September 1995.

[Die00] Thomas G. Dietterich. Ensemble Methods in Machine Learning. In Gerhard Goos, Juris Hartmanis, and Jan Van Leeuwen, editors, *Multiple Classifier Systems*, volume 1857, pages 1–15. Springer Berlin Heidelberg, Berlin, Heidelberg, 2000.

[DKN15] David J. Dittman, Taghi M. Khoshgoftaar, and Amri Napolitano. The Effect of Data Sampling When Using Random Forest on Imbalanced Bioinformatics Data. In *2015 IEEE International Conference on Information Reuse and Integration*, pages 457–463, August 2015.

[DKS95] James Dougherty, Ron Kohavi, and Mehran Sahami. Supervised and Unsupervised Discretization of Continuous Features. In *Machine Learning Proceedings 1995*, pages 194–202. Elsevier, 1995.

[Doa12] AnHai Doan. *Principles of Data Integration.* Elsevier Science & Technology Books, 2012.

[DP85] Rina Dechter and Judea Pearl. Generalized best-first search strategies and the optimality of A*. *Journal of the ACM*, 32(3):505–536, July 1985.

[dPOP17] Alex G. C. de Sá, Walter José G. S. Pinto, Luiz Otavio V. B. Oliveira, and Gisele L. Pappa. RECIPE: A Grammar-Based Framework for Automatically Evolving Classification Pipelines. In James McDermott, Mauro Castelli, Lukas Sekanina, Evert Haasdijk, and Pablo García-Sánchez, editors, *Genetic Programming*, pages 246–261, Cham, 2017. Springer International Publishing.

[EA22] Yotam Elor and Hadar Averbuch-Elor. To SMOTE, or not to SMOTE?, May 2022.

[EHN94] Kutluhan Erol, James Hendler, and Dana S Nau. UMCP: A Sound and Complete Procedure for Hierarchical Task-Network Planning. *Proceedings of the second Artificial Intelligence Planning Systems conference (AIPS)*, 1994.

[EKR$^+$22] Sabri Eyuboglu, Bojan Karlaš, Christopher Ré, Ce Zhang, and James Zou. Dcbench: A benchmark for data-centric AI systems. In *Proceedings of the Sixth Workshop on Data Management for End-To-End Machine Learning*, DEEM '22, pages 1–4, New York, NY, USA, June 2022. Association for Computing Machinery.

[EMS$^+$20] Nick Erickson, Jonas Mueller, Alexander Shirkov, Hang Zhang, Pedro Larroy, Mu Li, and Alexander Smola. AutoGluon-Tabular: Robust and Accurate AutoML for Structured Data, March 2020.

[EPG$^+$19] David Eriksson, Michael Pearce, Jacob Gardner, Ryan D Turner, and Matthias Poloczek. Scalable Global Optimization via Local Bayesian Optimization. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[fea19] Home of AutoViz featurewiz, AutoViML and. AutoViML/Auto_ViML,

2019.

[FEF$^+$22] Matthias Feurer, Katharina Eggensperger, Stefan Falkner, Marius Lindauer, and Frank Hutter. Auto-sklearn 2.0. *Journal of Machine Learning Research*, January 2022.

[FFR22] Harald Foidl, Michael Felderer, and Rudolf Ramler. Data smells: Categories, causes and consequences, and detection of suspicious data in AI-based systems. In *Proceedings of the 1st International Conference on AI Engineering: Software Engineering for AI*, pages 229–239, Pittsburgh Pennsylvania, May 2022. ACM.

[FKE$^+$15] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. Efficient and Robust Automated Machine Learning. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[FKH18] Stefan Falkner, Aaron Klein, and Frank Hutter. BOHB: Robust and Efficient Hyperparameter Optimization at Scale. In *Proceedings of the 35th International Conference on Machine Learning*, pages 1437–1446. PMLR, July 2018.

[FKK$^+$23] Alvaro A. A. Fernandes, Martin Koehler, Nikolaos Konstantinou, Pavel Pankin, Norman W. Paton, and Rizos Sakellariou. Data Preparation: A Technological Perspective and Review. *SN Computer Science*, 4(4):425, June 2023.

[FSK07] Alexander I.J Forrester, András Sóbester, and Andy J Keane. Multi-fidelity optimization via surrogate modelling. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 463(2088):3251–3269, October 2007.

[FV14] Benoit Frenay and Michel Verleysen. Classification in the Presence of Label Noise: A Survey. *IEEE Transactions on Neural Networks and Learning Systems*, 25(5):845–869, May 2014.

[Gar22] Roman Garnett. *Bayesian Optimization.* Cambridge University Press, 2022.

[GBA22] Joseph Giovanelli, Besim Bilalli, and Alberto Abelló. Data pre-processing pipeline generation for AutoETL. *Information Systems*, 108:101957, September 2022.

[GCD$^+$24] Haotian Gao, Shaofeng Cai, Tien Tuan Anh Dinh, Zhiyong Huang, and Beng Chin Ooi. CtxPipe: Context-aware Data Preparation Pipeline Construction for Machine Learning. *Proceedings of the ACM on Management* *of Data*, 2(6):1–27, December 2024.

[GGS24] Stefan Grafberger, Paul Groth, and Sebastian Schelter. Towards Interactively Improving ML Data Preparation Code via "Shadow Pipelines". In *Proceedings of the Eighth Workshop on Data Management for End-to-End Machine Learning*, DEEM '24, pages 7–11, New York, NY, USA, June 2024. Association for Computing Machinery.

[GLH15] Salvador García, Julián Luengo, and Francisco Herrera. *Data Preprocessing in Data Mining*. Intelligent Systems Reference Library. Springer International Publishing, Cham, 2015.

[GM12] Lise Getoor and Ashwin Machanavajjhala. Entity resolution. *Proceedings of the VLDB Endowment*, 5(12):2018–2019, August 2012.

[GNOT92] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, December 1992.

[Goo18] Google. AutoML Solutions - Train models without ML expertise. https://cloud.google.com/automl, 2018.

[GV21] Pieter Gijsbers and Joaquin Vanschoren. GAMA: A General Automated Machine Learning Assistant. In Yuxiao Dong, Georgiana Ifrim, Dunja Mladenić, Craig Saunders, and Sofie Van Hoecke, editors, *Machine Learning and Knowledge Discovery in Databases. Applied Data Science and Demo Track*, pages 560–564, Cham, 2021. Springer International Publishing.

[GVO17] Pieter Gijsbers, Joaquin Vanschoren, and Randal S Olson. Speeding up Tree-based Pipeline Optimization. *Proceedings of the International Workshop on Automatic Selection, Configuration and Composition of Machine Learning Algorithms (AutoML 2017)*, 2017.

