Assessing the Use of AutoML for Data-Driven Software Engineering

Fabio Calefato, Luigi Quaranta, Filippo Lanubile
University of Bari
Bari, Italy
{fabio.calefato, luigi.quaranta, filippo.lanubile}@uniba.it

Marcos Kalinowski
PUC-Rio
Rio de Janeiro, Brazil
kalinowski@inf.puc-rio.br

Abstract—Background. **Due to the widespread adoption of Artificial Intelligence (AI) and Machine Learning (ML) for building software applications, companies are struggling to recruit employees with a deep understanding of such technologies. In this scenario, AutoML is soaring as a promising solution to fill the AI/ML skills gap since it promises to automate the building of end-to-end AI/ML pipelines that would normally be engineered by specialized team members.** *Aims*. **Despite the growing interest and high expectations, there is a dearth of information about the extent to which AutoML is currently adopted by teams developing AI/ML-enabled systems and how it is perceived by practitioners and researchers.** *Method*. **To fill these gaps, in this paper, we present a mixed-method study comprising a benchmark of 12 end-to-end AutoML tools on two SE datasets and a user survey with follow-up interviews to further our understanding of AutoML adoption and perception.** *Results*. **We found that AutoML solutions can generate models that outperform those trained and optimized by researchers to perform classification tasks in the SE domain. Also, our findings show that the currently available AutoML solutions do not live up to their names as they do not equally support automation across the stages of the ML development workflow and for all the team members.** *Conclusions*. **We derive insights to inform the SE research community on how AutoML can facilitate their activities and tool builders on how to design the next generation of AutoML technologies.**

Index Terms—**AutoAI, benchmark, mixed-method study**

I. Introduction

The recent advancements in Artificial Intelligence (AI) and Machine Learning (ML) have led to their widespread industrial adoption in a variety of domains, such as automotive, business, and healthcare [1, 2]. Building well-performing AI-augmented applications requires more than just highly specialized human experts with a deep knowledge of AI/ML-related technologies. Working in specialized teams, data engineers, domain experts, statisticians, and software engineers help data scientists develop AI/ML-enabled systems by building end-to-end pipelines comprising a variety of stages, from data preprocessing and feature engineering to model integration, monitoring, and fine-tuning. However, companies are currently struggling to recruit AI/ML experts [3–5].

Given this severe AI/ML skills shortage, *automated machine learning* (AutoML) [6] has gained momentum as it aims to fill this gap by automatizing the building of end-to-end AI/ML pipelines that would normally be engineered by specialized teams. AutoML is in fact appealing to both companies lacking in-house expertise as well as those that already have specialists who thus benefit from automation when executing complex, time-consuming activities such as data exploration, feature engineering, and hyperparameter optimization. Lately, AutoML has also been employed in Software Engineering (SE) research; Tanaka et al. [7] experimented with AutoML to develop software defect prediction models, achieving state-of-the-art performances.

Notwithstanding the growing interest and high expectations, we have little knowledge regarding the extent to which AutoML is currently adopted by teams developing AI/ML-enabled systems. Vanschoren et al. [8] found that AutoML accounted for less than 2% of the workflows uploaded to OpenML,[1] an online platform collecting public ML datasets and benchmark results. Van der Blom et al. [9] conducted a survey with members of AI/ML teams, which revealed that about 20-30% of the respondents had never used AutoML.

Moreover, we have a limited grasp of how AutoML is perceived by practitioners. In particular, Wang et al. [10] have explored the perceptions of adopting AutoML, focusing on the perspective of data scientists and how they use it for building models and fine-tuning hyperparameters. Nonetheless, AutoML tools aim to cover the AI/ML workflow end to end, and little is still known about the perception and expectations of other team members. Although the boundaries are often blurred, a mature AI/ML team consists of several roles other than those pertaining to data scientists such as *data engineers*, who focus on setting up data preparation pipelines, and *ML engineers* (often referred to also as MLOps), who focus on engineering activities like model integration, deployment, and monitoring [11, 12]. In [13], Wang et al. analyzed several Stack Overflow questions and GitHub discussions to define a taxonomy of 26 developer-specific challenges when using AutoML. In addition, Majidi et al. [14] sampled ~22k GitHub projects adopting AutoML tools and found that they are mostly used for model training and evaluation.

Finally, there are several published benchmarks comparing the performance of AutoML systems (e.g., [15–17]); for example, research has reported that AutoML does not outperform conventional forecasting strategies used in time series analysis [18]. However, with the exception of Tanaka
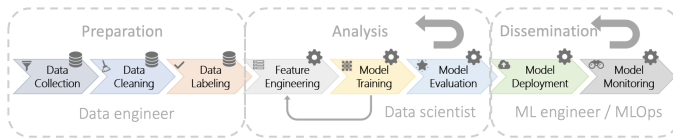
---

[1]https://openml.org

Fig. 1. A typical ML workflow with activities and roles (adapted from Amershi et al. [21]).

et al. [7], none of these benchmarks have leveraged datasets from the SE domain so far, notwithstanding the several potential applications of AutoML in this field.

To bridge these gaps, in this paper, we present a mixed-method study comprising (i) a benchmark of 12 end-to-end AutoML tools on two software engineering datasets, to collect initial evidence about the performance of state-of-the-art AutoML solutions when applied to SE-specific tasks – namely sentiment analysis from technical text – and (ii) a user survey with follow-up interviews with software engineers working on AI/ML projects, to further our understanding of their perception and extent of AutoML adoption.

Our contributions are appealing to both industry and the SE research community as, respectively, we (i) show what stages of a typical ML workflow are more/less suitable for automation according to SE practitioners and (ii) provide evidence that AutoML solutions may generate models that outperform those trained and optimized by researchers to perform classification tasks – such as sentiment analysis and emotion recognition in the SE domain – and at the same time reach performance close to complex and time-consuming solutions based on fine-tuning pre-trained transformers (e.g., BERT [19]).

The remainder of this paper is organized as follows. In Sect. II, we review background literature. In Sect. III, we illustrate the design of the mixed-method study. The results of the quantitative and qualitative analyses are reported and discussed in Sect. IV and V, respectively. Finally, we draw conclusions in Sect. VI.

II. BACKGROUND

A. Machine Learning Development Workflow

Fig. 1 presents a typical machine-learning workflow with activities and responsibilities. Compared to a generic, say agile, development workflow of a traditional software-engineered system two main differences arise. First, with AI/ML-enabled systems, the final output of the development workflow is not the code itself, but rather the models' outcome (e.g., the predictions made, the informed business decisions taken) [20]. The second difference pertains to the amount of experimentation needed to converge to a satisfying model performance for the problem at hand; in fact, albeit the workflow may appear linear, much like agile software processes, a machine learning workflow contains several feedback loops due to the frequent iterations involving model selection, hyper-parameters tuning, and dataset refinement [21].

Preparation. The data preparation stage is typically the responsibility of data engineers. If no data is already available, the first step in the preparation process is *data collection*, where data engineers retrieve raw pieces of information from multiple data sources and then integrate them into a dataset. Once acquired, the next step to perform with the dataset is *data cleaning*, which involves removing inaccurate or noisy records, a common activity in all forms of data science. The typical preprocessing steps regard the imputation of missing values as well as removing duplicates and inconsistent values. The last step in data preparation is *data labeling*, the time-intensive task of assigning ground truth labels to each record in a dataset since most of the supervised learning techniques require 'gold' labels to be able to build a model. This step refers to both 'assisted data labeling' and 'manual data labeling,' depending on whether ML is used to complete the annotation and save time and budget, as compared to only using human annotators (e.g., data engineers themselves, domain experts, and crowd workers) to label the entire dataset.

