Knowledge and Information Systems (2023) 65:5097–5149
https://doi.org/10.1007/s10115-023-01935-1

REVIEW

Eight years of AutoML: categorisation, review and trends

Rafael Barbudo[1,2] · Sebastián Ventura[1,2] · José Raúl Romero[1,2]

Received: 16 September 2022 / Revised: 25 April 2023 / Accepted: 10 July 2023 /
Published online: 8 August 2023

Abstract
Knowledge extraction through machine learning techniques has been successfully applied in a large number of application domains. However, apart from the required technical knowledge and background in the application domain, it usually involves a number of time-consuming and repetitive steps. Automated machine learning (AutoML) emerged in 2014 as an attempt to mitigate these issues, making machine learning methods more practicable to both data scientists and domain experts. AutoML is a broad area encompassing a wide range of approaches aimed at addressing a diversity of tasks over the different phases of the knowledge discovery process being automated with specific techniques. To provide a big picture of the whole area, we have conducted a systematic literature review based on a proposed taxonomy that permits categorising 447 primary studies selected from a search of 31,048 papers. This review performs an extensive and rigorous analysis of the AutoML field, scrutinising how the primary studies have addressed the dimensions of the taxonomy, and identifying any gaps that remain unexplored as well as potential future trends. The analysis of these studies has yielded some intriguing findings. For instance, we have observed a significant growth in the number of publications since 2018. Additionally, it is noteworthy that the algorithm selection problem has gradually been superseded by the challenge of workflow composition, which automates more than one phase of the knowledge discovery process simultaneously. Of all the tasks in AutoML, the growth of neural architecture search is particularly noticeable.

Keywords AutoML · Algorithm selection · Hyper-parameter optimisation · Neural architecture search · Workflow composition · Systematic literature review

✉ José Raúl Romero
jrromero@uco.es

Rafael Barbudo
rbarbudo@uco.es

Sebastián Ventura
sventura@uco.es

1 Department of Computer Science and Numerical Analysis, University of Córdoba, 14071 Córdoba, Spain

2 Andalusian Research Institute in Data Science and Computational Intelligence (DaSCI), Córdoba, Spain

1 Introduction

Despite the huge amount of data available in corporations, there are still a large number of them, specially small- and medium-sized organisations, that do not handle and take advantage of such potential information. Definitely, its analysis could bring valuable insights into their working activity [9]. Extracting useful and novel knowledge for decision-making from raw data is a complex process that involves different phases [12] and requires strong background in multiple fields, such as pattern mining, databases, statistics, data visualisation or machine learning. In general, data scientists are responsible for identifying relevant data sources, automating the collection processes, preprocessing and analysing the acquired data, and communicating their findings to the stakeholders, so that they can propose solutions and strategies to business challenges [27]. The role of these data professionals is crucial in reaching quality decisions, requiring from their experience, intuition and the application of their know-how.

Along the knowledge discovery process, different phases are accomplished, some of which are human-dependent like interpreting the mined patterns. In contrast, others are made up of tasks that are repetitive and time-consuming. Also, it comes into play that the area is progressing fast and the algorithms and methods used for cleaning data or learning from them are becoming more and more sophisticated, and also more and more numerous [21]. In fact, some methods may be better suited than others to the data characteristics and formulation of a particular organisation [24]. This means that automated methods can make it more practicable for the data professional to explore the alternatives that best fit the problem at hand, as well as alleviating those tasks that are more time-consuming. An illustrative example of this situation is model building, where the large number of available machine learning (ML) algorithms makes it difficult to take the right decisions for the construction of the most appropriate models [1, 128, 344]. Here, apart from selecting the right algorithms, their performance could heavily rely on the selected values for their hyper-parameters [4, 229, 364]. In this context, the term automated machine learning (AutoML) [20] was first coined in 2014 to encompass those approaches that are committed to precisely automate these challenging tasks, allowing data scientists to cover a wider spectrum of alternatives, as well as shifting the focus to those phases requiring from their know-how and intuition and, ultimately, bringing the knowledge discovery process closer to domain experts [77, 129]. Since then, AutoML is a rapidly expanding field. Indeed, some AutoML proposals have already shown that they can outperform humans for certain tasks, such as the design of neural network architectures [8, 483].