[GZ19] Carlos Vladimiro Gonzalez Zelaya. Towards Explaining the Effects of Data Preprocessing on Machine Learning. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, pages 2086–2090, April 2019.

[H2O20] H2O. H2O Driverless AI. https://h2o.ai/platform/ai-cloud/make/h2o-driverless-ai/, 2020.

[HA04] Victoria Hodge and Jim Austin. A Survey of Outlier Detection Methodologies. *Artificial Intelligence Review*, 22(2):85–126, October 2004.

[HAMS21] Timothy M Hospedales, Antreas Antoniou, Paul Micaelli, and Amos J. Storkey. Meta-Learning in Neural Networks: A Survey. *IEEE Transac-*

tions on Pattern Analysis and Machine Intelligence, 44(9):1–1, 2021.

[HBGL08] Haibo He, Yang Bai, Edwardo A. Garcia, and Shutao Li. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 1322–1328, June 2008.

[HDW94] G. Holmes, A. Donkin, and I.H. Witten. WEKA: A machine learning workbench. In *Proceedings of ANZIIS '94 - Australian New Zealnd Intelligent Information Systems Conference*, pages 357–361, Brisbane, Qld., Australia, 1994. IEEE.

[HFT01] Trevor Hastie, Jerome Friedman, and Robert Tibshirani. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer, New York, NY, 2001.

[HHK15] Jeffrey Heer, Joseph M Hellerstein, and Sean Kandel. Predictive Interaction for Data Transformation. In *Conference on Innovative Data Systems Research*, CIDR15, 2015.

[HHL11] Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Sequential Model-Based Optimization for General Algorithm Configuration. In Carlos A. Coello Coello, editor, *Learning and Intelligent Optimization*, volume 6683, pages 507–523. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[HHR$^+$23] Benjamin Hilprecht, Christian Hammacher, Eduardo S Reis, Mohamed Abdelaal, and Carsten Binnig. DiffML: End-to-end Differentiable ML Pipelines. In *Proceedings of the Seventh Workshop on Data Management for End-to-End Machine Learning*, DEEM '23, pages 1–7, New York, NY, USA, June 2023. Association for Computing Machinery.

[HKV19] Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren. *Automated Machine Learning*. The Springer Series on Challenges in Machine Learning. Springer Nature, Cham, 2019.

[HMEH22] Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A Transformer That Solves Small Tabular Classification Problems in a Second. In *The Eleventh International Conference on Learning Representations*, September 2022.

[HMH23] Noah Hollmann, Samuel Müller, and Frank Hutter. Large language models for automated data science: Introducing CAAFE for context-aware automated feature engineering. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, pages 44753–44775, Red Hook, NY, USA, December 2023. Curran Associates Inc.

[HMP$^+$25] Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637(8045):319–326, January 2025.

[HN20] Mazhar Hameed and Felix Naumann. Data Preparation. *ACM SIGMOD Record*, 49(3):18–29, December 2020.

[HS98] Mauricio A. Hernández and Salvatore J. Stolfo. Real-world Data is Dirty: Data Cleansing and The Merge/Purge Problem. *Data Mining and Knowledge Discovery*, 2(1):9–37, January 1998.

[HTQ$^+$23] Muhammad Usman Hadi, Qasem Al Tashi, Rizwan Qureshi, Abbas Shah, Amgad Muneer, Muhammad Irfan, Anas Zafar, Muhammad Bilal Shaikh, Naveed Akhtar, Jia Wu, and Seyedali Mirjalili. Large Language Models: A Comprehensive Survey of its Applications, Challenges, Limitations, and Future Prospects, November 2023.

[HVKR20] Yuval Heffetz, Roman Vainshtein, Gilad Katz, and Lior Rokach. DeepLine: AutoML Tool for Pipelines Generation using Deep Reinforcement Learning and Hierarchical Actions Filtering. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, pages 2103–2113, New York, NY, USA, August 2020. Association for Computing Machinery.

[HYAP24] Sungwon Han, Jinsung Yoon, Sercan Ö. Arik, and Tomas Pfister. Large language models can automatically engineer features for few-shot tabular learning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *ICML'24*, pages 17454–17479, Vienna, Austria, July 2024. JMLR.org.

[HZ07] E. A. Hansen and R. Zhou. Anytime Heuristic Search. *Journal of Artificial Intelligence Research*, 28:267–297, March 2007.

[HZC21] Xin He, Kaiyong Zhao, and Xiaowen Chu. AutoML: A survey of the state-of-the-art. *Knowledge-Based Systems*, 212:106622, January 2021.

[JCSH23] Haifeng Jin, Francois Chollet, Qingquan Song, and Xia Hu. AutoKeras: An AutoML Library for Deep Learning. *Journal of Machine Learning Research*, 24, January 2023.

[JMG23] Mohammad Hossein Jarrahi, Ali Memariani, and Shion Guha. The Principles of Data-Centric AI. *Communications of the ACM*, 66(8):84–92, July 2023.

[JPN$^+$20] Abhinav Jain, Hima Patel, Lokesh Nagalapatti, Nitin Gupta, Sameep

Mehta, Shanmukha Guttula, Shashank Mujumdar, Shazia Afzal, Ruhi Sharma Mittal, and Vitobha Munigala. Overview and Importance of Data Quality for Machine Learning Tasks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, pages 3561–3562, New York, NY, USA, August 2020. Association for Computing Machinery.

[JSLH22] Weikuan Jia, Meili Sun, Jian Lian, and Sujuan Hou. Feature dimensionality reduction: A review. *Complex &amp; Intelligent Systems*, 8(3):2663–2693, January 2022.

[JSW98] Donald R. Jones, Matthias Schonlau, and William J. Welch. Efficient Global Optimization of Expensive Black-Box Functions. *Journal of Global Optimization*, 13(4):455–492, December 1998.

[JT16] Kevin Jamieson and Ameet Talwalkar. Non-stochastic Best Arm Identification and Hyperparameter Optimization. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pages 240–248. PMLR, May 2016.

[JVK$^+$24] Johannes Jakubik, Michael Vössing, Niklas Kühl, Jannis Walk, and Gerhard Satzger. Data-Centric Artificial Intelligence. *Business &amp; Information Systems Engineering*, 66(4):507–515, March 2024.

[KBE14] Brent Komer, James Bergstra, and Chris Eliasmith. Hyperopt-Sklearn: Automatic Hyperparameter Configuration for Scikit-Learn. In *Python in Science Conference*, pages 32–37, Austin, Texas, 2014.

