# Demystifying Artificial Intelligence for Data Preparation

Chengliang Chai
Beijing Institute of Technology
Beijing, China
ccl@bit.edu.cn

Nan Tang
QCRI / HKUST (GZ)
Qatar / China
ntang@hbku.edu.qa

Ju Fan
Renmin University of China
Beijing, China
fanj@ruc.edu.cn

Yuyu Luo
Tsinghua University
Beijing, China
luoyy18@mails.tsinghua.edu.cn

## ABSTRACT

Data preparation – the process of discovering, integrating, transforming, cleaning, and annotating data – is one of the oldest, hardest, yet inevitable data management problems. Unfortunately, data preparation is known to be iterative, requires high human cost, and is error-prone. Recent advances in artificial intelligence (AI) have shown very promising results on many data preparation tasks. At a high level, AI for data preparation (AI4DP) should have the following abilities. First, the AI model should capture real-world knowledge so as to solve various tasks. Second, it is important to easily adapt to new datasets/tasks. Third, data preparation is a complicated pipeline with many operations, which results in a large number of candidates to select the optimum, and thus it is crucial to effectively and efficiently explore the large space of possible pipelines. In this tutorial, we will cover three important topics to address the above issues: demystifying *foundation models* to inject knowledge for data preparation, tuning and adapting *pre-trained language models* for data preparation, and orchestrating *data preparation pipelines* for different downstream applications.

## CCS CONCEPTS

• **Computing methodologies → Machine learning**; **Artificial intelligence**; • **Information systems → Data cleaning**; **Information integration**.

## KEYWORDS

data preparation; artificial intelligence; foundation models

**ACM Reference Format:**
Chengliang Chai, Nan Tang, Ju Fan, and Yuyu Luo. 2023. Demystifying Artificial Intelligence for Data Preparation. In *Companion of the 2023 International Conference on Management of Data (SIGMOD-Companion '23), June 18–23, 2023, Seattle, WA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3555041.3589406

## 1 INTRODUCTION

An oft-cited statistic [11, 21] is that data scientists spend at least 80% of their time on data preparation, including discovering data sets from a large data repository such as data warehouses and data lakes [30, 31, 59, 61], cleaning the data set by correcting erroneous values or imputing missing values [1, 8, 9, 20], integrating the discovered data sets from multiple sources into a single and unified data set, enriching a data set with other data sets [25, 26, 56], understanding the data set through exploration and visualization [51–54, 65–67], selecting appropriate features among all features [10, 83] and conducting some transformations [15, 49], transforming the data into a uniform representation [22, 36, 39], and labeling raw data into a form suitable for machine learning [27, 46].

Despite decades of efforts from both academia [71] and industries [19, 89], data preparation is still one of the most time consuming and least enjoyable work of data scientists. For all data science applications, data preparation plays an important role so as to fully unlock the value of big data.

Traditionally, it requires to manually orchestrate these data preparation operations, each of which usually takes the lion's share of time and efforts of experts to achieve high-quality results. Recently, to alleviate this issue, artificial intelligence (AI) powered data preparation, especially using deep learning models [77], has shown promising results that significantly improve the performance of many data preparation tasks. We believe that AI will bring unique opportunities for data preparation. However, AI for data preparation (or AI4DP for short) faces the following challenges.

*(C1) Incorporating real-world knowledge:* AI4DP needs to capture the real-world knowledge through learning from large corpora.

*(C2) Adapting to new datasets/tasks:* when new datasets/tasks with different distributions come, AI4DP should quickly adjust to them, rather than learning from scratch.

*(C3) Exploring large search space:* data preparation corresponds to a complex pipeline, which results in a large number of candidate solutions, and thus it is necessary to quickly prune ineffective pipelines and find the optimal one.

In this tutorial, we choose three important topics, which are extensively studied to address the above challenges and lead to better solutions for many data preparation tasks, as shown in Figure 1. First, through learning from very large corpus, foundation models are injected with plenty of knowledge such that they can be directly applied to various tasks without much fine-tuning. Second, when it comes to new data/tasks, we can fine-tune the pre-trained language models (PLMs), which are pre-trained with large corpus, with task-specific labeled examples. Third, we investigate how to use trial-and-error strategies to explore the large space of data preparation pipelines. Next, we provide an overview of these three research topics of AI4DP.

**Foundation Models.** Foundation models [5] (*e.g.,* OpenAI's GPT-3 [7], AI21 lab's Jurassic-1 [70], NVIDIA's NeMo [42]) are giant language models trained on broad data and considered to have world knowledge, such that these foundation models can be used

*Ju Fan is the corresponding author.

*SIGMOD-Companion '23, June 18–23, 2023, Seattle, WA, USA*

ACM ISBN 978-1-4503-9507-6/23/06…$15.00
https://doi.org/10.1145/3555041.3589406

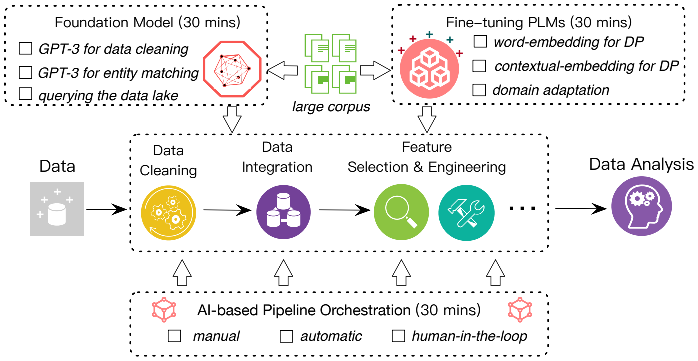
Foundation Model (30 mins)
□ GPT-3 for data cleaning
□ GPT-3 for entity matching
□ querying the data lake
large corpus
Fine-tuning PLMs (30 mins)
□ word-embedding for DP
□ contextual-embedding for DP
□ domain adaptation
Data
Data Cleaning
Data Integration
Feature Selection & Engineering
...
Data Analysis
AI-based Pipeline Orchestration (30 mins)
□ manual
□ automatic
□ human-in-the-loop

**Figure 1: AI for Data Preparation.**

for many downstream applications. These foundation models have widespread deployment, such that users can access these foundation models via standard APIs. In particular, users have choices of either zero-shot learning (*i.e.,* no example) or few-shot learning (*i.e.,* with few examples); and users can write different textual prompts.

We will introduce how these foundation models work for different data preparation problems, such as data cleaning and entity resolution [58]. We will demonstrate using GPT-3 hosted on Microsoft Azure for various data preparation tasks. Based on these analysis and examples, we will discuss the opportunities and limitations of foundation models.