AutoML approaches can be analysed from multiple perspectives. Different concrete operations or tasks like algorithm selection or hyper-parameter optimisation can be performed to automate the knowledge discovery process. Also, a wide range of techniques can be applied to complete a task. It is common in this field to find studies not explicitly categorised as AutoML, even though they are specifically oriented towards addressing AutoML problems like the design of neural network architectures, the selection of the best preprocessing or ML algorithms [1, 6] and their hyper-parameter values [33, 105], as well as the automatic composition of workflows for knowledge discovery [129, 285, 307]. This might blur the scope, applicability and categorisation of these methods. Because of the growth of AutoML in recent years, other works have explored this field across specific domains of knowledge. Nevertheless, we contend that a comprehensive global review, accompanied by a taxonomy of the field, would provide a holistic overview and situational analysis of the ongoing efforts. This would help identify emerging trends, assess the extent of coverage of different phases of the process, and highlight the dominant techniques being employed in each case. Therefore, this literature review focuses on the breadth of the knowledge area, prior to the requisite deepening of specific categories of tasks, phases or techniques.

More specifically, Luo [26] conducted a review of the various approaches that deal with the selection of the most appropriate ML algorithms and the optimisation of their hyper-parameters. The study mainly focused on the supervised setting and the biomedical environment. With respect to the hyper-parameter optimisation, Hutter et al. [19] examined approaches that use Bayesian optimisation and provided a concise overview of traditional techniques. Furthermore, Elsken et al. [10] reviewed the design of neural network architectures. In terms of automatic workflow optimisation, Serban et al. [31] evaluated the Intelligent Discovery Assistants, which were described as systems that aid users in analysing their data. This comprehensive definition encompasses a vast number of diverse proposals, some of which are related to AutoML. Specifically, the authors assessed a limited number of approaches using AI planning to generate knowledge discovery workflows automatically. Recently, He et al. [16] published a survey covering the automation of various phases of the knowledge discovery process, being specially oriented to the design of neural network architectures. Also, Hutter et al. [20] presented a volume mainly focused on particular tasks and AutoML systems, such as Auto-sklearn, Auto-Net or Automatic Statistician.

To the best of our knowledge, this paper presents the first systematic literature review that provides a comprehensive overview of the research field of AutoML, without any specific emphasis on particular tasks or phases. Considering that this area embraces a broad range of proposals, we first categorise and then analyse them according to the following perspectives: phases, tasks and techniques. Regarding the phases, we discuss to what extent AutoML has covered the knowledge discovery process, as this could help practitioners to identify the appropriate initiative to use at a given moment. As for the tasks, it is interesting to provide insights on how phases have been automated. This information will contribute to understand the level of expertise required by those implementing an AutoML method. We also study those techniques being considered in AutoML and identify possible relationships among them, as well as cross-relations with phases and tasks.

Following a rigorous methodology, we conduct a systematic literature review that starts with an exhaustive search returning 31,048 manuscripts, from which we detected and examined a total of 447 primary studies between 2014 and 2021. All these studies are analysed and filtered according to a precisely defined protocol based on best practices [22]. As a result, we first categorise the main characteristics of current AutoML developments, including the terms referring to the conducted tasks, the covered phases and the most widely applied techniques. These terms are formulated and interrelated in form of a high-level taxonomy that provides a frame to understand how current AutoML studies are organised. The analysis of the primary studies also leads to interesting findings. Just to highlight a few, the qualitative analysis reveals that those phases of the knowledge extraction process considered as inherently human—e.g. the interpretation of results—have been barely considered yet. Also, it has been observed that the increasing complexity of the problems addressed has led to the automation of more than one phase simultaneously. In this vein, algorithm selection is gradually replaced by the composition of knowledge discovery workflows. It is also interesting to note that interesting relationships of techniques with phases and tasks have been found. For example, evolutionary computation and reinforcement learning was widely used by early proposals for the design of neural network architectures, while more recent studies are mainly based on gradient-based methods.

The rest of the paper is organised as follows: Section 2 provides some theoretical background on the knowledge discovery process and the most recurrent tasks within AutoML.