Analysis. This stage is the core of a machine learning development workflow, which typically pertains to data scientists who perform several tasks to eventually build and release a validated model. The first step in the analysis process is *feature engineering*, which refers to all those activities performed to extract and select from the dataset the most informative features for building ML models. The typical feature engineering tasks include normalizing and discretizing numeric features, creating one-hot encoding and embeddings for categorical features, basic processing such as tokenization for text features, and extracting date- and time-related features from timestamped columns. For deep learning models (e.g. convolutional neural networks), feature engineering is less explicit and often blended with the next stage. During *model training* the chosen models are trained on the annotated dataset, using the selected features. Once built, models are evaluated in the *model evaluation* using predefined performance metrics (e.g., log-loss, accuracy, AUC). This is a critical step that typically involves extensive human evaluation, with data scientists assessing the output models to ensure they meet the required performance criteria.

Dissemination. The final stage of the machine learning development workflow typically pertains to ML engineers and software developers. *Model deployment* is the step where the evaluated machine learning model is integrated into targeted production environments to make practical predictions or business decisions based on new data. During *model monitoring*, engineers continuously monitor the deployed models for possible errors and decrease in performance during real-world execution (e.g., model drift).

B. AutoML

AutoML is the portmanteau of *Automated* and *Machine Learning*. The term is generally used to refer to solutions aiming at democratizing ML to non-experts by easing the difficulty of developing ML models and, at the same time, drastically increasing the productivity of experts.

AutoML has been gaining a lot of momentum in recent years, especially after the release of commercial solutions from large companies such as Amazon, Microsoft, and IBM. Despite the recent interest, AutoML is not entirely a new

trend because, since the 1990s, many solutions have been proposed for automatic Combined Algorithm Selection and Hyperparameter (CASH) optimization [17]. With respect to the workflow depicted in Fig. 1, AutoML solutions typically focus on automating the 'core' tasks of the Analysis stage (i.e., feature engineering, model training, and evaluation), with less attention devoted to preparation and dissemination (we further discuss this in Sect. III-A, where we describe the tools selected for our analysis).

Research in the area of AutoML has also gained significant traction leading to many improvements in terms of both performance and reduced execution runtime [6]. Researchers have published several benchmarks that compare the performance of models trained on datasets from various domains using AutoML solutions. The reported findings are varied. Paldino et al. [18] found that AutoML does not outperform conventional forecasting strategies used in time series analysis. Hanussek et al. [22] used 4 AutoML tools to perform regression and classification tasks on 12 different popular datasets from OpenML and found that the automated frameworks perform better than or equal to the machine learning community in 7 out of 12 cases. Ferreira et al. [23] report similar findings in a study comparing the performance of 8 open-source AutoML tools on 12 OpenML datasets. In a similar study conducted on a broader set of 57 classification datasets and 30 regression datasets retrieved from OpenML, Balaji and Allen [24] report high variances in the results. Gijsbers et al. [15] built a framework to benchmark AutoML solutions and conducted a comparison of 4 systems across 39 datasets against a Random Forest classifier used as baseline; with enough time budged (4h), none of the AutoML solutions outperformed the Random Forest classifier. Guyon et al. [16] found a performance gap between human-tweaked and AutoML-generated model pipelines retrieved from various editions of the 'AutoML Challenge.' Finally, in the SE domain, Tanaka et al. [7] studied the effectiveness of AutoML in predicting the number of defects in software modules; using auto-sklearn, they achieved a prediction performance similar to those reported in prior studies.

C. XAI

AI/ML techniques are in general not easy to understand, even for experts. This has led to a growing interest in the topic of XAI [25], i.e., techniques focusing on *interpretability* (i.e., the understandability of the intuition behind a model's output) and *explainability* (i.e., the understandability of a model's internal logic and mechanics) [26]. Model explanations may apply to a single prediction (local) or to a model's entire behavior (global). Furthermore, interpretability can be *direct*, to indicate the use of self-explainable models whose internal logic is intrinsically transparent and therefore generally understandable (e.g., decision trees), or *post-hoc*, which involves the use of an auxiliary method to explain a complex, black-box model's behavior (e.g., feature importance scores, heatmaps, and natural language descriptions of its inner working and decision logic). Interpretability features may also apply to any
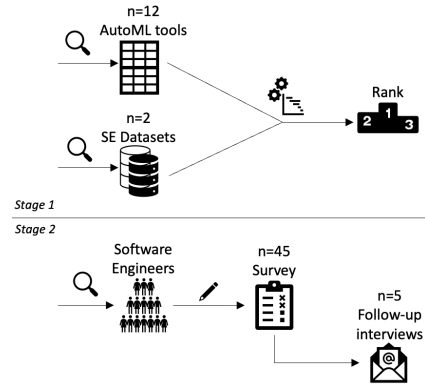

Fig. 2. The two experimental stages executed in the study.

type of algorithm (agnostic) or only to one family (model-specific). Finally, fairness indicates the presence of features to detect and deal with biases (e.g., ethnicity, gender, disability status) in models and training data.

III. STUDY DESIGN

In this work, we investigate the use of AutoML tools for SE. In particular, we ask the two following research questions:

- **RQ1** – *How do AutoML tools perform on SE tasks?*
- **RQ2** – *How do software engineers currently use AutoML? How does it fit their workflow? What are their perceptions?*

To answer the questions, we follow a mixed-methods approach as depicted in the two-stage research framework in Fig. 2. In *stage 1*, to answer RQ1 we perform a quantitative assessment of AutoML tools performance. Accordingly, we first select *12* AutoML solutions, both open-source (4) and commercial (8). The list of selected tools is reported in Table I (see Sect. III-A). Then, we choose a text-classification task – i.e., sentiment analysis from technical text – which has recently gained considerable attraction in the SE research community; contextually, we retrieve two datasets totaling over 10k documents written by developers (Sect. III-B). Finally, we apply the selected tools to the datasets and compare their performance using the same metrics adopted in previous benchmark studies that developed SE-specific tools for the tasks (Sect. III-C). In *stage 2*, to answer RQ2 we perform a qualitative analysis aimed to garner insights about the current extent of adoption of AutoML at the various stages of the AI/ML workflow and the potential different perceptions of software engineers about using AutoML solutions. Accordingly, we carried out an online survey with 45 participants; we also conducted follow-up interviews with 5 participants to deepen the relevant themes that emerged from the survey analysis (Sect. III-D).

A. AutoML Tools

Table I lists the main characteristics of the tools sampled for conducting the AutoML benchmark. The process of sampling the tools was conducted in two steps.

First, we selected commercial solutions. As a seed, we started by including the AutoML solutions provided by the companies identified as leaders in the *Gartner's Magic Quadrant For Data Science And ML Platforms*, namely Amazon,

Google, IBM, and Microsoft. Then, we performed a web search on Google using the keywords 'automl' and 'autoai' and selected the first four commercial products found, thus doubling the number of those already included (8 overall).