Next, we will discuss some recent architectures that address the limitations of foundation models. Jurassic-X [40], AI21 Labs' MRKL (pronounced "miracle") system implementation, employs a modular and neuro-symbolic architecture that routes a natural language query to a module that can best respond to the input, where a module could be a language model, a math calculator, a currency converter, or an API call to a database. Retro [6], a system from Google DeepMind, improves language models by retrieving from trillions of tokens, where these tokens are explicit document chunks not knowledge implicitly embedded in foundation models.

Symphony [14] is along the same line of Jurassic-X and Retro but is highly customized to answer natural language queries over multi-modal data lakes.

**Fine-tuning PLMs.** Recently, PLMs have attracted significant attentions due to the good performance on various natural language processing (NLP) tasks. Typically, they are constructed by deep neural networks associated with multiple transformer layers such as BERT [24], BART [45], RoBERTa [50], and T5 [68], which are trained over extremely large corpora like Wikipedia. To train such models, self-training is always applied to mask some tokens or sentences and predict them considering the semantics of context in an unsupervised manner. Different from foundation models that can be directly used via APIs, we always have to fine-tune the pre-trained language models customized to specific tasks.

In this tutorial, we will cover various topics of data preparation such as blocking, entity matching, scheme matching that are well solved by fine-tuning PLMs. To be specific, we first introduce how to leverage word embeddings obtained from PLMs to solve the problem of blocking, entity matching and schema mathching. The key idea is to learn a good representation for each entity based on the word embeddings such that the more similar a pair of entities are, the more likely they are matching. However, purely relying on word embeddings requires a large amount of training examples, which are often expensive to obtain. Therefore, we will next introduce how to regard transformer-based PLMs as pre-training, and fine-tune data preparation tasks with a relatively small number of training examples. Besides, based on the PLMs, we can also well handle the domain shift between different data preparation tasks, which is a common case in practice (*e.g.,* entity matching tasks of the paper domain and restaurant domain). In this way, the task of one domain can be well adapted to another one by using just a small number of tuples. Finally, we will introduce how to utilize the PLMs to uniformly support data matching tasks like entity matching, schema matching, entity linking, ontology matching, etc.

**Data preparation pipeline orchestration.** In practice, the process of data preparation is typically composed of a series of steps, such as data transformation, data cleaning and feature engineering, which naturally form a *data preparation pipeline*. These data preparation pipelines are indispensable to a wide range of tasks, such as machine learning (ML) and exploratory data analysis (EDA), as they turn raw data into a format that is ready for downstream tasks. However, orchestrating a good data preparation pipeline is highly challenging for data scientists, not only because they have to explore a large and complex search space, but also because the performance of each pipeline is domain-specific, or even dataset-specific. To address the challenge, many approaches have been proposed to support data preparation pipeline orchestration.

In this tutorial, we will provide a taxonomy that categorizes the existing approaches into *manual pipeline orchestration*, *automatic pipeline generation* and *human-in-the-loop pipeline generation*. For manual pipeline orchestration, we will provide a statistical analysis of real-world pipelines, which are either from public platforms or enterprises. For automatic pipeline generation, we will discuss how to apply learning-based approaches, such as Bayesian optimization and meta-learning, genetic programming and reinforcement learning algorithms, to explore the search space to find optimized pipelines. Finally, we will introduce human-in-the-loop pipeline generation that attempts to provide a dedicate balance between human control and automation.

**Open Problems.** At the end of each topic discussed above, we will dedicate 5 minutes to open problems, paying particular attention to the data problems that are largely not tackled but should be solved in the research of AI for data preparation.

## 2 TARGET AUDIENCE AND LENGTH

**Target Audience.** The intended audience include SIGMOD attendees from both research and industry communities that are interested in either existing tools/techniques on AI4DP or promising research directions on AI4DP. We will not require any prior background knowledge in database or machine learning domain. The tutorial will be self-contained, and we will include a broad introduction and motivating examples for non-specialists to follow.

**Length.** The intended length of this tutorial is 1.5 hours. Each of the three topics will take 30 minutes.

## 3 TUTORIAL OUTLINE

We start with a brief overview of this tutorial, to give the audience a clear outline and talk goals. Within each topic, we will first give a crash course of used models, followed by their applications in different data preparation tasks. At the end of each topic, we will provide research challenges and open problems.

## 3.1 Foundation Models for Data Preparation

Recently, foundation models have attracted much attention because it has learned extensive knowledge that can benefit various downstream tasks without tuning much. Currently, foundation models have shown to be powerful in natural language processing tasks [7], and thus it is promising to explore whether they can benefit data preparation tasks. Therefore, in this tutorial, we will first generally introduce the foundation models, followed by how they can be leveraged to solve data preparation tasks and the limitations.

(1) **Foundation Models:** We will start by introducing basic concepts of foundation models, using running examples with GPT-3 [7].
  - Summary and sample applications of foundation models, including various NLP tasks, healthcare, education, biomedicine, etc.
  - Zero-shot vs. few-shot for foundation models, which means that foundation models can be applied on various tasks with no or very few labels.
  - Fine-tuninig foundation models, but it typically requires fewer labeled samples to achieve well-performed results.

*Take-away:* attendees will be familiar with foundation models, including their architectures, typical applications, and different optimization techniques.

(2) **Foundation Models for Data Preparation:** Given the basic concept, we will then discuss two typical data preparation tasks using GPT-3, data cleaning and entity matching [58, 75].
  - GPT-3 for data cleaning: we will discuss the effects of different prompts *i.e.,*, zero-shot vs. few-shot learning to solve the data cleaning task. Zero-shot uses the task description and example as prompt, and few-shot adds demonstrations of how to conduct the task.
  - GPT-3 for entity matching: similar to data cleaning, foundation models can also be applied to solve entity matching tasks, *i.e.,* answering Yes/No for a pair of entities almost purely relying on the models without training.
  - Limitation of foundation models.

*Take-away:* attendees will learn how to use foundation models for data cleaning and entity resolution, and understand their limitations such as lack of access to current information, lack of access to proprietary information sources, and lack of reasoning.

(3) **Lifting Limitations of Foundation Models:** We will provide concrete examples to show failure cases of GPT-3 and how recent architectures can tackle these problems.
  - Jurassic-X [40]: A modular, neuro-symbolic architecture. It adopts an extendable set of modules (both neural and symbolic), and a router that routes every incoming query input to a module that can best respond to the input.
  - Retro [6]: Retrieving from trillions of tokens. It enhances foundation models and language models by conditioning on data chunks retrieved from a large corpus, and then used the retrieved data to answer the given query.

*Take-away:* attendees will learn different solutions to tackle the inherent limitations of foundation models.