The research questions and the review methodology are explained in Sect. 3. Then, in Sect. 4 we introduce our AutoML taxonomy, which serves as a guideline for the analysis of the extracted primary studies. Some relevant quantitative findings are reported in Sect. 5, while Sect. 6 discusses other more qualitative findings from the analysis process. Then, Sect. 7 performs a cross-analysis between phases, tasks and techniques. The conducted analysis has served to identify open issues and future lines of research that are discussed in Sect. 8. Finally, we conclude with Sect. 9.

2 Background

The term AutoML first appeared in 2014 to refer to those approaches automating the machine learning process. In the first instance, it cannot be attributed to a single author or reference, but to a complete community. More precisely, the first occurrence was in the AutoML workshop, which has been held jointly with the *International Conference on Machine Learning* (ICML) for the last years (2014–2021). Recently, this workshop has been transformed into the AutoML conference in 2022. To the best of our knowledge, the first papers explicitly referring to AutoML were published in 2015 [14, 129]. Regardless of these dates, notice that some problems addressed by AutoML have been studied for decades, such as the selection of the best ML algorithms and the tuning of their respective hyper-parameter values.

As mentioned above, AutoML aims at automating the different phases of the knowledge discovery process (Sect. 2.1) by performing some particular tasks (Sect. 2.2) in different ways. AutoML is not only intended to be useful by both domain experts and data scientists. Indeed, certain tasks like hyper-parameter optimisation or neural architecture search (NAS) are intended to improve the performance of the generated models, even though they still require some prior knowledge on the applicable algorithms and their hyper-parameters. In this context, Olson et al. [307] stated that AutoML is not a replacement for data scientists nor ML practitioners, but a "data science assistant". Similarly, Gil et al. [13] proposed the development of interactive AutoML approaches that leverage the experience and background of both domain experts and data scientists, as they can complement and constrain the available data and solutions to their particular needs and requirements.

2.1 Phases of the knowledge discovery process

Different methodologies, each with its own phases, have been proposed to conduct the knowledge discovery process. A representative example is the Knowledge Discovery in Databases (KDD) process that was originally described by Fayyad et al. [12]. It is an iterative and interactive process whose cornerstone is the data mining phase, which aims at extracting patterns from large databases. This process starts by studying the application domain and any prior knowledge relevant to the process. Then, a target dataset is built by focusing on a subset of variables, or data samples, on which the discovery is to be performed. To ensure its quality, a number of data cleaning and transformation methods is applied. Once the dataset has been properly preprocessed, the data mining phase is conducted and the resulting patterns are evaluated. In this context, different data mining methods are identified: classification, regression, clustering, summarisation, dependency modelling or outliers detection, among others. Finally, the derived knowledge is documented, reported and incorporated to another system for further actions like decision-making.

Alternatively, the CRoss-Industry Standard Process for Data Mining (CRISP-DM) was defined by Chapman et al. [7]. This process, which is specially designed for industry, consists on a cycle encompassing six phases. The first two phases, business understanding and data understanding, are closely related to the domain understanding in KDD. In a third phase, raw data is subject to a set of transformations to improve its quality, thus helping to deploy better models during the fourth phase, which is equivalent to the data mining phase defined by KDD. The resulting models are thoroughly evaluated and the previous steps are revisited if needed, until the business objectives are met. Finally, the acquired knowledge is organised and presented to the customers. Closely related, the Sample, Explore, Modify, Model, Assess (SEMMA) [2] is another process consisting in a cycle with five stages, as referred by the five terms of the process name. In short, these stages can be seen as equivalent to those defined by KDD or CRISP-DM. Notice that all these methodologies are analogous with respect to the phases they involve, the data mining phase being their cornerstone.

2.2 Tasks in AutoML

A preliminary analysis of the manuscripts compiled by the aforementioned surveys and reviews revealed the existence of a set of common operations or tasks conducted in the field of AutoML, namely hyper-parameter optimisation, neural architecture search, algorithm selection and workflow composition. This section briefly introduces these tasks.