[KCH$^+$03] Won Kim, Byoung-Ju Choi, Eui-Kyeong Hong, Soo-Kyung Kim, and Doheon Lee. A Taxonomy of Dirty Data. *Data Mining and Knowledge Discovery*, 7(1):81–99, January 2003.

[KDS$^+$24] Sushant Kumar, Sumit Datta, Vishakha Singh, Sanjay Kumar Singh, and Ritesh Sharma. Opportunities and Challenges in Data-Centric AI. *IEEE Access*, 12:33173–33189, 2024.

[KEVDS19] Justian Knobbout, Huub Everaert, and Esther Van Der Stappen. From dirty data to multiple versions of truth: How different choices in data cleaning lead to different learning analytics outcomes. In *Humanizing Technology for a Sustainable Society*, pages 49–66. Univresity of Maribor Press, 2019.

[KFB$^+$17] Aaron Klein, Stefan Falkner, Simon Bartels, Philipp Hennig, and Frank Hutter. Fast Bayesian Optimization of Machine Learning Hyperparameters on Large Datasets. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, pages 528–536. PMLR, April

2017.

[KFG$^+$16] Sanjay Krishnan, Michael J. Franklin, Ken Goldberg, Jiannan Wang, and Eugene Wu. ActiveClean: An Interactive Data Cleaning Framework For Modern Machine Learning. In *Proceedings of the 2016 International Conference on Management of Data*, pages 2117–2120, San Francisco California USA, June 2016. ACM.

[KFGW17] Sanjay Krishnan, Michael J. Franklin, Ken Goldberg, and Eugene Wu. BoostClean: Automated Error Detection and Repair for Machine Learning, November 2017.

[KGH07] Taghi M. Khoshgoftaar, Moiz Golawala, and Jason Van Hulse. An Empirical Study of Learning from Imbalanced Data Using Random Forest. In *19th IEEE International Conference on Tools with Artificial Intelligence(ICTAI 2007)*, volume 2, pages 310–317, October 2007.

[KL17] Pang Wei Koh and Percy Liang. Understanding Black-box Predictions via Influence Functions. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1885–1894. PMLR, July 2017.

[KLM96] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement Learning: A Survey. *Journal of Artificial Intelligence Research*, 4:237–285, May 1996.

[KLW$^+$20] Bojan Karlaš, Peng Li, Renzhi Wu, Nezihe Merve Gürel, Xu Chu, Wentao Wu, and Ce Zhang. Nearest neighbor classifiers over incomplete information. *Proceedings of the VLDB Endowment*, 14(3):255–267, November 2020.

[KMP17] Ambika Kaul, Saket Maheshwary, and Vikram Pudi. AutoLearn — Automated Feature Generation and Selection. In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 217–226, November 2017.

[KNH$^+$24] Agata Kozina, Michał Nadolny, Marcin Hernes, Ewa Walaszczyk, and Artur Rot. One Hot Encoding and Hashing_Trick Transformation - Performance Comparision. In *2024 14th International Conference on Advanced Computer Information Technologies (ACIT)*, pages 699–704, September 2024.

[KP20] Nikolaos Konstantinou and Norman W. Paton. Feedback driven improvement of data preparation pipelines. *Information Systems*, 92:101480, September 2020.

[KPZS20] Jonathan Krauß, Bruno Machado Pacheco, Hanno Maximilian Zang, and Robert Heinrich Schmitt. Automated machine learning for predictive quality in production. *Procedia CIRP*, 93:443–448, 2020.

[Kra18] Tim Kraska. Northstar. *Proceedings of the VLDB Endowment*, 11(12):2150–2164, August 2018.

[Kra20] Jonathan Krauß. AutoML Benchmark. https://jonathankrauss.github.io/AutoML-Benchmark//AutoML-Benchmark/, 2020.

[Kri09] Alex Krizhevsky. Learning Multiple Layers of Features from Tiny Images, 2009.

[KSA+21] Matloob Khushi, Kamran Shaukat, Talha Mahboob Alam, Ibrahim A. Hameed, Shahadat Uddin, Suhuai Luo, Xiaoyan Yang, and Maranatha Consuelo Reyes. A Comparative Performance Analysis of Data Resampling Methods on Imbalance Medical Data. *IEEE Access*, 9:109960–109975, 2021.

[KSHS+21] Shubhra Kanti Karmaker ("Santu"), Md. Mahadi Hassan, Micah J. Smith, Lei Xu, Chengxiang Zhai, and Kalyan Veeramachaneni. AutoML to Date and Beyond: Challenges and Opportunities. *ACM Computing Surveys*, 54(8):1–36, October 2021.

[KSS16] Gilad Katz, Eui Chul Richard Shin, and Dawn Song. ExploreKit: Automatic Feature Generation and Selection. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 979–984, Barcelona, Spain, December 2016. IEEE.

[KTH+17] Lars Kotthoff, Chris Thornton, Holger H. Hoos, Frank Hutter, and Kevin Leyton-Brown. Auto-WEKA 2.0: Automatic model selection and hyperparameter optimization in WEKA. *Journal of Machine Learning Research*, 18(25):1–5, 2017.

[Kum21] Arun Kumar. Automation of Data Prep, ML, and Data Science: New Cure or Snake Oil? In *Proceedings of the 2021 International Conference on Management of Data*, SIGMOD '21, pages 2878–2880, New York, NY, USA, June 2021. Association for Computing Machinery.

[KV15] James Max Kanter and Kalyan Veeramachaneni. Deep feature synthesis: Towards automating data science endeavors. In *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 1–10, Campus des Cordeliers, Paris, France, October 2015. IEEE.

[KW19] Sanjay Krishnan and Eugene Wu. AlphaClean: Automatic Generation of Data Cleaning Pipelines, May 2019.

[KWB$^+$25] Aida Kostikova, Zhipin Wang, Deidamea Bajri, Ole Pütz, Benjamin Paaßen, and Steffen Eger. LLLMs: A Data-Driven Survey of Evolving Research on Limitations of Large Language Models, May 2025.

[KWF$^+$15] Sanjay Krishnan, Jiannan Wang, Michael J Franklin, Ken Goldberg, Tim Kraska, Tova Milo, and Eugene Wu. SampleClean: Fast and Reliable Analytics on Dirty Data. *IEEE Data Eng. Bull.*, 38:59–75, 2015.