Second, to identify open-source solutions, we explored the GitHub topics pages for the keywords 'automl' and 'autoai;' overall, we identified 831 repositories. Because OSS solutions are generally libraries that require at least some coding, we opportunistically chose to retain only tools written in Python (395). Furthermore, given the popularity of the topic, we selected the repositories having 200+ stars (99), to ensure removing small or personal projects. Also, we removed: (i) 'stale' projects, retaining only the repositories updated in the last twelve months (79), deemed a sufficient amount of time, considering the hype around the topic; (ii) false positives, i.e., repositories not actually containing AutoML tools (38); (iii) repositories with README files missing or not in English (34); (iv) libraries supporting only Computer Vision tasks or a specific workflow activity, such as hyperparameter optimization (HPO), neural architectural search (NAS), and model compression (28); (v) smaller-scale solutions from vendors of larger commercial solutions already selected in the previous step (25); (vi) repositories containing resources for educational purposes or competitions (23). Finally, we manually inspected the existing documentation of the remaining tools and retained only those supporting both text features (NLP) and providing straightforward examples of how to automate a text analysis task (see the experimental datasets and task descriptions in Sect. III-B). At the end of this step, we selected four open-source AutoML tools (marked with a * in Table I).

Next, we briefly comment on the characteristics of the selected twelve tools, grouped by type.

Open-source solutions. We observe that open-source solutions can only be run on-premise and can only be used as APIs (i.e., requiring code development) or command-line applications (AutoGoal and Ludwig). All solutions support image, tabular, and textual data types. With respect to the stages and activities highlighted in the typical ML workflow in Fig. 1, all these solutions fully support the analysis stage as the core activities of ML-enabled systems development. Regarding preparation, they support data cleaning but provide no support for ML-assisted data labeling. Also, with respect to the dissemination stage, none of the sampled solutions support model deployment and monitoring, with the sole exception of Ludwig, which allows model pipelines to be served by spawning a REST API with one command. Finally, none of the open-source solutions currently supports explainability.

Commercial solutions. The landscape of commercial AutoML solutions is different than the previous one. First, these solutions run on cloud infrastructure with the sole exception of RapidMiner Studio, which is also the only desktop solution; all the other platforms are web applications also offering programmatic access via APIs. The model analysis stage is also well supported by commercial solutions. Compared to OSS solutions, they better support the preparation stage, by also providing support for assisted data labeling, and

dissemination, as they all allow one-click model deployment and monitoring.

In addition, with the exception of RapidMiner Studio, all the benchmarked commercial AutoML platforms support XAI, albeit to a slightly different extent. We found that, in general, these solutions ensure fairness by providing features to spot model/data biases and implement both model-dependent and independent (agnostic) features. In terms of explanation scope, they offer both local and global explanations. Regarding the support to interpretability, self-explaining models are generally supported thus, with the exception of BigML, these tools focus on providing *post-hoc* explanation.

Wrapping up, we note that, as compared to open-source solutions, commercial AutoML platforms offer code-free and more end-to-end solutions – as they cover the dissemination stage, offering model deployment and monitoring automation – and also support XAI to various extents.

B. Datasets

We selected two text-classification datasets from the SE domain. The first dataset is retrieved from Novielli et al. [27] and contains 7,122 sentences from GitHub pull-request and commit comments manually annotated with sentiment polarity labels based on Shaver et al.'s framework [28] (28.3% positive, 29.3% negative, and 42.4% neutral). The second dataset is retrieved from Ortu et al. [29] and includes about 6,000 issue comments and sentences authored by software developers of popular open-source software projects, such as Apache and Spring. This Jira dataset is manually annotated with six discrete emotion labels also from the Shaver et al.'s framework [28] (i.e., love, joy, anger, fear, surprise, and sadness), whereas this study focuses on emotion polarity (i.e., the positive, negative, or neutral valence conveyed by texts). Therefore, positive emotions, i.e., 'love' and 'joy,' are translated into a positive polarity label. Similarly, 'sadness,' 'anger,' and 'fear' are mapped to the negative polarity class. Instead, the cases labeled as 'surprise' are discarded because this emotion label could be either considered positive or negative, depending on the expectations of the author of a text. Unlike the GitHub dataset, the Jira dataset is not well-balanced (31.3% positive and 68.7% negative).

C. Metrics

To evaluate the AutoML tools' performance, we rely on the same metrics of Precision (P), Recall (R), and F1-measure (F) used in previous works, thus enabling a performance comparison. We also report macro- and micro-averaged metrics to show overall classification performance scores across the classes. Macro-averaging averages the metrics equally over the classes, whereas micro-averaging calculates the scores over the data points in all classes, thus it tends to be influenced by the performance of the majority class [30].

Because tools can be configured to generate models that maximize the classification accuracy with respect to a specific metric, to enable fair comparison we choose to optimize with respect to F1-measure.

TABLE I

A BREAKDOWN OF THE CHARACTERISTICS OF AUTOML SOLUTIONS BENCHMARKED IN THIS STUDY (*=OPEN SOURCE, ?=FEATURE ABSENCE/PRESENCE INFERRED UPON CODE INSPECTION, NOT FROM DOCUMENTATION).

| Tool | Infra struct. | Solution | Data types | Preparation | | | Analysis | | Dissemination | | XAI |
				Data clean.	Data label.	Feature engin.	Model traninig	Model eval.	Model deploym.	Model monitor.	
AutoGluon*	On prem.	API	img, tab, txt	Y	N	Y	Y	Y	N	N	N
AutoGoal*	On prem.	API, CLI	img, tab, txt	N?	N	Y?	Y	Y	N	N	N
AutoKeras*	On prem.	API	img, tab, txt	Y?	N	Y?	Y	Y	N	N	N
Amazon SageMaker AutoPilot	Cloud	API, Web	img, tab, txt	Y	Y	Y	Y	Y	Y	Y	Y
BigML	Cloud	API, Web	img, tab, txt	Y	N	Y	Y	Y	Y	Y	Y
DataRobot AI Cloud	Cloud	API, Web	img, tab, txt	Y	Y	Y	Y	Y	Y	Y	Y
Google Vertex AI	Cloud	API, Web	img, tab, txt, vid	Y	Y	Y	Y	Y	Y	Y	Y
H20 Driverless AI	Cloud, On prem.	API, Web	img, tab, txt	Y	Y	Y	Y	Y	Y	Y	Y
IBM Watson AutoAI	Cloud	API, Web	img, tab, txt	Y	N	Y	Y	Y	Y	Y	Y
Ludwig AI AutoML*	On prem.	API, CLI	img, tab, txt	Y?	N	Y	Y	Y	Y	N	N
MS Azure AutoML	Cloud	API, Web	img, tab, txt	Y	Y	Y	Y	Y	Y	Y	Y
Rapid Miner Studio	On prem.	Desktop	img, tab, txt	Y	N	Y	Y	Y	Y	Y	N

Regarding the sentiment polarity dataset, the same metrics were used by Calefato et al. [31] to assess the performance of Senti4SD, a SE-specific classifier, for each of the three classes (positive, negative, and neutral). Regarding the emotion dataset, instead, Ortu et al. [29] did not provide an assessment of classification performance along with the Jira gold standard; however, a classification performance in terms of Precision, Recall, and F-measure is reported in the benchmark study by Novielli et al. [27].

For the sake of completion, we also compare our results to those reported in Zhang et al. [32], who show that fine-tuned pre-trained transformer-based models (e.g., BERT) outperform the existing SE-specific sentiment analysis tools.

Finally, to ensure a fair comparison among the tools, we manually changed their configuration, thus creating a shared experimental setting resembling that of prior work by: (i) using a common optimization metric during the model analysis stage—we opted for maximizing F1-score; (ii) setting the same train/test data split—specifically, 70% training, 20% validation, and 10% test. Because it was not always possible to set a time budget, we ignore the time performance in our benchmark as execution times are not comparable.