2.2.1 Hyper-parameter optimisation and neural architecture search

Most ML algorithms have at least one parameter (a.k.a. hyper-parameter) controlling how they behave. Illustrative examples are the kernel of a support vector machine or the maximum depth of a decision tree. Not properly tuning such hyper-parameters can greatly hamper the performance of these algorithms. In this context, the hyper-parameter optimisation (HPO) task aims at automatically selecting the hyper-parameters values that maximise the performance of a given algorithm [4]. As this problem has been studied for decades, a large number of techniques have been applied like grid search, random search, evolutionary algorithms, racing algorithms or Bayesian optimisation, among others. Hutter et al. [19] reviewed the HPO problem focusing on Bayesian optimisation and giving a brief overview of the traditional techniques. They also analysed some approaches focused on reducing the number of hyper-parameters and discussed how to combine them with traditional HPO approaches.

Recently, the availability of new scalable HPO approaches [323] has supported the development of deep neural networks, which have achieved a great success in various application domains like speech [17] and image recognition [15]. This has led to the emergence of the field known as neural architecture search, which is an instance of the HPO problem aiming at simultaneously optimising the architecture and hyper-parameters of the artificial neural networks [10]. It is worth noting that these automatically designed neural networks are comparable or even outperform hand-designed architectures [8, 483]. Elsken et al. [10] published a survey that categorises the NAS approaches according to three dimensions: search space, search strategy and performance estimation strategy. More recently, a survey mostly covering NAS, as well as some preprocessing steps of the knowledge discovery process, has appeared in the literature [16]. Regarding the NAS approaches, they are categorised according to their search space, optimisation technique and model evaluation method. As for the preprocessing, it considers both data preparation and feature engineering methods.

2.2.2 Algorithm selection

It is well known that there is no single algorithm that outperforms the others for all the available problems [36]. In the context of ML, algorithms are based on a set of assumptions about the data, i.e. their inductive bias, what makes some of them more suitable for a particular dataset. Rice [29] formalised the algorithm selection problem (AS) with the aim of selecting the best algorithm for each situation and thus limiting the aforementioned shortcoming. It is usually treated as a ML problem where the dataset characteristics, e.g. the number of attributes and classes, are used to predict the expected algorithm performance, which could be measured by its accuracy. The aim is to predict whether an algorithm—or a set of algorithms—is suitable for a specific dataset. Recently, Luo [26] reviewed a number of proposals dealing with the AS and/or the HPO during the data mining phase. More specifically, it is focused on supervised learning, i.e. classification and regression. Similarly, Tripathy and Panda [35] reviewed those proposals that exploit meta-learning techniques to carry out the selection of the best data mining algorithm. It is worth noting that the AS problem has also been considered in other application domains like the combinatorial optimisation search, which was reviewed by Kotthoff [23].

A related task is the Combined Algorithm Selection and Hyper-parameter optimisation (CASH). It was defined by Thornton et al. [34], who applied a Bayesian optimisation algorithm to jointly address the AS and HPO problems. This was achieved by treating the selection of the ML algorithm as an hyper-parameter itself, thus building a conditional space of hyper-parameters where the selection of a given algorithm triggers the optimisation of other certain hyper-parameters. In addition, assigning specific values of certain hyper-parameters may trigger the optimisation of other hyper-parameters. Previous to its formalisation, this problem was already addressed by Escalante et al. [11] with particle swarm optimisation.

2.2.3 Workflow composition

While the AS and the HPO provides a valuable assistance during the knowledge discovery process, their use is rather limited to data mining phase [6]. However, selecting and tuning the best algorithm for each phase independently could hamper the performance and even generate invalid sequences of algorithms. There are relationships, synergies and constraints between such algorithms that should be thoroughly examined to reach a pipeline with a good performance. Therefore, some authors have begun to propose methods that can automatically compose knowledge discovery workflows involving two or more phases of the process. These proposals rely on different techniques such as particle swarm optimisation [11], Bayesian optimisation [34, 129], evolutionary computation [307] or automated planning and scheduling [285].