(4) **Foundation Models for Querying/Discovering Data Lakes:** Combining the merits of Jurassic-X and Retro, we will introduce a recent system Symphony [14] for querying/discovering data lakes using natural language.
  - Indexing data lakes in Symphony.
  - Query decomposition: how to decompose a complicated natural language query into sub-queries.
  - Retrieval: discover dataset (*e.g.,* tables or text) for each sub-query.
  - Route the discovered dataset and corresponding sub-query to a specific module, *e.g.,* a language model for question answering or a database for executing SQL queries using *e.g.,* TableQA model PASTA [35] or Text-to-SQL model SCPrompt [34].

*Take-away:* attendees will learn how foundation models can help to manage and query data lakes, which is very different from traditional data lake management systems such as Data Civilizer [21].

**Open Problems.** Foundation models are rapidly evolving, with newer and better models appearing each year. However, it is worth noting that these models are primarily designed to perform natural language generation tasks, which do not require high precision. On the other hand, many data preparation tasks, such as data cleaning and data transformation, require precise results. Hence, the inherent discrepancy between the main goals of foundation models and many data preparation tasks creates interesting open problems.

- *Neuro-symbolic AI.* How to tell foundation models explicit rules or constraints (*e.g.,* functional dependencies or check constraints) such that foundation models can better reason about data preparation tasks?
- *Explainable AI.* Whether foundation models can explain how a certain data preparation task is computed (*e.g.,* missing value imputation), *e.g.,* based on which data instances?
- *Human-centered AI.* Because foundation models cannot fully replace humans for data preparation tasks, an interesting problem is how to build AI-assistant based on foundation models that can significantly reduce human cost, *e.g.,* by providing top-$k$ possible repairs.

## 3.2 PLMs for Data Preparation

In the field of NLP, although various deep neural networks like recurrent neural network (RNN) and convolutional neural network (CNN) have been widely applied to improve the performance. However, since the neural networks always have plenty of parameters and the datasets of most tasks are not large enough, the trained models are likely to overfit and fail to generalize well. To overcome this problem, PLMs are designed to learn universal representations on large corpus, so as to benefit the downstream tasks. In this tutorial, we will first introduce the basic concept of the PLMs, followed by how to use it to solve data preparation tasks.

(1) **PLMs:** We will start by introducing basic concepts of PLMs, which can be generally classified into two categories.
  - The first-generation PLMs for non-contextual embeddings: they learn good embeddings without much considering

the downstream task, like Skip-Gram [55] and Glove [62], which are hard to capture high-level properties in context.

- The second-generation PLMs for contextual embeddings: they aim to learn contextual embeddings, which still need to consider downstream tasks such as BERT [23], ELMo [63] and GPT-3 [7]. Unlike the first generation, the embedding of a word changes dynamically based on the context.
- We will briefly introduce how to build the encoders of pre-trained models (convolutional models [41], recurrent models [38] or transformers [82]).
- How to fine-tune downstream tasks using pre-trained models.

*Take-away:* attendees will be familiar with pre-trained language models, their architectures, some typical pre-training tasks and how to tune them.

(2) **Word Embeddings for Data Preparation:** We will then discuss how to leverage the pre-trained word embeddings (the first-generation PLMs) to achieve a good performance on data preparation tasks.

- Entity matching: entities are represented by word embeddings of tokens [28, 57], which can be conducted by different encoders like RNN, attention-based models, etc.
- Blocking is the inevitable step before entity matching to quickly prune the large number of entity pairs that are highly not to be matched. Word embeddings can be utilized to hash non-matching entities to different hash blocks [28].
- Column type annotation [90] is to annotate the type (*e.g.,* `Name`, `Age`, `Company`) of each attribute in the relational table, which considers the embeddings of both attributes and cell values.

*Take-away:* attendees will learn how to leverage the word embeddings to capture the semantics of the tabular data, so as to improve the performance compared with traditional methods.

(3) **Contextual embeddings for Data Preparation:** Although the pre-trained embeddings can improve the performance, large amount of training examples are necessary and the data scientist has to design complicated neural networks. Hence, recent studies (the second-generation pre-trained model) mainly leverage transformer-based models to generate highly contextualized embeddings based on fine-tuning over dowbstream tasks.

- Entity matching: Ditto [47] regard entity matching as a sequence-pair classification problem using the transformer-based model. It also allows to inject domain knowledge to further improve the performance.
- Blocking: DeepBlocker [76] uses fastText [4], a pre-trained character-level embedding to block entity pairs.
- Column type annotation: Doduo [74] can be taken as a multi-task learning framework using pre-trained language models, so as to predict column types using a single model.

*Take-away:* attendees will learn how to leverage the architecture of pre-trained models to conduct data preparation tasks. In this way, they can fine-tune these tasks with relatively small number of training examples without designing complex neural networks.

(4) **Domain Adaptation:** Next, we will introduce domain adaptation that given a well-labeled source dataset, how to train a model for another target dataset by aligning features of both datasets based on the PLMs [79, 81]. Specifically, we will talk about three categories of adaptation methods using entity matching as an example.

- Discrepancy-based methods measure the alignment loss by computing distribution discrepancy between source and target datasets.
- Adversarial-based methods measure alignment loss by a domain classifier and generative adversarial networks.
- Reconstruction-based methods fulfill alignment loss by introducing an auxiliary unsupervised reconstruction task.

*Take-away:* attendees will learn how to handle the out-of-distribution problem by domain adaptation.

(5) **Unified Data Matching:** Furthermore, a recent study [80] has focused on how to leverage the PLMs to build a unified architecture to support common data matching tasks, like entity matching, entity linking, entity alignment, column type annotation, string matching, schema matching and ontology matching. The basic idea is to design a unified encoder for any pair of data to be matched, use a mixture-of-experts layer to align the matching semantics of various tasks and finally match the data.

*Take-away:* attendees will learn how to address various data matching tasks using a single unified model.

**Open Problems.** Finally, we will discuss some open problems about pre-trained language models for data preparation.

- *Automatic domain knowledge injection.* Existing works [47, 76] have demonstrated that injecting domain knowledge can significantly improve the performance of PLMs for data preparation tasks. However, it heavily relies on human efforts to specify domain rules and customize model architecture. Therefore, an interesting research question is, can we automatically identify and collect domain knowledge in the wild and encode various types of domain knowledge by designing a model-agnostic interface?
- *Data imputation.* Existing methods mainly investigate on tailoring pre-trained language models for supporting matching tasks (*e.g.,* entity matching) in data preparation but ignore the others (*e.g.,* data imputation). Can the contextual embeddings generated by PLMs benefit such tasks?
- *Domain-adaptive data augmentation.* We always need labeled data to fine-tune the PLMs for specific data preparation tasks. However, the model performance will degrade due to the lack of high-quality labeled data or domain shift between the training and testing sets. Therefore, a natural question is: can we synthesize labeled data by considering the domain adaption problem?