D. Survey & Interviews

Recruitment. To recruit survey participants, we sought ML and software engineers with real-world experience on AutoML. We recruited participants through public announcements posted on Twitter, relevant subreddits, and LinkedIn groups as well as through personal contact networks. No monetary compensation was offered.

Survey. To create and host the web survey we used Google Forms. Overall, the survey contained 43 questions structured into four parts (see the supplemental online material).[2]

[2]https://doi.org/0.6084/m9.figshare.22640248

The first two parts are intended to collect the participants' demographic. We asked respondents to report their overall working experience, experience with AI/ML-based systems development, their job role as well as typical team size, company characteristics, typical project type, and development workflow; participants having no experience whatsoever with AI/ML projects were redirected to a disqualification page. Furthermore, we screened participants based on their familiarity and experience with AutoML and collected information about their general attitudes toward using AutoML tools. The third part was conditionally filled out by respondents who reported having some familiarity and experience with AutoML; it collected feedback on how AutoML currently fits their development workflows, in general, and their specific task(s), in particular, as well as points of friction and challenges, if any. The fourth part was filled out by all survey respondents as it aimed to collect insights about how participants would want to use it, what workflow stages and steps they wish AutoML tools supported better, and any features missing. The survey ended with an open-ended question to collect any other feedback from the participants and their consent to be contacted for a follow-up interview. Finally, two of the authors engaged in a process of inductive coding to extract and iteratively synthesize through discussion the common themes that arose from both the analysis of the open-ended questions.

Follow-up interviews. We conducted follow-up interviews with survey respondents who gave consent; they agreed to further elaborate on their answers and reflect on the themes that emerged from the survey analysis over email or chat. After completing the interviews, two of the authors analyzed the written answers to identify and extract excerpts that were deemed to expand further the relevant themes identified from the analysis of the open-ended answers.

Tool	Dataset (algorithm)	Positive			Neutral			Negative			Macro-avg			Micro-avg
		P	R	F	P	R	F	P	R	F	P	R	F	
AutoGluon	GitHub (Weight. Ensem.)	0.73	0.73	0.73	0.78	0.85	0.81	0.74	0.63	0.68	0.75	0.74	0.74	0.75
	Jira (Weight. Ensem.)	0.77	0.85	0.81	0.90	0.90	0.90	0.76	0.67	0.71	0.81	0.81	0.81	0.86
AutoGoal	GitHub (Perceptron)	0.84	0.79	0.81	0.82	0.82	0.82	0.77	0.82	0.80	0.81	0.81	0.81	0.81
	Jira (LinearSVC)	0.54	0.71	0.61	0.84	0.79	0.66	0.72	0.61	0.66	0.70	0.70	0.70	0.75
AutoKeras	GitHub (Neural Net.)	0.91	0.95	0.93	0.93	0.88	0.90	0.89	0.91	0.90	0.91	0.91	0.91	0.91
	Jira (Neural Net.)	0.80	0.86	0.83	0.92	0.89	0.91	0.75	0.80	0.77	0.82	**0.85**	**0.83**	**0.87**
Amazon SageMaker Autopilot	GitHub (XGB)	0.74	0.63	0.68	0.70	0.84	0.77	0.73	0.63	0.67	0.73	0.70	0.71	0,72
	Jira (XGB)	0.79	0.80	0.79	0.88	0.92	0.90	0.88	0.66	0.75	**0.85**	0.80	0.82	0.86
BigML	GitHub (Decision Forest)	0.78	0.59	0.67	0.65	0.85	0.74	0.75	0.62	0.68	0.73	0.69	0.70	0.70
	Jira (Decision Forest)	0.77	0.83	0.80	0.90	0.91	0.90	0.88	0.68	0.77	**0.85**	0.81	0.82	**0.87**
DataRobot AI Cloud	GitHub (CNN)	0.95	0.90	0.92	0.90	0.95	0.93	0.91	0.88	0.90	**0.92**	0.91	**0.92**	**0.92**
	Jira (CNN)	0.78	0.79	0.78	0.90	0.92	0.91	0.82	0.70	0.76	0.83	0.80	0.82	**0.87**
Google Vertex AI	GitHub (?)	0.93	0.92	0.93	0.88	0.92	0.90	0.88	0.83	0.86	0.90	0.89	0.89	0.89
	Jira (?)	0.77	0.91	0.83	0.90	0.91	0.91	0.86	0.60	0.70	0.84	0.80	0.82	**0.87**
H2O Driverless AI	GitHub (XGB)	0.72	0.58	0.64	0.66	0.82	0.73	0.63	0.54	0.58	0.67	0.65	0.65	0.67
	Jira (LightGBM)	0.72	0.83	0.77	0.88	0.86	0.87	0.69	0.59	0.64	0.76	0.76	0.6	0.82
IBM Watson AutoAI	GitHub (XGB)	0.39	0.62	0.48	0.80	0.60	0.69	0.46	0.53	0.49	0.55	0.58	0.55	0.58
	Jira (XGB)	0.55	0.73	0.63	0.93	0.77	0.84	0.24	0.69	0.35	0.57	0.73	0.61	0.76
Ludwig AutoML	GitHub (Nueral Net.)	0.90	0.89	0.90	0.85	0.89	0.87	0.86	0.81	0.83	0.87	0.86	0.87	0.87
	Jira (Nueral Net.)	0.71	0.79	0.75	0.87	0.88	0.88	0.88	0.64	0.74	0.82	0.77	0.79	0.83
MS Azure AutoML	GitHub (XGB)	0.88	0.87	0.88	0.,87	0.88	0.88	0.84	0.84	0.84	0.87	0.86	0.86	0.87
	Jira (LightGBM)	0.83	0.75	0.78	0.88	0.92	0.90	0.77	0.71	0.74	0.83	0.80	0.81	0.86
Novielli et al. [27]	GitHub (Senti4SD)	0.95	0.91	0.93	0.90	0.93	0.92	0.92	0.90	0.91	**0.92**	**0.92**	**0.92**	**0.92**
	Jira (SentiCR)	0.79	0.88	0.83	0.81	0.91	0.63	0.80	0.89	0.72	0.83	0.78	0.80	0.86
Zhang et al. [32]	GitHub (RoBERTa)	0.93	0.96	0.94	0.91	0.92	0.92	0.93	0.89	0.91	**0.92**	**0.92**	**0.92**	**0.92**

IV. RESULTS

A. Quantitative Analysis

In this section, we report the performance of the twelve AutoML tools on the two sentiment analysis datasets described earlier. Results are reported in Table II. For each dataset, we highlight in bold the best performance in terms of the two main metrics (i.e., macro- and micro-averaged F1-scores).