As can be inferred from the applied techniques, workflow composition has been mainly addressed as an optimisation problem. This poses a number of new challenges related to the increase of the search space. Apart from optimising the structure of the workflow itself, the best algorithms should be selected for each phase as well. This is even more challenging when the hyper-parameters of these algorithms need to be considered. To address this problem, some proposals fix the workflow structure in advance to generate simpler workflows [30, 34, 129, 285] or give preference to the shorter ones during the optimisation process [307]. Although not specifically oriented towards the workflow composition problem, Serban et al. [31] reviewed a number of intelligent discovery assistants, which are systems that advise users during the data analysis process. Among them, they identified a set of approaches using automated planning and scheduling techniques to automatically compose ML workflows [5, 37]. More recently, Hutter et al. [20] analyse in detail some AutoML tools performing this task.

3 Review methodology

As an approach for the development of this study, we have followed the guidelines by Kitchenham and Charters [22]. They provide a rigorous and structured approach to identifying and summarising all relevant research on a particular topic, ensuring that the review process is transparent and minimises potential sources of bias. These guidelines can assist us in conducting our study in a manner that is replicable by other researchers, which is essential for building a reliable body of knowledge in the field of AutoML. Note that this approach was created according to the experience of domain experts in a variety of disciplines interested in evidence-based practice, and has undergone several revisions since it was first published. In this way, they have gradually refined a process that leads to high-quality SLRs by carrying out several steps that researchers should follow. In general, the review process comprises three main steps, namely *planning*, *conducting* and *reporting*. In terms of the *planning* step, it involves identifying the need for the review, which is motivated in Sect. 1, and defining the research questions (see Sect. 3.1) along with creating a review protocol. The *conducting* step involves identifying a wide range of primary studies pertaining to the research question and then selecting those that are most relevant (see Sect. 3.2). The quality of primary studies is evaluated, and to prevent any potential biases, the decision is solely founded on the source of the primary studies. Upon obtaining the primary studies, data extraction forms are developed to record the information accurately that researchers obtain from them (see Sect. 3.3), and later synthesised. Finally, the *reporting* step involves specifying mechanisms for dissemination and formatting the main report. To ensure replicability, the review protocol, as well as the outcomes of both the search and review processes, is available as supplementary material.

3.1 Research questions

This paper aims at responding to the following research questions (RQs):

RQ1: From the primary studies, is it possible to extract a common terminology in the field of AutoML?

In recent years, there has been a significant increase in the number of papers on AutoML. However, many of these papers are not explicitly classified as AutoML, which could lead to a blurred understanding of their scope, applicability and categorisation. The lack of a common terminology can also lead to confusion and inconsistencies in the field, hindering progress and collaboration. Therefore, we envisage that the establishment of a common taxonomy will assist researchers and practitioners in understanding and communicating each other’s work more precisely and consistently, ultimately advancing the field of AutoML.

RQ2: In quantitative terms, how does the research in AutoML evolve?

The aim of this research question is to examine the quantitative evolution of the field of AutoML. The rationale behind this lies in the fact that, while some AutoML problems, such as AS and HPO, have been studied for decades, the term AutoML as a comprehensive

Table 1 The three search strings defined

Search string #1	"machine learning" AND ("algorithm selection" OR "algorithm recommendation" OR ((hyperparameter OR "hyper parameter") AND (optimization OR tuning)) OR (auto* AND (workflow OR pipeline) AND (planning OR design OR generation OR composition)))
Search string #2	"neural architectur* search" OR (auto* AND ("neural network" OR "deep learning") AND (design OR search OR evol*) AND architectur*)
Search string #3	Auto* AND "machine learning" AND ((data AND (preprocessing OR "preprocessing" OR integration OR clean* OR transformation)) OR (feature AND (engineering OR selection OR extraction)) OR postprocessing OR "post-processing" OR (knowledge AND (filtering OR integration)) OR ((pattern OR model) AND (interpretation OR explanation)))

discipline was only introduced in 2014. By answering this question, researchers and practitioners will gain insight into the rate at which new studies are being published. Understanding the quantitative development of AutoML can also assist in identifying emerging trends and patterns in research, as well as potential areas for future work.

RQ3: Which are the phases of the knowledge discovery process covered by the different tasks performed in AutoML and what different techniques are applied?