## 3.3 Orchestrating Data Preparation Pipelines

In previous sections, we discuss how to leverage sophisticated AI techniques to benefit each specific data preparation task, like entity matching, schema matching. However, real-world data preparation usually requires a series of steps, such as data wrangling (*e.g.,* joining multiple tables), data cleaning (*e.g.,* imputing missing values), and feature engineering (*e.g.,* reducing dimensionality via PCA). Thus, *data preparation pipelines* (or *pipelines* for short) arose to formalize the workflow of multiple steps where data moves from one step to its subsequent steps. For example, Scikit-Learn [72], the well-known Python machine learning library, allows users to *explicitly* define pipelines to assemble several steps using "`sklearn.pipeline`". On the other hand, data scientists can *implicitly* orchestrate pipelines by using functions from various toolkits or writing any functions based on their experiences.

Traditionally, the pipeline is always orchestrated by experts, which is time-consuming and hard to discover the optimal solution. In this tutorial, we will first highlight two main challenges for data scientists in orchestrating good data preparation pipelines.

- *Large and complex search space.* Even a simple pipeline is composed of several steps, where each step can be implemented by different algorithms (called *operators* in this tutorial). Moreover, there many be complex dependencies among operators. For example, the choices of missing value imputation may help some feature engineering operators, while harming others. Therefore, it is very challenging to explore such a large combinatorial space.
- *Domain- or even dataset-specific optimization.* The performance of a pipeline, or even each operator, heavily depends on the downstream tasks and distribution of underlying datasets. Therefore, there is no pipelines dominating others, and, given a new task, data scientists have to repeatedly evaluate different pipelines, which is time- and effort-consuming.

We will introduce some state-of-the-art frameworks and tools to generate data preparation pipeline with the help of AI, which can be broadly categorized into *manual pipeline orchestration*, *automatic pipeline generation* and *human-in-the-loop pipeline generation.*

(1) **Manual Pipeline Orchestration:** Traditionally, data scientists orchestrate pipelines manually based on their experiences. Thus, it is desirable to investigate human-generated pipelines, which will shed light on the pros and cons of manual pipeline orchestration. To this end, we will introduce some recent studies [44, 64, 85, 86] on analyzing real-world pipelines, which are extracted from either public platforms (*e.g.,* Github and OpenML) or enterprises (*e.g.,* Microsoft and Google). We will focus the following aspects.
   - *Operator-level:* To understand what operations that data scientists commonly use for data preparation, we will provide a categorization of different operators, including data wrangling, data cleaning and feature transformation, and discuss the usages of the operators in real-world pipelines.
   - *Pipeline-level:* We will analyze the pros and cons of human-generated pipelines. *Pros:* these pipelines are very flexible and can be easily injected with domain knowledge and user experiences. *Cons:* human orchestrated pipelines may have "blind spots" as data scientists may not be aware of sophisticated and useful operators, such as `PolynomialFeatures` provided by Scikit-Learn [72], and they rarely try different combinations of operators to find the best one.

Take-away: attendees will be familiar with human-orchestrated pipelines, their course-grained and fine-grained statistical characteristics, and the advantages and limitations of these pipelines.

(2) **Automatic Pipeline Generation:** Recent advances in deep learning have boosted extensive research on *automatic pipeline generation.* The basic idea is to first define a combinatorial space of operators with different functionalities, and then apply deep learning techniques to judiciously explore the search space and find the optimized pipelines. Based on the applied deep learning methods, the existing approaches can be divided into three categories.
   - *Bayesian optimization and meta-learning:* Some recent studies have extended Automated machine learning (AutoML) from hyper-parameter tuning and neural architecture search to an attempt of generating end-to-end pipelines. Auto-WEKA [78] utilizes *Bayesian optimization*, a well-adopted optimization technique in AutoML, to find optimized pipelines, which may be expensive to evaluate many pipelines. To address the problem, Auto-Sklearn [32, 33] and TensorOBOE [88] further develop *meta-learning* approaches to first find promising pipelines that work well in "similar" datasets, and then use Bayesian optimization to perform fine-grained evaluation on the pipelines. The key technical challenge here is to develop a *surrogate model* to predict the performance of pipelines based on dataset characteristics and domain-specific tasks. Moreover, Alpine Meadow further proposes an exploitation-exploration strategy for effectively searching pipelines [73]. We will systemically compare the existing approaches in this tutorial.
   - *Genetic programming:* TPOT [60] introduces a tree-based representation model of data preparation pipelines, and optimizes the pipelines using *genetic programming.* Specifically, given a new dataset, TPOT initially generates a number of random tree-based pipelines, and then uses the standard genetic programming algorithm [43] to find the pipelines that achieve high performance on the dataset.
   - *Reinforcement learning:* As the previous approaches may incur high computational cost, some recent studies develop reinforcement learning solutions to balance quality and efficiency of pipeline generation [3, 29, 37]. The basic idea is to model pipeline generation as a Markov Decision Process, where an *agent* takes a current pipeline as state and performs an action of selecting an operator to update the pipeline. When a pipeline is generated, the agent will get a reward from the *environment*, based on which the agent updates its policy. We will introduce the detailed design of the existing reinforcement learning-based methods, such as Learn2Clean [3], Deepline [37], and ATENA [29].

Take-away: attendees will learn how to apply Bayesian optimization and meta-learning, genetic programming and reinforcement learning algorithms to generate data preparation pipelines, and understand the pros and cons of automatic pipeline generation.

(3) **Human-in-the-loop Pipeline Generation:** It is intuitive to combine *human control* and *automation* to orchestrate better pipelines that take advantages of both approaches.

Based on this idea, there are some recent studies that attempt to support *human-in-the-loop* pipeline generation.

- *Recommendation-based approaches:* One way to involve human is to recommend candidate pipelines that the users can choose from, and then updates the recommendation based on users' feedback. DORIAN [69] adopts such an approach to suggest users *relevant* pipelines that are previously run on similar datasets and tasks. Auto-Suggest [87] employs deep learning models (*e.g.,* RNN) to recommend the next data preparation operators, such as Join, Pivot, Groupby, and Relationalize JSON. Auto-Pipeline [89] extends Auto-Suggest to recommend full pipelines that transform input tables to a user-specified "target" table.
- *Combination-based approaches:* A recent study [13] introduces an approach HAIPipe that combines human-orchestrated and automatic-generated pipelines. The basic idea is not only to leverage real-world human-orchestrated pipelines to incorporate specific domain knowledge, but also to combine optimized automatic-generated pipelines to eliminate the blind spots of each other. Experiments on real-world data preparation pipelines (*i.e.,* Jupyter notebooks from Kaggle) show HAIPipe can significantly improve the performance.
- *Program synthesis approaches:* Program synthesis with few-shot learning, such as OpenAI's Codex [12] and GitHub's Copilot [18], is a promising direction for human-in-the-loop pipeline generation. It utilizes large-scale language models (such as GPT-3) to learn from previous software documents and public code repositories, and can interact with users to suggest the next lines of code based on the context the users are working in. Moreover, the users have the flexibility of controlling the suggestion by writing natural language comments. In this tutorial, we will discuss the impressive performance of this approach, as well as their limitations on understanding the specific datasets.