GitHub dataset. The GitHub dataset is the more balanced of the two analyzed in this benchmark. Previous work by Novielli et al. [27] and Zhang et al. [32] report performance in terms of micro- and macro-average F1-score between 0.91 and 0.92, which we use as a reference for state-of-the-art performance on the dataset. In the case of Zhang et al. [32],

the best performance reported was obtained using the pre-trained transformer model RoBERTa; in the case of Novielli et al. [27], the best results reported refer to the Senti4SD tool. DataRobot AI Cloud obtained the best results among the AutoML solutions. It performed as good as the pre-trained transformers assessed in Zhang et al. [32] and obtained a slightly smaller macro-averaged Recall as compared to the Senti4SD tool benchmarked in Novielli et al. [27] (0.91 vs. 0.92). The second best-performing tool is AutoKeras, which obtained a score of 0.91 on all the metrics. The worst results were obtained by IBM Watson AutoAI, which obtained 0.58 micro-average and macro-averaged scores ranging between 0.55 and 0.58, and H20 Driverless AI, which obtained 0.67

micro-average and macro-averaged scores ranging between 0.65 and 0.67. Regarding the remaining tools, we observe a subset of solutions whose micro-average F1-scores range between 0.70 and 0.75 (i.e., AutoGluon, Amazon SageMaker Autopilot, and BigML) and another subset consisting of tools with a performance between 0.81 and 0.89 (i.e., AutoGoal, Ludwig AutoML, MS Azure AutoML, and Google Vertex AI).

Jira dataset. The Jira dataset is less balanced than the GitHub one, which might explain the worse performance in general. Novielli et al. [27] report a micro-average score of .86 and macro-averaged scores between 0.78 and 0.83. AutoKeras, BigML, and Google Vertex AI outperform by a small margin the SE-specific tools as they reach a micro-average score of 0.87, followed by MS Azure AutoML and Amazon SageMaker Autopilot (0.86). The performances of the AutoML tools in terms of macro-averaged Precision, Recall, and F1, although varied, are equal to or better than (values between 0.80 and 0.85) the best performance reported in Novielli et al. [27] (between 0.80 and 0.83). For this dataset, the worst performances are those achieved by IBM Watson AutoAI (macro-averaged scores between 0.57 and 0.73, micro-average 0.76) and AutoGoal (~0.70 macro-average scores, 0.75 micro-average).

Overall, we found that AutoKeras (open source) and DataRobot AI Cloud (commercial) perform best on both datasets; both AutoML tools obtained the best scores with neural network models, reaching a performance that is on par with SE-specific research tools for the GitHub dataset and slightly better for the Jira dataset. Other solutions such as Google Vertex AI and MS Azure AutoML (commercial) and Ludwig AutoML (open source) perform well on average on both datasets; these solutions built either a neural network or a light/extreme gradient boosting machine model. IBM Watson AutoAI is the AutoML tool that performed consistently worse on both datasets.

B. Qualitative Analysis

1) Survey: We received 51 answers. However, 6 came from respondents who reported having heard of AutoML but never used it; hence, we excluded them and the analysis reported next focuses on the remaining $n = 45$ subjects.

Participants' background. Here we check whether sampled participants are representative of the target population (i.e., software engineers working on AI/ML projects and having experience with AutoML). The subjects identify themselves mostly as male (39, 87%) and have ages in the range of 26-35 (25, 56%) and 36-45 (14, 31%). They come from different countries, in particular Italy (9, 20%), Brazil (8, 18%), the United States (7, 16%), and Canada (6, 13%).

The respondents reported having mostly a background in CS, namely a PhD (16, 36%), MSc (10, 22%), or BSc (6, 13%); the remaining respondents have an MSc in AI/Computational Intelligence (7, 16%), BSc in Statistics (4, 9%), and PhD in Physics (2, 4%). About half of them have an overall working experience between 5-10 years (17, 26%) or more than 10 years (14, 22%). On a five-point Likert scale

(1=*Not experienced at all*, 5=*Very experienced*) in SE practices (e.g., designing, implementing, testing, and maintaining complex software applications), most of them reported a high (18, 40%) or very high (15, 33%) level of expertise. Regarding AI/ML specifically, the respondents mostly reported having some experience (i.e., 3-4 years, 20, 45%) or considerable experience (i.e., 5-10 years, 15, 33%), with the remaining ones (10, 22%) having limited experience (1-2 years). Most of the respondents identify their work activities (answers not mutually exclusive) as pertaining to the positions of ML engineer (34), followed by MLOps (15), Researcher (7), and Data Scientist (7) in either a tech company (67%) or non-commercial/academic research lab (24%).

The sizes of the companies for which respondents work are varied; about half of the respondents are working for either very large companies (10,000+ employees, 27%) or very small ones (10-50 employees, 25%). Most of the subjects (17, 38%) work in teams of 6 to 9 members; very small (2-3 members) and very large (16+) teams are also common (27% and 22% of the responses, respectively). This shows that AutoML is a technology appealing to all types of companies, regardless of their size. Half of the respondents report that ~35-60% of their team members (median=3) have a strong SE background. Most of the teams they work in are either well-balanced, interdisciplinary teams (23, 51%) or teams made of mostly AI/ML people (17, 38%); less common are the cases where most team members are software engineers (5, 11%).

Respondents mostly work on projects in the domain of healthcare (19), banking & insurance (16), and telecommunication (15), where AI/ML models are either part of a product (10, 22%) or integrated into a larger, production-ready software system (25, 56%). The primary use cases for AI/ML projects are classification (15), prediction (13), forecasting (12), risk evaluation (11), and anomaly detection (10). The typical lengths of projects are 4-6 months (14, 31%) and 7-12 months (13, 29%); less common are projects longer than one year (11, 24%) or taking only 2-3 months (7, 16%). With respect to the AI/ML reference workflow in Fig. 1, respondents report participating mostly in the analysis stage, i.e., feature engineering (39), model evaluation (34), and training (30). Dissemination activities are also common, i.e., model deployment (22) and monitoring (13), whereas the involvement in preparation activities, i.e., data cleaning (6), labeling (7), and collection (8), is far less common. Only one respondent reported being involved in the whole end-to-end process.

Usage and features of AutoML. Overall, the survey reveals that among software engineers there is considerable interest in AutoML solutions, whether applied on a regular basis (15, 33%) or on select projects (23, 51%), with the remaining respondents (7, 16%) reporting having used it at least in internal projects. They report using or having experimented with both proprietary cloud solutions (in order of popularity MS Azure AutoML, Amazon SageMaker AutoPilot, and IBM Watson AutoAI) and open-source tools (AutoKeras, TPOT, auto-sklearn, nni, and Alteryx); the cited reasons are dif-
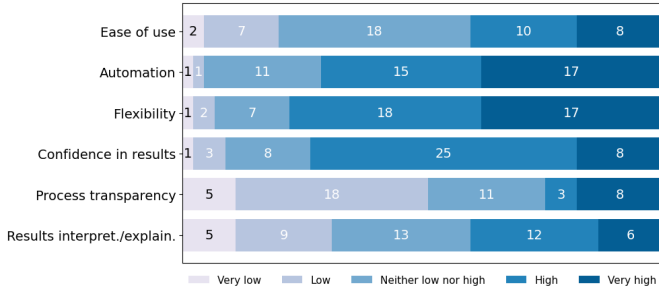
Ease of use | 2 | 7 | 18 | 10 | 8
Automation | 1 1 | 11 | 15 | 17
Flexibility | 1 2 | 7 | 18 | 17
Confidence in results | 1 3 | 8 | 25 | 8
Process transparency | 5 | 18 | 11 | 3 | 8
Results interpret./explain. | 5 | 9 | 13 | 12 | 6

Very low | Low | Neither low nor high | High | Very high

Fig. 3. Features more relevant for choosing an AutoML solution.

ferent: for proprietary tools, the reasons are vendor lock-in and clients' trust in renowned companies; for open-source solutions, it is the flexibility in adapting to custom workflows.