[LAB$^+$24] Md Tahmid Rahman Laskar, Sawsan Alqahtani, M Saiful Bari, Mizanur Rahman, Mohammad Abdullah Matin Khan, Haidar Khan, Israt Jahan, Amran Bhuiyan, Chee Wei Tan, Md Rizwan Parvez, Enamul Hoque, Shafiq Joty, and Jimmy Huang. A Systematic Survey and Critical Review on Evaluating Large Language Models: Challenges, Limitations, and Recommendations. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 13785–13816, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[Law17] Neil D. Lawrence. Data Readiness Levels, May 2017.

[LBBH98] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[LBH15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, May 2015.

[LCCR23] Peng Li, Zhiyi Chen, Xu Chu, and Kexin Rong. DiffPrep: Differentiable Data Preprocessing Pipeline Search for Learning over Tabular Data. *Proceedings of the ACM on Management of Data*, 1(2):1–26, June 2023.

[LCW$^+$17] Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P. Trevino, Jiliang Tang, and Huan Liu. Feature Selection. *ACM Computing Surveys*, 50(6):1–45, December 2017.

[LDZ25] Yin Lin, Bolin Ding, and Jingren Zhou. Large Language Models as Pretrained Data Engineers: Techniques and Opportunities. *IEEE Data Engineering Bulletin*, 2025.

[LEF$^+$22] Marius Lindauer, Katharina Eggensperger, Matthias Feurer, André Biedenkapp, Difan Deng, Carolin Benjamins, Tim Ruhkopf, René Sass, and Frank Hutter. SMAC3: A Versatile Bayesian Optimization Package for Hyperparameter Optimization. *Journal of Machine Learning Research*, 23(54):1–9, 2022.

[Len02] Maurizio Lenzerini. Data integration: A theoretical perspective. In *Pro-*

ceedings of the Twenty-First ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS '02, pages 233–246, New York, NY, USA, June 2002. Association for Computing Machinery.

[LG14] Vu Le and Sumit Gulwani. FlashExtract: A framework for data extraction by examples. In *Proceedings of the 35th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 542–553, Edinburgh United Kingdom, June 2014. ACM.

[LJD+18] Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization. *Journal of Machine Learning Research*, 18(185):1–52, 2018.

[LLR+23] Roque Lopez, Raoni Lourenco, Remi Rampin, Sonia Castelo, Aécio S. R. Santos, Jorge Henrique Piazentin Ono, Claudio Silva, and Juliana Freire. AlphaD3M: An Open-Source AutoML Library for Multiple ML Tasks. In *Proceedings of the Second International Conference on Automated Machine Learning*, pages 22/1–22. PMLR, December 2023.

[LMWY13] Ji Liu, Przemyslaw Musialski, Peter Wonka, and Jieping Ye. Tensor Completion for Estimating Missing Values in Visual Data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):208–220, January 2013.

[LP20] E LeDell and S Poirier. H2O AutoML: Scalable Automatic Machine Learning. In *7th ICML Workshop on Automated Machine Learning*, 2020.

[LRB+21] Peng Li, Xi Rao, Jennifer Blase, Yue Zhang, Xu Chu, and Ce Zhang. CleanML: A Study for Evaluating the Impact of Data Cleaning on ML Classification Tasks. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, pages 13–24, April 2021.

[LRV+20] Sijia Liu, Parikshit Ram, Deepak Vijaykeerthy, Djallel Bouneffouf, Gregory Bramble, Horst Samulowitz, Dakuo Wang, Andrew Conn, and Alexander Gray. An ADMM Based Framework for AutoML Pipeline Configuration. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):4892–4899, April 2020.

[LSY19] Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: Differentiable Architecture Search, April 2019.

[LT19] Wei-Chao Lin and Chih-Fong Tsai. Missing value imputation: A review and analysis of the literature (2006–2017). *Artificial Intelligence Review*, 53(2):1487–1509, April 2019.

[MA20] Mohammad Mahdavi and Ziawasch Abedjan. Baran. *Proceedings of the VLDB Endowment*, 13(12):1948–1961, August 2020.

[Mac67] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, volume 5.1, pages 281–298. University of California Press, January 1967.

[MACF$^+$19] Mohammad Mahdavi, Ziawasch Abedjan, Raul Castro Fernandez, Samuel Madden, Mourad Ouzzani, Michael Stonebraker, and Nan Tang. Raha: A Configuration-Free Error Detection System. In *Proceedings of the 2019 International Conference on Management of Data*, pages 865–882, Amsterdam Netherlands, June 2019. ACM.

[MBF$^+$25] Sedir Mohammed, Lukas Budach, Moritz Feuerpfeil, Nina Ihde, Andrea Nathansen, Nele Noack, Hendrik Patzlaff, Felix Naumann, and Hazar Harmouch. The effects of data quality on machine learning performance on tabular data. *Information Systems*, 132:102549, July 2025.

[MBY$^+$23] Mark Mazumder, Colby Banbury, Xiaozhe Yao, Bojan Karlaš, William Gaviria Rojas, Sudnya Diamos, Greg Diamos, Lynn He, Alicia Parrish, Hannah Rose Kirk, Jessica Quaye, Charvi Rastogi, Douwe Kiela, David Jurado, David Kanter, Rafael Mosquera, Will Cukierski, Juan Ciro, Lora Aroyo, Bilge Acun, Lingjiao Chen, Mehul Raje, Max Bartolo, Evan Sabri Eyuboglu, Amirata Ghorbani, Emmett Goodman, Addison Howard, Oana Inel, Tariq Kane, Christine R. Kirkpatrick, D. Sculley, Tzu-Sheng Kuo, Jonas W. Mueller, Tristan Thrush, Joaquin Vanschoren, Margaret Warren, Adina Williams, Serena Yeung, Newsha Ardalani, Praveen Paritosh, Ce Zhang, James Y. Zou, Carole-Jean Wu, Cody Coleman, Andrew Ng, Peter Mattson, and Vijay Janapa Reddi. DataPerf: Benchmarks for Data-Centric AI Development. *Advances in Neural Information Processing Systems*, 36:5320–5347, December 2023.

[MCCD13] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space, September 2013.

[MDM19] Piero Molino, Yaroslav Dudin, and Sai Sumanth Miryala. Ludwig: A type-based declarative deep learning toolbox, September 2019.

[MEM$^+$24] Chiara Michelutti, Jens Eckert, Milko Monecke, Julian Klein, and Sabine Glesner. A Systematic Study on the Potentials and Limitations of LLM-assisted Software Development. In *2024 2nd International Conference on Foundation and Large Language Models (FLLM)*, pages 330–338, November 2024.

[MHSG18] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. UMAP: Uniform Manifold Approximation and Projection. *Journal of Open Source Software*, 3(29):861, September 2018.