Take-away: attendees will understand that a data preparation task should be solved by a dedicate balance between human control and automation, and learn how the existing recommendation-based, combination-based and program synthesis approaches support human-in-the-loop pipeline generation.

**Open problems.** We have a key observation that human-orchestrated and automatic-generated pipelines are *complementary.* Specifically, human-orchestrated pipelines allow users to easily inject their domain knowledge, while automatic-generated pipelines are optimized by exploring a large search space. The observation suggests that it is promising to study the balance between human control and automation in data preparation pipeline orchestration, which will create some interesting open problems.

- *Search space refinement.* It is interesting to study how to utilize human guidance to not only constrict the search space of possible pipelines to avoid exhaustive search, but also to define operations that are specific to particular tasks.
- *Domain knowledge injection.* The contextual awareness, domain knowledge and experiences of human are important to data preparation, but they are not easy to be encoded in the pipeline orchestration process. Thus, an interesting problem is how to inject domain knowledge to automatic pipeline generation algorithms.
- *Smooth integration with AutoML.* The objective of AutoML is to automate the entire life cycle of machine learning. Although some AutoML methods consider data preparation pipeline generation, the proposed techniques are relative simple. Thus, an open problem is how to smoothly integrate pipeline generation with other AutoML tasks, such as hyper-parameter tuning and model selection.

## 4 RELATED TUTORIALS

Badaro and Papotti [2] gave a tutorial on Transformers for tabular data representation in VLDB 2022, which is more general purpose and can serve natural language inference, question answering, table retrieval, table metadata prediction, and data imputation. Conceptually, it has certain overlap of our second topic. However, we focus on more recent works that are not covered by their tutorial.

Li et al. [48] presented a tutorial on data augmentation for ML-driven data preparation and integration in VLDB 2022. Many solutions in this tutorial employ Transformer-based models. However, content-wise, the overlap with our tutorial is marginal.

Whang et al. [84] presented a tutorial on data collection and data quality for deep learning in VLDB 2020. They focus on how data management techniques (like data acquisition, data cleaning) can benefit deep learning, but our focus is how deep learning, especially pre-trained models can benefit data management tasks.

Xu et al. [16, 17] presented two tutorial in SIGMOD 2016 and VLDB 2016, mainly discussing qualitative data cleaning, without deep learning based solutions. Hence, readers who are interested in traditional data cleaning methods may refer to these tutorials.

In summary, our tutorial has marginal overlap with previous tutorials in major database conferences.

## 5 BIOGRAPHY

**Chengliang Chai** is an Associate Professor at School of Computer Science & Technology, Beijing Institute of Technology, China. He received his PhD from Tsinghua Universitity in 2020 and received the ACM China Doctoral Dissertation Award. Dr Chai's main research interests are data preparation and database system.

**Nan Tang** is a Scientist at Qatar Computing Research Institute, HBKU, Qatar. He has received the best paper award of VLDB 2010, and several papers have been invited to VLDBJ and TKDE as the best papers series. Dr Nan's main research interests are data preparation and data-centric AI.

**Ju Fan** is a Professor at the DEKE Lab, MOE China, and School of Information, Renmin University of China. He received his PhD from Tsinghua Universitity in 2013 and received the ACM China Rising Star Award. Dr Fan's main research interests are data preparation and database systems.

**Yuyu Luo** is a fifth-year PhD candidate at the Department of Computer Science, Tsinghua University, China. His research interests are data preparation and data visualization.

## ACKNOWLEDGEMENT

This work is supported by NSF of China (62102215, 62122090, 62072461) and Zhejiang Lab's International Talent Fund for Young Professionals.

# REFERENCES

[1] Z. Abedjan, X. Chu, D. Deng, R. C. Fernandez, I. F. Ilyas, M. Ouzzani, P. Papotti, M. Stonebraker, and N. Tang. Detecting data errors: Where are we and what needs to be done? *Proc. VLDB Endow.*, 9(12):993–1004, 2016.

[2] G. Badaro and P. Papotti. Transformers for tabular data representation: A tutorial on models and applications. *Proc. VLDB Endow.*, 15(12):3746–3749, 2022.

[3] L. Berti-Équille. Learn2clean: Optimizing the sequence of tasks for web data preparation. In L. Liu, R. W. White, A. Mantrach, F. Silvestri, J. J. McAuley, R. Baeza-Yates, and L. Zia, editors, *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 2580–2586. ACM, 2019.

[4] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. Enriching word vectors with subword information. *Trans. Assoc. Comput. Linguistics*, 5:135–146, 2017.

[5] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, E. Brynjolfsson, S. Buch, D. Card, R. Castellon, N. S. Chatterji, A. S. Chen, K. Creel, J. Q. Davis, D. Demszky, C. Donahue, M. Doumbouya, E. Durmus, S. Ermon, J. Etchemendy, K. Ethayarajh, L. Fei-Fei, C. Finn, T. Gale, L. Gillespie, K. Goel, N. D. Goodman, S. Grossman, N. Guha, T. Hashimoto, P. Henderson, J. Hewitt, D. E. Ho, J. Hong, K. Hsu, J. Huang, T. Icard, S. Jain, D. Jurafsky, P. Kalluri, S. Karamcheti, G. Keeling, F. Khani, O. Khattab, P. W. Koh, M. S. Krass, R. Krishna, R. Kuditipudi, and et al. On the opportunities and risks of foundation models. *CoRR*, abs/2108.07258, 2021.

[6] S. Borgeaud, A. Mensch, J. Hoffmann, T. Cai, E. Rutherford, K. Millican, G. van den Driessche, J. Lespiau, B. Damoc, A. Clark, D. de Las Casas, A. Guy, J. Menick, R. Ring, T. Hennigan, S. Huang, L. Maggiore, C. Jones, A. Cassirer, A. Brock, M. Paganini, G. Irving, O. Vinyals, S. Osindero, K. Simonyan, J. W. Rae, E. Elsen, and L. Sifre. Improving language models by retrieving from trillions of tokens. *CoRR*, abs/2112.04426, 2021.

[7] T. B. Brown, B. Mann, and et al. Language models are few-shot learners. *CoRR*, abs/2005.14165, 2020.