Table I shows that the features in the analysis stage (feature engineering plus model training and evaluation) are the most widely supported among the reviewed tools. Therefore, it is not surprising that the questionnaire respondents report mostly using AutoML for model training (45, 100%), model evaluation (33, 73%), and feature engineering (20, 44%). Notably, the use of AutoML for model deployment and monitoring (17, 20%) is considerably more common than data preparation activities (5, 10%). None of the participants reported using AutoML in *full automation* mode, with humans out of the loop. Most respondents (22, 49%) use AutoML in *human-directed automation* mode, whereby they can select methods and parameters before execution; the remaining ones use it in *system-suggested automation* mode (12, 27%, i.e., the system decides on parameters/methods, and human approval is requested before execution) and *system-directed automation* (9, 20%, i.e., the system executes automatically, and humans are in the loop and can intervene anytime). However, when choosing an AutoML solution (Fig. 3), respondents value the flexibility to customize and adapt the workflow to their needs (78% either *very high* or *high*) and automation capabilities (71% either *very high* or *high*) more than ease of use (40%). They report having a substantially high level of confidence in the results obtained using AutoML tools (73% either *very high* or *high*). In addition, few respondents find the process to be transparent (51% either *very low* or *low*) and they are also somewhat conflicted when it comes to the interpretability/explainability of results obtained via AutoML.

2) Emerging Themes: Five of the eight respondents who had given their consent agreed to a follow-up interview. All of them opted for conducting it over email. As such, we defined and emailed four main questions; then, they answered by email and we replied back to request further clarification when needed. Here we report the analysis of the answers to both the open-ended questions from the survey and the follow-up interviews (see the supplemental material). We followed a sequential explanatory strategy whereby we first performed an open coding of the responses to the open-ended survey questions to identify common themes; then, we formulated *ad hoc* questions for the interviewees to gather further insights. We were able to identify three emerging themes, which we illustrate next by providing relevant excerpts. We use the code P_x to report quotes from survey participants and PI_y to

distinguish those who also agreed to the follow-up interview (Table III provides an overview).

AI/ML team members wear different hats. Working on AI/ML projects is a team sport involving different activities and skills. Still, we found that in these teams, member roles tend to be blurred; especially in smaller teams, it is not unusual for surveyed participants to perform activities belonging to different stages of the AI/ML workflow, in particular, analysis and dissemination; (only a few reported participating in data preparation activities). This has a considerable impact on expectations and perceived usefulness of AutoML.

Albeit rarely involved in it, participants are skeptical that AutoML might ever automate data preparation activities (P_2, P_{11}, P_{21}, P_{33}):

"*I'm not involved, not often at least, but data people in my teams tell me that [AutoML] support is garbage. I don't think it's even possible to automate it. I wish, but it's too domain dependent...*" (PI_{12})

The analysis stage is the one that AutoML appears to support better and with which most of the respondents reported to be satisfied, although some (P_{14}, P_{31}) feel that "*feature engineering might be improved*" (P_8). The dissemination stage is the one that has more room for improvement. Our analysis suggests that this deficiency might be because ML models do not always end up being deployed into production systems ("*sometimes our customers just want us to uncover insights about features and relationships between them,*" P_{26}). Still, several respondents (P_4, PI_{12}, P_{16}, PI_{23}) were vocal about wishing to see better automation support for model deployment and monitoring:

"*Cloud solutions force you to use their infrastructure. [They] don't help us deploy where we want. They are not flexible. As an ML engineer, I want to choose where to deploy and how to serve models. We have our own infrastructure and monitoring dashboard that we'd like to keep using.*" (PI_{23})

"*[AutoML] is just a commercial name for a glorified version of HPO and CASH tools. Automation right now happens only at the analysis stage [...] The other activities [are] not first-class citizens.*" (P_{16})

ML-as-a-Service vs. in-house solutions. Another theme that emerged is that cloud solutions that offer ML as a service might be more appealing to smaller teams, where roles tend to be blurred:

"*These MLaaS commercial tools force you to embrace their solution [end-to-end]. Management and customers trust them because they are sold by well-known vendors. But having worked in teams of different sizes on different projects, I feel that those are more useful in scenarios where*

they ask you to build a fast PoC [...] to show the customer how good the model can be and then give the data scientists a baseline to work on if the project is greenlit." (PI_{40})

This lack of flexibility is not appreciated by respondents working in larger teams with more clearly defined roles. They would want to see AutoML not as one tool, but either as a collection of ("*smaller scope tools, something to plug into your own workflow*", P_{17}) or as a 'generator' of custom, project-specific workflows rather than an 'automator':

"*We tinkered with many AutoML alternatives before deciding to build one around OSS libraries and tools we already used and liked. It's actually more of an AutoML workflow generator.*" (PI_{23})

Tension between automation desire and user agency. The analysis revealed the existence of a contrast among respondents between the need for human control and a higher desire for automation. In particular, our findings suggest that the desire for higher automation varies with the stages of the ML workflow in which participants are involved and also clashes with the difficulty of actually achieving it. As reported earlier, data preparation is especially domain-dependent, and albeit those involved in data preparation do wish for more automation, automating domain knowledge elicitation remains a prohibitive task. The analysis stage is the one where the highest level of automation is achieved (P_9, P_{16}, P_{25}, P_{28}); yet, respondents are also aware of the need for rather using AutoML in "*cruise control mode*" (P_{25}), where humans "*stay in the loop to oversee and steer the system, and possibly get explanations if need be*" (PI_{29}), due to regulatory and organizational requirements. Given the large presence of ML engineers and MLOps among respondents, it is not surprising that we found participants to be eager to achieve better support for automating the dissemination stage (PI_{12}, P_{16}, PI_{23}, PI_{44}), where human oversight is perceived both less problematic and intrinsic to monitoring:

"*Full automation is problematic, with ethical concerns and regulations [...]. So you want control when you need to understand data, understand models. But there are no such issues with what you call dissemination. Better (if not fully) automated deployment would be amazing... there user control comes just with monitoring.*" (PI_{44})

"*Dissemination [is poorly supported] because most of the models rarely get into production. However, for us ML engineers and MLOps there is a lot of code generation involved to integrate them into the final system and deploy the whole thing. A lot of repetition is there that could be automated with AutoML.*" (PI_{12})

V. DISCUSSION

In this study, we investigated two research questions, namely RQ1 (AutoML tools benchmark) and RQ2 (software engineers' perspective on using AutoML). RQ1 is answered in Section V-A, whereas the findings related to RQ2 are discussed in Sections V-B and V-C.

A. AutoML as a Solution to Augment SE Research

Our benchmark revealed that, on average, AutoKeras (open source) and DataRobot AI Cloud (commercial) are the best performing AutoML solutions for sentiment analysis on the two datasets analyzed. These tools perform on par with both the pre-trained transformer-based models and SE-specific research tools in the case of the GitHub dataset, and slightly better in the case of the Jira dataset.

Although there is no gold standard or concrete threshold of various evaluation metrics to decide whether a SE-specific sentiment analysis tool can be put into real use, our experiment results show that the use of AutoML is practical for sentiment analysis tasks in SE. It is worth noticing that the performance of the best AutoML tool is on par with state-of-the-art results and SE-specific research tools built by human experts, who oversaw all the steps within the development pipeline. On the contrary, Zöller and Huber [17] compared the performance of a set of five AutoML tools (different than those benchmarked in this study) against a couple of Kaggle notebooks containing the best-performing pipelines developed by human experts on a couple of public datasets; they found the models built by human experts to outperform the AutoML tools, which in general achieved mediocre performance in comparison. As such, although the mileage may vary with the specific tool and tasks other than sentiment analysis, we suggest researchers consider the use of AutoML solutions to train initial models as baselines or starting points for future work. Finally, we note that our assessment here pertains only to the analysis stage. We ignored the preparation and dissemination stages, which we discuss next.