[Mic18a] Microsoft. Automated Machine Learning (AutoML) | Microsoft Azure. https://azure.microsoft.com/en-us/solutions/automated-machine-learning, 2018.

[Mic18b] Microsoft. Neural Network Intelligence. Microsoft, 2018.

[MKS$^+$15] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015.

[MLJ18] MLJAR. Mljar/mljar-supervised. MLJAR, 2018.

[MM21] Nuno Moniz and Hugo Monteiro. No Free Lunch in imbalanced learning. *Knowledge-Based Systems*, 227:107222, September 2021.

[MM22] Alhassan Mumuni and Fuseini Mumuni. Data augmentation: A comprehensive survey of modern approaches. *Array*, 16:100258, December 2022.

[MMN$^+$25] Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. Large Language Models: A Survey, March 2025.

[MP24] Donato Malerba and Vincenzo Pasquadibisceglie. Data-Centric AI. *Journal of Intelligent Information Systems*, 62(6):1493–1502, October 2024.

[MSK21] Mohammad Motamedi, Nikolay Sakharnykh, and Tim Kaldewey. A Data-Centric Approach for Training Deep Neural Networks with Less Data, October 2021.

[Mun12] M. Arthur Munson. A study on the importance of and time spent on different modeling steps. *SIGKDD Explor. Newsl.*, 13(2):65–71, May 2012.

[MV25] Marine Le Morvan and Gaël Varoquaux. Imputation for prediction: Beware of diminishing returns, February 2025.

[MW22] Felix Mohr and Marcel Wever. Naive automated machine learning. *Machine Learning*, 112(4):1131–1170, September 2022.

[MWH18] Felix Mohr, Marcel Wever, and Eyke Hüllermeier. ML-Plan: Automated machine learning via hierarchical planning. *Machine Learning*, 107(8-

10):1495–1515, July 2018.

[NAR13] Nazri Mohd Nawi, Walid Hasen Atomi, and M.Z. Rehman. The Effect of Data Pre-processing on Optimized Training of Artificial Neural Networks. *Procedia Technology*, 11:32–39, 2013.

[Nav01] Gonzalo Navarro. A guided tour to approximate string matching. *ACM Computing Surveys*, 33(1):31–88, March 2001.

[NCA$^+$21] Felix Neutatz, Binger Chen, Ziawasch Abedjan, Eugene Wu, and TU Berlin. From Cleaning before ML to Cleaning for ML. *IEEE Data Engineering Bulletin*, 2021.

[NCA$^+$22] Felix Neutatz, Binger Chen, Yazan Alkhatib, Jingwen Ye, and Ziawasch Abedjan. Data Cleaning and AutoML: Would an Optimizer Choose to Clean? *Datenbank-Spektrum*, 22(2):121–130, May 2022.

[Nea93] Radford M Neal. Probabilistic Inference Using Markov Chain Monte Carlo Methods, 1993.

[NVS$^+$22] Nikolay O. Nikitin, Pavel Vychuzhanin, Mikhail Sarafanov, Iana S. Polonskaia, Ilia Revin, Irina V. Barabanova, Gleb Maximov, Anna V. Kalyuzhnaya, and Alexander Boukhanovsky. Automated evolutionary approach for the design of composite machine learning pipelines. *Future Generation Computer Systems*, 127:109–125, February 2022.

[NWC$^+$20] Alfredo Nazabal, Christopher K. I. Williams, Giovanni Colavizza, Camila Rangel Smith, and Angus Williams. Data Engineering for Data Analytics: A Classification of the Issues, and Case Studies, April 2020.

[OAA$^+$24] OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fish-

man, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, C. J. Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sher-

win Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. GPT-4 Technical Report, March 2024.

[OM16] Randal S. Olson and Jason H. Moore. TPOT: A Tree-Based Pipeline Optimization Tool for Automating Machine Learning. In Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren, editors, *Automated Machine Learning*, pages 151–160. Springer International Publishing, Cham, 2016.

[OMB+24] Luis Oala, Manil Maskey, Lilith Bat-Leah, Alicia Parrish, Nezihe Merve, Tzu-Sheng Kuo, Yang Liu, Rotem Dror, and Danilo Brajovic. DMLR: Data-centric Machine Learning Research - Past, Present and Future. *Journal of Data-centric Machine Learning Research*, 2024.

[Par16] Preston Parry. ClimbsRocks/auto_ml, 2016.

[Pat19] Norman Paton. Automating Data Preparation: Can We? Should We? Must We? In *Proceedings of the 21st International Workshop on Design, Optimization, Languages and Analytical Processing of Big Data*, 2019.

[PBGN23] Rajvardhan Patil, Sorio Boit, Venkat Gudivada, and Jagadeesh Nandigam. A Survey of Text Representation and Embedding Techniques in NLP. *IEEE Access*, 11:36120–36146, 2023.

[PDTL20] Jose Picado, John Davis, Arash Termehchy, and Ga Young Lee. Learning Over Dirty Data Without Cleaning. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, SIGMOD '20, pages 1301–1316, New York, NY, USA, May 2020. Association for Computing Machinery.

[Pea01] Karl Pearson. LIII. *On lines and planes of closest fit to systems of points in space. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, November 1901.

[Pea84] Judea Pearl. *Heuristics.* Addison-Wesley Longman Publishing Co., Inc., Reading, Mass, April 1984.

[Pet09] Leif Peterson. K-nearest neighbor. *Scholarpedia*, 4(2):1883, 2009.

[PGM+19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural*

Information Processing Systems, volume 32. Curran Associates, Inc., 2019.

[PLW02] Leo L. Pipino, Yang W. Lee, and Richard Y. Wang. Data quality assessment. *Communications of the ACM*, 45(4):211–218, April 2002.

[PNJK19] Laurent Parmentier, Olivier Nicol, Laetitia Jourdan, and Marie-Eleonore Kessaci. TPOT-SH: A Faster Optimization Algorithm to Solve the AutoML Problem on Large Datasets. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 471–478, November 2019.

[PPV$^+$11] Fabian Pedregosa, Fabian Pedregosa, Gael Varoquaux, Gael Varoquaux, Normalesup Org, Alexandre Gramfort, Alexandre Gramfort, Vincent Michel, Vincent Michel, Logilab Fr, Bertrand Thirion, Bertrand Thirion, Olivier Grisel, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, Alexandre Tp, and David Cournapeau. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 2011.