[8] C. Chai, L. Cao, G. Li, J. Li, Y. Luo, and S. Madden. Human-in-the-loop outlier detection. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, pages 19–33, 2020.

[9] C. Chai, G. Li, J. Li, D. Deng, and J. Feng. Cost-effective crowdsourced entity resolution: A partial-order approach. In *SIGMOD*, pages 969–984, 2016.

[10] C. Chai, J. Liu, N. Tang, G. Li, and Y. Luo. Selective data acquisition in the wild for model charging. *PVLDB*, 15(7):1466–1478, 2022.

[11] C. Chai, J. Wang, Y. Luo, Z. Niu, and G. Li. Data management for machine learning: A survey. *TKDE*, 2022.

[12] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. de Oliveira Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, A. Ray, R. Puri, G. Krueger, M. Petrov, H. Khlaaf, G. Sastry, P. Mishkin, B. Chan, S. Gray, N. Ryder, M. Pavlov, A. Power, L. Kaiser, M. Bavarian, C. Winter, P. Tillet, F. P. Such, D. Cummings, M. Plappert, F. Chantzis, E. Barnes, A. Herbert-Voss, W. H. Guss, A. Nichol, A. Paino, N. Tezak, J. Tang, I. Babuschkin, S. Balaji, S. Jain, W. Saunders, C. Hesse, A. N. Carr, J. Leike, J. Achiam, V. Misra, E. Morikawa, A. Radford, M. Knight, M. Brundage, M. Murati, K. Mayer, P. Welinder, B. McGrew, D. Amodei, S. McCandlish, I. Sutskever, and W. Zaremba. Evaluating large language models trained on code. *CoRR*, abs/2107.03374, 2021.

[13] S. Chen, N. Tang, J. Fan, X. Yan, C. Chai, G. Li, and X. Du. Haipipe: Combining human-generated and machine-generated pipelines for data preparation. In *SIGMOD '23: International Conference on Management of Data, Seattle, WA, USA, June 18 - 23, 2023*, page to appear. ACM, 2023.

[14] Z. Chen, Z. Gu, L. Cao, J. Fan, S. Madden, and N. Tang. Symphony: Towards natural language query answering over multi-modal data lakes. In *Conference on Innovative Data Systems Research, CIDR 2023, Amsterdam, The Netherlands, January 8-151, 2023.* www.cidrdb.org, 2023.

[15] N. Chepurko, R. Marcus, E. Zgraggen, R. C. Fernandez, T. Kraska, and D. R. Karger. ARDA: automatic relational data augmentation for machine learning. *Proc. VLDB Endow.*, 13(9):1373–1387, 2020.

[16] X. Chu and I. F. Ilyas. Qualitative data cleaning. *Proc. VLDB Endow.*, 9(13):1605–1608, 2016.

[17] X. Chu, I. F. Ilyas, S. Krishnan, and J. Wang. Data cleaning: Overview and emerging challenges. In *Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, June 26 - July 01, 2016*, pages 2201–2206. ACM, 2016.

[18] G. Copilot. https://github.com/features/copilot.

[19] E. Cortez, P. A. Bernstein, Y. He, and L. Novik. Annotating database schemas to help enterprise search. *Proc. VLDB Endow.*, 8(12):1936–1939, 2015.

[20] M. Dallachiesa, A. Ebaid, A. Eldawy, A. K. Elmagarmid, I. F. Ilyas, M. Ouzzani, and N. Tang. NADEEF: a commodity data cleaning system. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2013, New York, NY, USA, June 22-27, 2013*, pages 541–552. ACM, 2013.

[21] D. Deng, R. C. Fernandez, Z. Abedjan, S. Wang, M. Stonebraker, A. K. Elmagarmid, I. F. Ilyas, S. Madden, M. Ouzzani, and N. Tang. The data civilizer system. In *8th Biennial Conference on Innovative Data Systems Research, CIDR 2017, Chaminade, CA, USA, January 8-11, 2017, Online Proceedings.* www.cidrdb.org, 2017.

[22] D. Deng, W. Tao, Z. Abedjan, A. K. Elmagarmid, I. F. Ilyas, G. Li, S. Madden, M. Ouzzani, M. Stonebraker, and N. Tang. Unsupervised string transformation learning for entity consolidation. In *35th IEEE International Conference on Data Engineering, ICDE 2019, Macao, China, April 8-11, 2019*, pages 196–207. IEEE, 2019.

[23] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019.

[24] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[25] A. Doan, A. Y. Halevy, and Z. G. Ives. *Principles of Data Integration.* Morgan Kaufmann, 2012.

[26] X. L. Dong, E. Gabrilovich, G. Heitz, W. Horn, K. Murphy, S. Sun, and W. Zhang. From data fusion to knowledge fusion. *CoRR*, abs/1503.00302, 2015.

[27] A. Drutsa, V. Fedorova, D. Ustalov, O. Megorskaya, E. Zerminova, and D. Baidakova. Crowdsourcing practice for efficient data labeling: Aggregation, incremental relabeling, and pricing. In *Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference 2020, online conference [Portland, OR, USA], June 14-19, 2020*, pages 2623–2627. ACM, 2020.

[28] M. Ebraheem, S. Thirumuruganathan, S. R. Joty, M. Ouzzani, and N. Tang. Distributed representations of tuples for entity resolution. *Proc. VLDB Endow.*, 11(11):1454–1467, 2018.

[29] O. B. El, T. Milo, and A. Somech. Automatically generating data exploration sessions using deep reinforcement learning. In D. Maier, R. Pottinger, A. Doan, W. Tan, A. Alawini, and H. Q. Ngo, editors, *Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference 2020, online conference [Portland, OR, USA], June 14-19, 2020*, pages 1527–1537. ACM, 2020.

[30] R. C. Fernandez, Z. Abedjan, F. Koko, G. Yuan, S. Madden, and M. Stonebraker. Aurum: A data discovery system. In *34th IEEE International Conference on Data Engineering, ICDE 2018, Paris, France, April 16-19, 2018*, pages 1001–1012. IEEE Computer Society, 2018.

[31] R. C. Fernandez, E. Mansour, A. A. Qahtan, A. K. Elmagarmid, I. F. Ilyas, S. Madden, M. Ouzzani, M. Stonebraker, and N. Tang. Seeping semantics: Linking datasets using word embeddings for data discovery. In *34th IEEE International Conference on Data Engineering, ICDE 2018, Paris, France, April 16-19, 2018*, pages 989–1000. IEEE Computer Society, 2018.

[32] M. Feurer, K. Eggensperger, S. Falkner, M. Lindauer, and F. Hutter. Auto-sklearn 2.0: The next generation. *CoRR*, abs/2007.04074, 2020.