Implications. Echoing the words of Wang et al. [10] and Crisan and Fiore-Gartland [33], our findings encourage SE researchers to use AutoML for *augmenting* their research activities; in particular, by automating model building and evaluation, they can spend their (limited) resources on time-consuming activities such data collection and cleaning, which still heavily depend on the elicitation of domain knowledge and human expertise.

B. Uneven Automation Through the AI/ML Workflow

The progress in AI/ML research in the last few years has been enormous and we now are witnessing a fundamental shift in software engineering where machine learning becomes the new software, powered by big data and computing infrastructure. Much like cloud resources have made it easy to access high-performing hardware, resources like the Hugging Face model registry, Large Language Models (LLMs), and AutoML are commoditizing ML models, which become either readily available to developers or extraordinarily easier to build and fine-tune. As a result, a project's success now depends on the (quality of) data rather than models.

Despite AI and ML being more and more data-centric, we found out that AutoML solutions currently provide little or no support to the data preparation stage. As P_2 put it succinctly, "*we just don't use AutoML for data preparation.*" While participants remain generally skeptical that data preparation can

be fully automated due to its heavy domain dependence and despite not usually being involved in such activities, they were also very vocal about the benefits that better tool support would bring to the AI/ML teams. Similarly, participants reported among their desiderata better support for dissemination – deployment in particular, since monitoring is seen as the means to enforce user oversight. Even if not completely automated, our respondents wish for an assisted and flexible solution that saves them from the tedium of writing boilerplate integration code and allows them to customize the deployment destination.

Our finding of uneven automation support of workflow stages adds a potential explanation for the low adoption of AutoML practices reported in previous surveys also conducted in the context of software engineering for ML [34, 9].

Implications. The automation of the analysis stage appears to have been tackled albeit still needing some improvements (e.g., explainability, feature engineering). To distinguish themselves from competitors, AutoML tool designers must focus on providing better assistance to data preparation and dissemination stages.

C. AutoML is a Misnomer (for non-Data Scientists)

In their studies on the use of AutoML among data scientists, Wang et al. [10] and Crisan and Fiore-Gartland [33] reported two common usage scenarios: (i) rapid model prototyping and (ii) data science democratization. Our qualitative analysis revealed that while the first scenario is also common among our participants, the second one is not, arguably because of the different target population: being versed in ML/software engineering practices and with coding abilities, our study participants do not see value in using AutoML as a means to lower the entry barriers for data science.

Consistently with previous research that already reported on the lack of customizability in existing AutoML solutions [35, 36], our findings add further evidence that experienced ML engineers/MLOps are more interested in flexibility and control than ease of use. As such, the desire for full automation appears to be inversely proportional to AI/ML expertise, and, given that ML engineers/MLOps rather look at these solutions as generators of custom AI/ML workflows, we argue that AutoML is currently a misnomer since the promise of automation is only fulfilled for data scientists at the model analysis stage.

Implications. Echoing the words of Xin et al. [35], to appeal to a broader audience, AutoML tool builders should refrain from using commercial names suggesting human-out-of-the-loop, full-automation capabilities (e.g., autopilot, driverless) given that full-automation has proven infeasible so far and only appealing to non-experts who want a one-click model building solution, while also being in contrast with the recent EU ethics regulations, which require AI/ML systems to empower humans through human-in-the-loop approaches.

Because AI/ML workflow activities are many and complex, AutoML users are also diverse, with different backgrounds and contrasting desires for control and automation. We argue that AutoML product designers should consider using per-

sonas [37], a well-known product design technique that allows to model archetypal users and better capture their goals, needs, and frustrations. In the case of AutoML, it is easy to envision at least three types of users, one for each of the three stages in the AI/ML workflow because, even though the roles may be blurred in teams, especially small, the user needs and focus at each stage are clearly defined and different.

D. Limitations

This study has a few limitations that can inform future research. Regarding the benchmark, some of the existing OSS AutoML solutions were excluded because, albeit supporting text features, their documentation was not providing clear examples similar to the chosen experimental task. While we acknowledge this as a potential limitation of the tool sampling process, we also point out that the focus here was not the completeness of the benchmark. In fact, our results already show that on-premise OSS solutions like AutoKeras can keep up with commercial cloud solutions and automatically generate models with similar if not better performance. Regarding the choice of sentiment analysis as the experimental task, we acknowledge a partial answer to RQ1. There is indeed a variety of other SE-specific classification tasks (e.g., defect prediction, bug classification). However, we initially opted for a task whose execution would require expertise that is typically outside of the knowledge domain of software engineers (i.e., NLP), thus showing how AutoML can be a Swiss knife for building ML models even on limited skills. Regarding the survey, albeit we found that participants come from a variety of backgrounds and use AutoML to different extents, the voluntary participation may have resulted in self-selection bias, with those responding being more active with or positive towards AutoML than those who did not.

VI. CONCLUSIONS

In this study, we investigated the use of AutoML for data-driven software engineering. Our findings suggest that AutoML promises to generate (classification) models that outperform those trained and optimized by researchers in the SE domain, making it a valuable aid for automating the heavy lifting in model analysis. However, we also found that the currently available AutoML solutions do not equally support automation across all stages of the ML development workflow, with data/ML engineers and MLOps currently being second-class citizens. Overall, our study contributed insights for the SE research community and tool builders on how to use AutoML technologies and improve the design of its next generation.

ACKNOWLEDGMENT

REFERENCES

[1] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[2] A. Luckow, K. Kennedy, M. Ziolkowski, E. Djerekarov, M. Cook, E. Duffy, M. Schleiss, B. Vorster, E. Weill, A. Kulshrestha *et al.*, "Artificial intelligence and deep learning applications for automotive manufacturing," in *2018 IEEE Int'l Conf. on Big Data (Big Data)*. IEEE, 2018, pp. 3144–3152.

[3] W. Markow, S. Braganza, B. Taska, S. M. Miller, and D. Hughes, "The quant crunch: How the demand for data science skills is disrupting the job market." [Online]. Available: https://www.ibm.com/downloads/cas/3RL3VXGA

[4] B. Chakravorti, A. Bhalla, R. Shankar Chaturvedi, and C. Filipovic, "50 global hubs for top ai talent." [Online]. Available: https://hbr.org/2021/12/50-global-hubs-for-top-ai-talent

[5] D. Zhang, S. Mishra, E. Brynjolfsson, J. Etchemendy, D. Ganguli, B. Grosz, T. Lyons, J. Manyika, J. C. Niebles, M. Sellitto *et al.*, "The ai index 2021 annual report," *arXiv preprint arXiv:2103.06312*, 2021.

[6] F. Hutter, L. Kotthoff, and J. Vanschoren, *Automated machine learning: methods, systems, challenges*. Springer Nature, 2019.

[7] K. Tanaka, A. Monden, and Z. Yücel, "Prediction of software defects using automated machine learning," in *2019 20th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, 2019, pp. 490–494.

[8] J. Vanschoren, J. N. Van Rijn, B. Bischl, and L. Torgo, "Openml: networked science in machine learning," *ACM SIGKDD Explorations Newsletter*, vol. 15, no. 2, pp. 49–60, 2014.

[9] K. Van der Blom, A. Serban, H. Hoos, and J. Visser, "AutoML Adoption in ML Software," *8th ICML Workshop on Automated Machine Learning*, 2021.