[QHZ$^+$24] Yichen Qian, Yongyi He, Rong Zhu, Jintao Huang, Zhijian Ma, Haibin Wang, Yaohua Wang, Xiuyu Sun, Defu Lian, Bolin Ding, and Jingren Zhou. UniDM: A Unified Framework for Data Manipulation with Large Language Models. *Conference on Machine Learning and Systems*, 2024.

[Que19] Alexandre Quemy. Data Pipeline Selection and Optimization. *Proceedings of the 21st International Workshop on Design, Optimization, Languages and Analytical Processing of Big Data (DOLAP 2019)*, 2019.

[Ras04] Carl Edward Rasmussen. Gaussian Processes in Machine Learning. In Olivier Bousquet, Ulrike von Luxburg, and Gunnar Rätsch, editors, *Advanced Lectures on Machine Learning: ML Summer Schools 2003, Canberra, Australia, February 2 - 14, 2003, Tübingen, Germany, August 4 - 16, 2003, Revised Lectures*, pages 63–71. Springer, Berlin, Heidelberg, 2004.

[RBCS22] Valerie Restat, Gerrit Boerner, André Conrad, and Uta Störl. GouDa - generation of universal data sets: Improving analysis and evaluation of data preparation pipelines. In *Proceedings of the Sixth Workshop on Data Management for End-To-End Machine Learning*, DEEM '22, pages 1–6, New York, NY, USA, June 2022. Association for Computing Machinery.

[RCIR17] Theodoros Rekatsinas, Xu Chu, Ihab F. Ilyas, and Christopher Ré. HoloClean. *Proceedings of the VLDB Endowment*, 10(11):1190–1201, August 2017.

[RD00] Erhard Rahm and Hong Hai Do. Data Cleaning: Problems and Current Approaches. *IEEE Techn. Bulletin on Data Engineering*, 2000.

[RDSW$^+$16] Alexander J Ratner, Christopher M De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. Data Programming: Creating Large Training Sets, Quickly. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[RLSL20] Esteban Real, Chen Liang, David R. So, and Quoc V. Le. AutoML-Zero: Evolving machine learning algorithms from scratch. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *ICML'20*, pages 8007–8019. JMLR.org, July 2020.

[RNSS18] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving Language Understanding by Generative Pre-Training, 2018.

[Rob52] Herbert Robbins. Some Aspects of the Sequential Design of Experiments, 1952.

[RRK$^+$22] Cedric Renggli, Luka Rimanic, Luka Kolar, Wentao Wu, and Ce Zhang. Automatic Feasibility Study via Data Quality Analysis for ML: A Case-Study on Label Noise, August 2022.

[RSS19] Herilalaina Rakotoarison, Marc Schoenauer, and Michèle Sebag. Automated Machine Learning with Monte-Carlo Tree Search. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pages 3296–3303, Macao, China, August 2019. International Joint Conferences on Artificial Intelligence Organization.

[RTL09] Payam Refaeilzadeh, Lei Tang, and Huan Liu. Cross-Validation. In *Encyclopedia of Database Systems*, pages 532–538. Springer, Boston, MA, 2009.

[Rud19] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, May 2019.

[RZS$^+$20] Michael Ruchte, Arber Zela, Julien Niklas Siems, Josif Grabocka, and Frank Hutter. NASLib: A Modular and Flexible Neural Architecture Search Library, October 2020.

[Sal17] Salesforce. Salesforce/TransmogrifAI. Salesforce, 2017.

[SC25] Joana Simões and João Correia. EDCA – An Evolutionary Data-Centric AutoML Framework for Efficient Pipelines. In Pablo García-Sánchez, Emma Hart, and Sarah L. Thomson, editors, *Applications of Evolutionary*

Computation, pages 71–88, Cham, 2025. Springer Nature Switzerland.

[Sch87] Jürgen Schmidhuber. *Evolutionary Principles in Self—Referential Learning.* PhD thesis, ech. Univ. Munich., 1987.

[Sch90] Robert E. Schapire. The strength of weak learnability. *Machine Learning*, 5(2):197–227, June 1990.

[Sch03] Robert E. Schapire. The Boosting Approach to Machine Learning: An Overview. In David D. Denison, Mark H. Hansen, Christopher C. Holmes, Bani Mallick, and Bin Yu, editors, *Nonlinear Estimation and Classification*, pages 149–171. Springer, New York, NY, 2003.

[SCvdS22] Nabeel Seedat, Jonathan Crabbé, and Mihaela van der Schaar. Data-SUITE: Data-centric identification of in-distribution incongruous examples. In *Proceedings of the 39th International Conference on Machine Learning*, pages 19467–19496. PMLR, June 2022.

[Sd04] Roland Soong and Michelle de Montigny. No Free Lunch In Data Fusion / Integration. In *Proceedings of the ESOMAR Worldwide Audience Measurement Conference*, Geneva, Switzerland, 2004.

[SDC$^+$17] Thomas Swearingen, Will Drevo, Bennett Cyphers, Alfredo Cuesta-Infante, Arun Ross, and Kalyan Veeramachaneni. ATM: A distributed, collaborative, scalable system for automated machine learning. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 151–162, Boston, MA, December 2017. IEEE.

[SGG24] Sebastian Schelter, Shubha Guha, and Stefan Grafberger. Automated Provenance-Based Screening of ML Data Preparation Pipelines. *Datenbank-Spektrum*, 24(3):187–196, September 2024.

[SHB93] Milan Sonka, Vaclav Hlavac, and Roger Boyle. Image pre-processing. In Milan Sonka, Vaclav Hlavac, and Roger Boyle, editors, *Image Processing, Analysis and Machine Vision*, pages 56–111. Springer US, Boston, MA, 1993.

[SHGC18] Charles Sutton, Timothy Hobson, James Geddes, and Rich Caruana. Data Diff: Interpretable, Executable Summaries of Changes in Distributions for Data Wrangling. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2279–2288, London United Kingdom, July 2018. ACM.

[SHK$^+$14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, January 2014.

[Sin23] Prerna Singh. Systematic review of data-centric approaches in artificial intelligence and machine learning. *Data Science and Management*, 6(3):144–157, September 2023.

[SIS+24] Imrus Salehin, Md. Shamiul Islam, Pritom Saha, S.M. Noman, Azra Tuni, Md. Mehedi Hasan, and Md. Abu Baten. AutoML: A systematic review on automated machine learning with neural architecture search. *Journal of Information and Intelligence*, 2(1):52–81, January 2024.

[SK19] Vraj Shah and Arun Kumar. The ML Data Prep Zoo: Towards Semi-Automatic Data Preparation for ML. In *Proceedings of the 3rd International Workshop on Data Management for End-to-End Machine Learning*, DEEM'19, pages 1–4, New York, NY, USA, June 2019. Association for Computing Machinery.