[33] M. Feurer, A. Klein, K. Eggensperger, J. T. Springenberg, M. Blum, and F. Hutter. Auto-sklearn: Efficient and robust automated machine learning. In F. Hutter, L. Kotthoff, and J. Vanschoren, editors, *Automated Machine Learning - Methods, Systems, Challenges*, The Springer Series on Challenges in Machine Learning, pages 113–134. Springer, 2019.

[34] Z. Gu, J. Fan, N. Tang, L. Cao, B. Jia, S. Madden, and X. Du. Few-shot text-to-sql translation using structure and content prompt learning. 2023.

[35] Z. Gu, J. Fan, N. Tang, P. Nakov, X. Zhao, and X. Du. PASTA: table-operations aware fact verification via sentence-table cloze pre-training. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 4971–4983. Association for Computational Linguistics, 2022.

[36] J. Heer, J. M. Hellerstein, and S. Kandel. Predictive interaction for data transformation. In *Seventh Biennial Conference on Innovative Data Systems Research, CIDR 2015, Asilomar, CA, USA, January 4-7, 2015, Online Proceedings.* www.cidrdb.org, 2015.

[37] Y. Heffetz, R. Vainshtein, G. Katz, and L. Rokach. Deepline: Automl tool for pipelines generation using deep reinforcement learning and hierarchical actions filtering. In R. Gupta, Y. Liu, J. Tang, and B. A. Prakash, editors, *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 2103–2113. ACM, 2020.

[38] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[39] Z. Jin, M. J. Cafarella, H. V. Jagadish, S. Kandel, M. Minar, and J. M. Hellerstein. CLX: towards verifiable PBE data transformation. In *Advances in Database Technology - 22nd International Conference on Extending Database Technology, EDBT 2019, Lisbon, Portugal, March 26-29, 2019*, pages 265–276, 2019.

[40] E. Karpas, O. Abend, Y. Belinkov, B. Lenz, O. Lieber, N. Ratner, Y. Shoham, H. Bata, Y. Levine, K. Leyton-Brown, D. Muhlgay, N. Rozen, E. Schwartz, G. Shachaf, S. Shalev-Shwartz, A. Shashua, and M. Tenenholtz. Mrkl systems: A modular, neuro-symbolic architecture that combines large language models, external knowledge sources and discrete reasoning, 2022.

[41] Y. Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751. ACL, 2014.

[42] O. Kuchaiev, J. Li, H. Nguyen, O. Hrinchuk, R. Leary, B. Ginsburg, S. Kriman, S. Beliaev, V. Lavrukhin, J. Cook, P. Castonguay, M. Popova, J. Huang, and J. M. Cohen. Nemo: a toolkit for building AI applications using neural modules. *CoRR*,

abs/1909.09577, 2019.

[43] W. B. Langdon, R. Poli, N. F. McPhee, and J. R. Koza. Genetic programming: An introduction and tutorial, with a survey of techniques and applications. In *Computational Intelligence: A Compendium*, volume 115 of *Studies in Computational Intelligence*, pages 927–1028. Springer, 2008.

[44] A. Lee, D. Xin, D. Lee, and A. G. Parameswaran. Demystifying a dark art: Understanding real-world machine learning model development. *CoRR*, abs/2005.01520, 2020.

[45] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*, 2020.

[46] G. Li, C. Chai, J. Fan, X. Weng, J. Li, Y. Zheng, Y. Li, X. Yu, X. Zhang, and H. Yuan. Cdb: A crowd-powered database system. *Proceedings of the VLDB Endowment*, 11(12):1926–1929, 2018.

[47] Y. Li, J. Li, Y. Suhara, A. Doan, and W. Tan. Deep entity matching with pre-trained language models. *Proc. VLDB Endow.*, 14(1):50–60, 2020.

[48] Y. Li, X. Wang, Z. Miao, and W. Tan. Data augmentation for ml-driven data preparation and integration. *Proc. VLDB Endow.*, 14(12):3182–3185, 2021.

[49] J. Liu, C. Chai, Y. Luo, Y. Lou, J. Feng, and N. Tang. Feature augmentation with reinforcement learning. In *38th IEEE International Conference on Data Engineering, ICDE 2022, Kuala Lumpur, Malaysia, May 9-12, 2022*, pages 3360–3372. IEEE, 2022.

[50] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[51] Y. Luo, X. Qin, C. Chai, N. Tang, G. Li, and W. Li. Steerable self-driving data visualization. *IEEE Trans. Knowl. Data Eng.*, 34(1):475–490, 2022.

[52] Y. Luo, X. Qin, N. Tang, and G. Li. Deepeye: Towards automatic data visualization. In *34th IEEE International Conference on Data Engineering, ICDE 2018, Paris, France, April 16-19, 2018*, pages 101–112. IEEE Computer Society, 2018.

[53] Y. Luo, N. Tang, G. Li, C. Chai, W. Li, and X. Qin. Synthesizing natural language to visualization (NL2VIS) benchmarks from NL2SQL benchmarks. In *SIGMOD '21: International Conference on Management of Data, Virtual Event, China, June 20-25, 2021*, pages 1235–1247. ACM, 2021.

[54] Y. Luo, N. Tang, G. Li, J. Tang, C. Chai, and X. Qin. Natural language to visualization by neural machine translation. *IEEE Trans. Vis. Comput. Graph.*, 28(1):217–226, 2022.

[55] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 3111–3119, 2013.

[56] R. J. Miller. Open data integration. *Proc. VLDB Endow.*, 11(12):2130–2139, 2018.

[57] S. Mudgal, H. Li, T. Rekatsinas, A. Doan, Y. Park, G. Krishnan, R. Deep, E. Arcaute, and V. Raghavendra. Deep learning for entity matching: A design space exploration. In *Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018*, pages 19–34, 2018.

[58] A. Narayan, I. Chami, L. Orr, and C. Ré. Can foundation models wrangle your data?, 2022.

[59] F. Nargesian, E. Zhu, R. J. Miller, K. Q. Pu, and P. C. Arocena. Data lake management: Challenges and opportunities. *Proc. VLDB Endow.*, 12(12):1986–1989, 2019.

[60] R. S. Olson and J. H. Moore. TPOT: A tree-based pipeline optimization tool for automating machine learning. In F. Hutter, L. Kotthoff, and J. Vanschoren, editors, *Proceedings of the 2016 Workshop on Automatic Machine Learning, AutoML 2016, co-located with 33rd International Conference on Machine Learning (ICML 2016), New York City, NY, USA, June 24, 2016*, volume 64 of *JMLR Workshop and Conference Proceedings*, pages 66–74. JMLR.org, 2016.