[10] D. Wang, J. D. Weisz, M. Muller, P. Ram, W. Geyer, C. Dugan, Y. Tausczik, H. Samulowitz, and A. Gray, "Human-AI Collaboration in Data Science: Exploring Data Scientists' Perceptions of Automated AI," *Proc. of ACM on Human-Computer Interaction*, vol. 3, no. CSCW, pp. 1–24, nov 2019.

[11] D. Sato, A. Wider, and C. Windheuser, "Continuous delivery for machine learning," Sep 2019. [Online]. Available: https://martinfowler.com/articles/cd4ml.html

[12] T. Menzies, "The five laws of se for ai," *IEEE Software*, vol. 37, no. 1, pp. 81–85, 2019.

[13] C. Wang, Z. Chen, and M. Zhou, "AutoML from Software Engineering Perspective: Landscapes and Challenges," in *2023 Proc. of 20th Int.l Conf. on Mining Software Repositories (MSR)*, may 2023.

[14] F. Majidi, M. Openja, F. Khomh, and H. Li, "An empirical study on the usage of automated machine learning tools," in *IEEE International Conference on Software Maintenance and Evolution, ICSME 2022, Limassol, Cyprus, October 3-7, 2022*. IEEE, 2022, pp. 59–70. [Online]. Available: https://doi.org/10.1109/ICSME55016.2022.00014

[15] P. Gijsbers, E. LeDell, J. Thomas, S. Poirier, B. Bischl, and J. Vanschoren, "An Open Source AutoML Benchmark," in *6th ICML Workshop on Automated Machine Learning*, jul 2019, pp. 1–8.

[16] I. Guyon, L. Sun-Hosoya, M. Boullé, H. J. Escalante, S. Escalera, Z. Liu, D. Jajetic, B. Ray, M. Saeed, M. Sebag, A. Statnikov, W.-W. Tu, and E. Viegas, "Analysis of the AutoML Challenge Series 2015–2018," in *Automated Machine Learning*, V. J. Hutter F., Kotthoff L., Ed. Springer, 2019, ch. 10, pp. 177–219.

[17] M.-A. Zöller and M. F. Huber, "Benchmark and Survey of Automated Machine Learning Frameworks," *Journal of Artificial Intelligence Research*, vol. 70, pp. 409–472, jan 2021.

[18] G. M. Paldino, J. De Stefani, F. De Caro, and G. Bontempi, "Does AutoML Outperform Naive Forecasting?" *Engineering Proceedings*, vol. 5, no. 1, p. 36, jul 2021.

[19] 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.

[20] W. Epperson, A. Y. Wang, R. DeLine, and S. M. Drucker, "Strategies for reuse and sharing among data scientists in software teams," 2022.

[21] S. Amershi, A. Begel, C. Bird, R. DeLine, H. Gall, E. Kamar, N. Nagappan, B. Nushi, and T. Zimmermann, "Software engineering for machine learning: A case study," in *2019 IEEE/ACM 41st Int'l Conf. on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*. IEEE, 2019, pp. 291–300.

[22] M. Hanussek, M. Blohm, and M. Kintz, "Can AutoML outperform humans? An evaluation on popular OpenML datasets using AutoML Benchmark," in *2020 2nd Int'l Conf. on Artificial Intelligence, Robotics and Control*. New York, NY, USA: ACM, dec 2020, pp. 29–32.

[23] L. Ferreira, A. Pilastri, C. M. Martins, P. M. Pires, and P. Cortez, "A Comparison of AutoML Tools for Machine Learning, Deep Learning and XGBoost," in *2021 Int'l Joint Conf. on Neural Networks (IJCNN)*, vol. 2021-July, no. October. IEEE, jul 2021, pp. 1–8.

[24] A. Balaji and A. Allen, "Benchmarking Automatic Machine Learning Frameworks," aug 2018.

[25] D. Gunning, M. Stefik, J. Choi, T. Miller, S. Stumpf, and G.-Z. Yang, "Xai—explainable artificial intelligence," *Science Robotics*, vol. 4, no. 37, p. eaay7120, 2019.

[26] P. Linardatos, V. Papastefanopoulos, and S. Kotsiantis, "Explainable ai: A review of machine learning interpretability methods," *Entropy*, vol. 23, no. 1, pp. 1–45, 2021.

[27] N. Novielli, F. Calefato, D. Dongiovanni, D. Girardi, and

F. Lanubile, "Can we use se-specific sentiment analysis tools in a cross-platform setting?" in *Proc. of 17th Int'l Conf. on Mining Software Repositories*. New York, NY, USA: ACM, 2020, p. 158–168.

[28] P. Shaver, J. Schwartz, D. Kirson, and C. O'connor, "Emotion knowledge: further exploration of a prototype approach." *Journal of personality and social psychology*, vol. 52, no. 6, p. 1061, 1987.

[29] M. Ortu, A. Murgia, G. Destefanis, P. Tourani, R. Tonelli, M. Marchesi, and B. Adams, "The emotional side of software developers in jira," in *2016 IEEE/ACM 13th Working Conf. on Mining Software Repositories (MSR)*. IEEE, 2016, pp. 480–483.

[30] F. Sebastiani, "Machine learning in automated text categorization," *ACM Comput. Surv.*, vol. 34, no. 1, p. 1–47, mar 2002.

[31] F. Calefato, F. Lanubile, F. Maiorano, and N. Novielli, "Sentiment polarity detection for software development," *Empirical Software Engineering*, vol. 23, no. 3, pp. 1352–1382, 2018.

[32] T. Zhang, B. Xu, F. Thung, S. A. Haryono, D. Lo, and L. Jiang, "Sentiment analysis for software engineering: How far can pre-trained transformer models go?" in *2020 IEEE Int'l Conf. on Software Maintenance and Evolution (ICSME)*. IEEE, 2020, pp. 70–80.

[33] A. Crisan and B. Fiore-Gartland, "Fits and starts: Enterprise use of automl and the role of humans in the loop," in *CHI '21: CHI Conf. on Human Factors in Computing Systems, Virtual Event / Yokohama, Japan, May 8-13, 2021*, Y. Kitamura, A. Quigley, K. Isbister, T. Igarashi, P. Bjørn, and S. M. Drucker, Eds. ACM, 2021, pp. 601:1–601:15. [Online]. Available: https://doi.org/10.1145/3411764.3445775

[34] A. Serban, K. van der Blom, H. Hoos, and J. Visser. (2020) The 2020 state of engineering practices for machine learning. [Online]. Available: https://se-ml.github.io/automl-report2020

[35] D. Xin, E. Y. Wu, D. J.-L. Lee, N. Salehi, and A. Parameswaran, "Whither AutoML? Understanding the Role of Automation in Machine Learning Workflows," in *Proc. of 2021 CHI Conf. on Human Factors in Computing Systems*. ACM, may 2021, pp. 1–16.

[36] Y. Sun, Q. Song, X. Gui, F. Ma, and T. Wang, "AutoML in The Wild: Obstacles, Workarounds, and Expectations," in *Proc of 2023 CHI Conf. on Human Factors in Computing Systems (CHI '23), April 23â•fi28, 2023, Hamburg, Germany*, vol. 1, no. 1. Association for Computing Machinery, feb 2023.

[37] J. Pruitt and J. Grudin, "Personas: practice and theory," in *Proceedings of the 2003 conference on Designing for user experiences*, 2003, pp. 1–15.