[SK24] Sebastian Strasser and Meike Klettke. Transparent Data Preprocessing for Machine Learning. In *Proceedings of the 2024 Workshop on Human-In-the-Loop Data Analytics*, HILDA 24, pages 1–6, New York, NY, USA, June 2024. Association for Computing Machinery.

[SKB23] Shafaq Siddiqi, Roman Kern, and Matthias Boehm. SAGA: A Scalable Framework for Optimizing Data Cleaning Pipelines for Machine Learning Applications. *Proceedings of the ACM on Management of Data*, 1(3):1–26, November 2023.

[SKKR01] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web*, pages 285–295, Hong Kong Hong Kong, April 2001. ACM.

[SKP+23] Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. Learning From Noisy Labels With Deep Neural Networks: A Survey. *IEEE Transactions on Neural Networks and Learning Systems*, 34(11):8135–8153, November 2023.

[skr18] skrub. Skrub-data/skrub. skrub-data, 2018.

[SLA12] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical Bayesian Optimization of Machine Learning Algorithms. In *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[Sli19] Aleksandrs Slivkins. Introduction to Multi-Armed Bandits. *Foundations and Trends® in Machine Learning*, 12(1-2):1–286, 2019.

[SSJ+19] Christian Steinruecken, Emma Smith, David Janz, James Lloyd, and Zoubin Ghahramani. The Automatic Statistician. In Frank Hutter, Lars

Kotthoff, and Joaquin Vanschoren, editors, *Automated Machine Learning: Methods, Systems, Challenges*, pages 161–173. Springer International Publishing, Cham, 2019.

[STE+23] Oleksandr Shchur, Caner Turkmen, Nick Erickson, Huibin Shen, Alexander Shirkov, Tony Hu, and Yuyang Wang. AutoGluon–TimeSeries: AutoML for Probabilistic Time Series Forecasting. In *Proceedings of the AutoML Conference*, 2023.

[Sto74] M. Stone. Cross-Validatory Choice and Assessment of Statistical Predictions. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 36(2):111–133, January 1974.

[Sut98] Richard S. Sutton. *Reinforcement Learning.* Adaptive Computation and Machine Learning. MIT Press, Cambridge, Mass, nachdruck edition, 1998.

[SVM14] C. O. S. Sorzano, J. Vargas, and A. Pascual Montano. A survey of dimensionality reduction techniques, March 2014.

[SZ19] Sherif Sakr and Albert Y. Zomaya, editors. *Encyclopedia of Big Data Technologies.* Springer International Publishing, Cham, 2019.

[SZB+19] Zeyuan Shang, Emanuel Zgraggen, Benedetto Buratti, Ferdinand Kossmann, Philipp Eichmann, Yeounoh Chung, Carsten Binnig, Eli Upfal, and Tim Kraska. Democratizing Data Science through Interactive Curation of ML Pipelines. In *Proceedings of the 2019 International Conference on Management of Data*, SIGMOD '19, pages 1171–1188, New York, NY, USA, June 2019. Association for Computing Machinery.

[SZW+24] Zhenqian Shen, Yongqi Zhang, Lanning Wei, Huan Zhao, and Quanming Yao. Automated Machine Learning: From Principles to Practices, February 2024.

[TFZ+24] Zhiqiang Tang, Haoyang Fang, Su Zhou, Taojiannan Yang, Zihan Zhong, Cuixiong Hu, Katrin Kirchhoff, and George Karypis. AutoGluon-Multimodal (AutoMM): Supercharging Multimodal AutoML with Foundation Models. In *Proceedings of the Third International Conference on Automated Machine Learning*, pages 15/1–35. PMLR, October 2024.

[THHL13] Chris Thornton, Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 847–855, Chicago Illinois USA, August 2013. ACM.

[Tho23] ThomasMeissnerDS. ThomasMeissnerDS/BlueCast, 2023.

[TK15] Abhishek Thakur and Artus Krohn-Grimberghe. AutoCompete: A Framework for Machine Learning Competition. In *Proceedings of the 2nd International Conference on Machine Learning, Workshop on Automated Machine Learning*. arXiv, July 2015.

[Tom76] I. Tomek. Two Modifications of CNN. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6(11):769–772, November 1976.

[TWG+19] Anh Truong, Austin Walters, Jeremy Goodsitt, Keegan Hines, C. Bayan Bruss, and Reza Farivar. Towards Automated Machine Learning: Evaluation and Comparison of AutoML Approaches and Tools. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 1471–1479, November 2019.

[vdMH08] Laurens van der Maaten and Geoffrey Hinton. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008.

[vdMPv09] Laurens van der Maaten, Eric Postma, and Jaap van den Herik. Dimensionality Reduction: A Comparative Review. *Journal of Machine Learning Research*, 2009.

[VRS+22] Anton Vakhrushev, Alexander Ryzhkov, Maxim Savchenko, Dmitry Simakov, Rinchin Damdinov, and Alexander Tuzhilin. LightAutoML: AutoML Solution for a Large Financial Services Ecosystem, April 2022.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[VvRBT14] Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. OpenML: Networked science in machine learning. *ACM SIGKDD Explorations Newsletter*, 15(2):49–60, June 2014.

[Wat89] Christopher J. C. H. Watkins. *Learning from Delayed Rewards.* PhD thesis, King's College, 1989.

[WD92] Christopher J. C. H. Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8(3-4):279–292, May 1992.

[WFWW20] Weiyuan Wu, Lampros Flokas, Eugene Wu, and Jiannan Wang. Complaint-driven Training Data Debugging for Query 2.0. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, pages 1317–1334, June 2020.

[Wil72] Dennis L. Wilson. Asymptotic Properties of Nearest Neighbor Rules Using Edited Data. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-2(3):408–421, July 1972.

[Win14] William E. Winkler. Matching and record linkage. *WIREs Computational Statistics*, 6(5):313–325, July 2014.

[WL20] Steven Euijong Whang and Jae-Gil Lee. Data collection and quality challenges for deep learning. *Proceedings of the VLDB Endowment*, 13(12):3429–3432, August 2020.

[WMTH19] Marcel Dominik Wever, Felix Mohr, Alexander Tornede, and Eyke Hüllermeier. Automating Multi-Label Classification Extending ML-Plan. In *6th ICML Workshop on Automated Machine Learning*, 2019.

[WRSL23] Steven Euijong Whang, Yuji Roh, Hwanjun Song, and Jae-Gil Lee. Data collection and quality challenges in deep learning: A data-centric AI perspective. *The VLDB Journal*, 32(4):791–813, July 2023.