[61] P. Ouellette, A. Sciortino, F. Nargesian, B. G. Bashardoost, E. Zhu, K. Pu, and R. J. Miller. RONIN: data lake exploration. *Proc. VLDB Endow.*, 14(12):2863–2866, 2021.

[62] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543, 2014.

[63] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics, 2018.

[64] F. Psallidas, Y. Zhu, B. Karlas, J. Henkel, M. Interlandi, S. Krishnan, B. Kroth, K. V. Emani, W. Wu, C. Zhang, M. Weimer, A. Floratou, C. Curino, and K. Karanasos. Data science through the looking glass: Analysis of millions of github notebooks and ML.NET pipelines. *SIGMOD Rec.*, 51(2):30–37, 2022.

[65] X. Qin, C. Chai, Y. Luo, N. Tang, and G. Li. Interactively discovering and ranking desired tuples without writing sql queries. In *SIGMOD*, pages 2745–2748, 2020.

[66] X. Qin, C. Chai, Y. Luo, T. Zhao, N. Tang, G. Li, J. Feng, X. Yu, and M. Ouzzani. Ranking desired tuples by database exploration. In *ICDE*, pages 1973–1978. IEEE, 2021.

[67] X. Qin, Y. Luo, N. Tang, and G. Li. Making data visualization more efficient and effective: a survey. *VLDB J.*, 29(1):93–117, 2020.

[68] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020.

[69] S. Redyuk, Z. Kaoudi, S. Schelter, and V. Markl. DORIAN in action: Assisted design of data science pipelines. *Proc. VLDB Endow.*, 15(12):3714–3717, 2022.

[70] L. Reed, C. Li, A. Ramirez, L. Wu, and M. A. Walker. Jurassic is (almost) all you need: Few-shot meaning-to-text generation for open-domain dialogue. *CoRR*, abs/2110.08094, 2021.

[71] E. K. Rezig, L. Cao, M. Stonebraker, G. Simonini, W. Tao, S. Madden, M. Ouzzani, N. Tang, and A. K. Elmagarmid. Data civilizer 2.0: A holistic framework for data preparation and analytics. *Proc. VLDB Endow.*, 12(12):1954–1957, 2019.

[72] Scikit-Learn. https://scikit-learn.org/stable/.

[73] Z. Shang, E. Zgraggen, B. Buratti, F. Kossmann, P. Eichmann, Y. Chung, C. Binnig, E. Upfal, and T. Kraska. Democratizing data science through interactive curation of ML pipelines. In *Proceedings of the 2019 International Conference on Management of Data, SIGMOD Conference 2019, Amsterdam, The Netherlands, June 30 - July 5, 2019*, pages 1171–1188. ACM, 2019.

[74] Y. Suhara, J. Li, Y. Li, D. Zhang, Ç. Demiralp, C. Chen, and W. Tan. Annotating columns with pre-trained language models. In *SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, pages 1493–1503. ACM, 2022.

[75] J. Tang, Y. Zuo, L. Cao, and S. Madden. Generic entity resolution models. In *Proceedings of the Workshop on Table Representation Learning, TRL@NeurIPS, 2022*, 2022.

[76] S. Thirumuruganathan, H. Li, N. Tang, M. Ouzzani, Y. Govind, D. Paulsen, G. Fung, and A. Doan. Deep learning for blocking in entity matching: A design space exploration. *Proc. VLDB Endow.*, 14(11):2459–2472, 2021.

[77] S. Thirumuruganathan, N. Tang, M. Ouzzani, and A. Doan. Data curation with deep learning. In *Proceedings of the 23rd International Conference on Extending Database Technology, EDBT 2020, Copenhagen, Denmark, March 30 - April 02, 2020*, pages 277–286. OpenProceedings.org, 2020.

[78] C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown. Auto-weka: Automated selection and hyper-parameter optimization of classification algorithms. *CoRR*, abs/1208.3719, 2012.

[79] J. Tu, J. Fan, N. Tang, P. Wang, C. Chai, G. Li, R. Fan, and X. Du. Domain adaptation for deep entity resolution. In Z. Ives, A. Bonifati, and A. E. Abbadi, editors, *SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, pages 443–457. ACM, 2022.

[80] J. Tu, J. Fan, N. Tang, P. Wang, G. Li, X. Du, X. Jia, and S. Gao. Unicorn: A unified multi-tasking model for supporting matching tasks in data integration. In *SIGMOD '23: International Conference on Management of Data, Seattle, WA, USA, June 18 - 23, 2023*, page to appear. ACM, 2023.

[81] J. Tu, X. Han, J. Fan, N. Tang, C. Chai, G. Li, and X. Du. DADER: hands-off entity resolution with domain adaptation. *Proc. VLDB Endow.*, 15(12):3666–3669, 2022.

[82] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017.

[83] J. Wang, C. Chai, N. Tang, J. Liu, and G. Li. Coresets over multiple tables for feature-rich and data-efficient machine learning. *Proc. VLDB Endow.*, 16(1):64–76, 2022.

[84] S. Whang and J. Lee. Data collection and quality challenges for deep learning. *Proc. VLDB Endow.*, 13(12):3429–3432, 2020.

[85] D. Xin, H. Miao, A. G. Parameswaran, and N. Polyzotis. Production machine learning pipelines: Empirical analysis and optimization opportunities. In *SIGMOD '21: International Conference on Management of Data, Virtual Event, China, June 20-25, 2021*, pages 2639–2652. ACM, 2021.

[86] D. Xin, E. Y. Wu, D. J. L. Lee, N. Salehi, and A. G. Parameswaran. Whither automl? understanding the role of automation in machine learning workflows. In *CHI '21: CHI Conference on Human Factors in Computing Systems, Virtual Event / Yokohama, Japan, May 8-13, 2021*, pages 83:1–83:16. ACM, 2021.

[87] C. Yan and Y. He. Auto-suggest: Learning-to-recommend data preparation steps using data science notebooks. In D. Maier, R. Pottinger, A. Doan, W. Tan, A. Alawini, and H. Q. Ngo, editors, *Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference 2020, online conference [Portland, OR, USA], June 14-19, 2020*, pages 1539–1554. ACM, 2020.

[88] C. Yang, J. Fan, Z. Wu, and M. Udell. Automl pipeline selection: Efficiently navigating the combinatorial space. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 1446–1456. ACM, 2020.

[89] J. Yang, Y. He, and S. Chaudhuri. Auto-pipeline: Synthesize data pipelines by-target using reinforcement learning and search. *Proc. VLDB Endow.*, 14(11):2563–2575, 2021.

[90] D. Zhang, Y. Suhara, J. Li, M. Hulsebos, Ç. Demiralp, and W. Tan. Sato: Contextual semantic type detection in tables. *Proc. VLDB Endow.*, 13(11):1835–1848, 2020.