[WSM$^+$19] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding, February 2019.

[WWWZ21] Chi Wang, Qingyun Wu, Markus Weimer, and Erkang Zhu. FLAML: A Fast and Lightweight AutoML Library. In *Proceedings of Machine Learning and Systems 2021*, 2021.

[YAKU19] Chengrun Yang, Yuji Akimoto, Dae Won Kim, and Madeleine Udell. OBOE: Collaborative Filtering for AutoML Model Selection. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1173–1183, Anchorage AK USA, July 2019. ACM.

[YFWU20] Chengrun Yang, Jicong Fan, Ziyang Wu, and Madeleine Udell. AutoML Pipeline Selection: Efficiently Navigating the Combinatorial Space. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, pages 1446–1456, New York, NY, USA, August 2020. Association for Computing Machinery.

[Yin19] Xue Ying. An Overview of Overfitting and its Solutions. *Journal of Physics: Conference Series*, 1168:022022, February 2019.

[YLW20] Jian Yang, Xuefeng Li, and Haifeng Wu. Hypernets: A General Automated Machine Learning Framework. DataCanvas, 2020.

[YMM$^+$20] Anatoly Yakovlev, Hesam Fathi Moghadam, Ali Moharrer, Jingxiao Cai, Nikan Chavoshi, Venkatanathan Varadarajan, Sandeep R. Agrawal, Sam

Idicula, Tomas Karnagel, Sanjay Jinturkar, and Nipun Agarwal. Oracle AutoML. *Proceedings of the VLDB Endowment*, 13(12):3166–3180, August 2020.

[ZBL+23] Daochen Zha, Zaid Pervaiz Bhat, Kwei-Herng Lai, Fan Yang, and Xia Hu. Data-centric AI: Perspectives and Challenges. In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*, Proceedings, pages 945–948. Society for Industrial and Applied Mathematics, January 2023.

[ZBL+25] Daochen Zha, Zaid Pervaiz Bhat, Kwei-Herng Lai, Fan Yang, Zhimeng Jiang, Shaochen Zhong, and Xia Hu. Data-centric Artificial Intelligence: A Survey. *ACM Computing Surveys*, 57(5):1–42, January 2025.

[ZD07] Patrick Ziegler and Klaus R. Dittrich. Data Integration — Problems, Approaches, and Perspectives. In John Krogstie, Andreas Lothe Opdahl, and Sjaak Brinkkemper, editors, *Conceptual Modelling in Information Systems Engineering*, pages 39–58. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[ZH21] Marc-André Zöller and Marco F. Huber. Benchmark and Survey of Automated Machine Learning Frameworks. *Journal of Artificial Intelligence Research*, 70:409–472, January 2021.

[ZLH21] Lucas Zimmer, Marius Lindauer, and Frank Hutter. Auto-Pytorch: Multi-Fidelity MetaLearning for Efficient and Robust AutoDL. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(9):3079–3090, September 2021.

[ZSDSS+21] Fernando Rezende Zagatti, Lucas Cardoso Silva, Lucas Nildaimon Dos Santos Silva, Bruno Silva Sette, Helena de Medeiros Caseli, Daniel Lucrédio, and Diego Furtado Silva. MetaPrep: Data preparation pipelines recommendation via meta-learning. In *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1197–1202, December 2021.

[ZVP+19] Jinan Zhou, Andrey Velichkevich, Kirill Prosvirov, Anubhav Garg, and Yuji Oshima. Katib: A Distributed General AutoML Platform on Kubernetes. *Proceedings of the 2019 USENIX Conference on Operational Machine Learning (OpML '19)*, 2019.

[ZZT+20] Xiang Zhao, Weixin Zeng, Jiuyang Tang, Wei Wang, and Fabian Suchanek. An Experimental Study of State-of-the-Art Entity Alignment Approaches. *IEEE Transactions on Knowledge and Data Engineering*, 34(6):1–1, 2020.

A ML Terminology

Machine learning	*Machine learning (ML)* is a subfield of artificial intelligence and of data science, focused on developing methods that can learn patterns from data in order to make predictions.
(Learning) Algorithm	A learning *algorithm* is the underlying procedure or method used to learn from data. Some common types of learning algorithms are neural networks, random forest, and k-nearest neighbors. In some papers, it is also called *inducer*.
Hyperparameter	A *hyperparameter* is an algorithm configuration variable that controls one of several aspects of the learning process (e.g., the learning rate—the magnitude of learning steps, which controls how fast new information overrides old information).
(ML) Model	A machine learning *model* is a system trained using a learning algorithm. It retains learned patterns and can use them to make predictions.
Inference	*Inference* is the process through which a trained model makes predictions on new data.
Task	A machine learning *task* is a specific problem to be solved by an ML model. Tasks can belong to different learning paradigms: supervised learning, where the model's training is guided by some known outcomes (e.g., regression—predicting values, or classification—affecting input data to known categories); unsupervised learning, where the model identifies patterns without external guidance (e.g., clustering—identifying groups of similar elements within the data); and others.
Dataset	A *dataset* is a structured collection of data, typically organized (for tabular data) into rows and columns.
Data point	A *data point* is a single observation, or instance of information. Data points usually correspond to the rows of a tabular dataset.
Feature	A *feature* is a property or variable describing data points. Features usually correspond to the columns of a tabular dataset. Features can belong to one of several types, such as numeric (e.g., age) or categorical (e.g., gender).
Target variable / label	A *target variable*, or *label*, is the outcome variable that a machine learning model is trained to predict.
Training set	A *training* dataset is a subset of data used to train a machine learning model. It is the dataset that the model learns on.
Test set	A *test* dataset is a subset of data used to evaluate the performance of a trained model after training is complete. It is kept separate from the training dataset, and is not used for learning.

Validation set	A *validation* dataset is an optional subset of data that can be used to evaluate the performance of a model during training, in order to adjust hyperparameters. It is kept separate from the training and test datasets, and is not used for learning.
Loss	A *loss* function is a mathematical function, or metric, that quantifies the performance of an ML model by measuring the difference between predicted values and labels representing known (real) values. A frequently used loss function for regression is mean squared error; cross-entropy is a common one for classification.
Data preparation	*Data preparation* is the process of making raw data suitable and optimized for machine learning through the application of data transformation operations.
Pipeline	A *pipeline* is a sequence of steps applied within the machine learning workflow, which can include elements of data preparation, modeling, and evaluation.

Table 4: Definitions of fundamental ML concepts relevant to data preparation