The knowledge discovery process is a fundamental aspect of data mining and involves several key phases. By analysing the different tasks performed in AutoML, researchers and practitioners can gain a better understanding of how each phase of the knowledge discovery process is being addressed in the field, and what techniques are being applied. This information can help categorise current and future proposals, identify strengths and weaknesses in the current approaches, and suggest potential areas for improvement or future research. In addition, a cross-category analysis can assist in identifying the relationships and synergies among phases, tasks and techniques.

RQ4: What are the emerging trends and open gaps for future research?

Given that AutoML is a broad and still emerging field of research, the rationale behind this research question is to identify emerging trends and open gaps in the field of AutoML for future research. This can help guide future research directions and ensure that research efforts are focused on areas that are most likely to lead to significant advances in the field. Also, this information can help guide researchers and practitioners towards important research areas and facilitate the development of more effective and efficient AutoML systems. Overall, this research question can provide valuable insights into the current state of AutoML research and future directions for the field.

3.2 Literature search and selection strategy

The first step consists in the automatic literature search by querying the following digital libraries and citation databases: IEEE Xplore, ScienceDirect, ACM Digital Library, SpringerLink, ISI Web of Knowledge and Scopus. Table 1 shows the three final search strings executed. From the output returned by a initial version of these search strings, we conducted an analysis

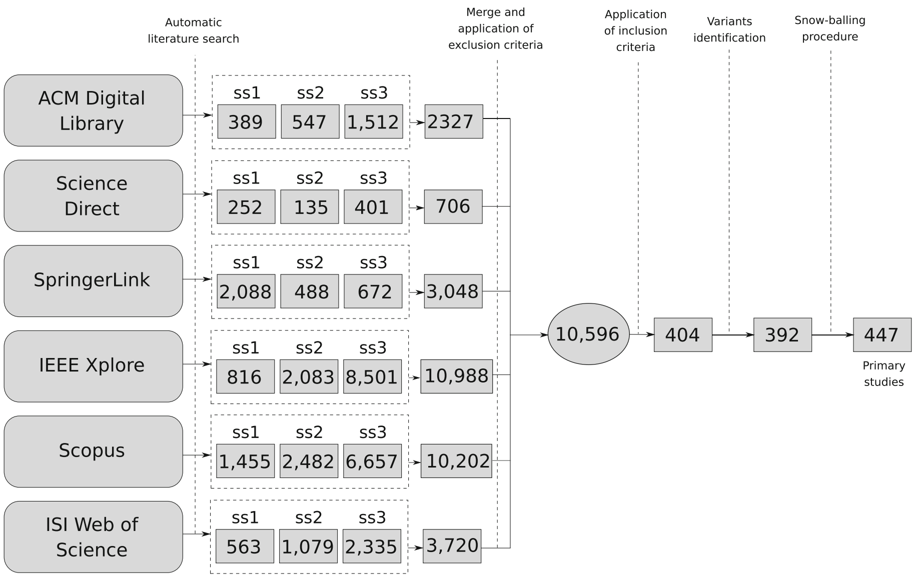


Fig. 1 Quantitative results of the search and selection steps of primary studies

to define the taxonomy that depicts the main characteristics of current AutoML approaches. This initial version of the taxonomy serves to refine these pilot search strings as well. For example, the strings are completed with sets of tasks and phases being used to determine a list of terms guiding the search. As a strategy, the search strings #1 and #2 seek to cover the main tasks of AutoML, while string #3 is more general, and includes those studies considering the automation of any phase composing the knowledge discovery process. In this way, we avoid limiting the search to the main AutoML tasks, thus offering a wider picture of the area. As shown in Fig. 1, a total of 31,048 papers are returned by these queries before removing duplicates and applying the following exclusion criteria:

1. Papers prior to the appearance of the term AutoML, i.e. 2014, are not considered.
2. Papers that are not written in English are not considered.
3. Papers with an unclear evidence of a blinded, peer-review process are not considered.
4. Papers that are not published in conferences ranked A* or A according to the CORE ranking system[1] are not considered.
5. Papers published in journals not indexed in JCR are not considered.

Once the filter was applied, we obtained a total of 10,596 candidate papers. It is important to mention that the first three criteria could be integrated into the search strings, depending on the specific query format of each database (see the review protocol in the supplementary material for further details). In order to make the results from the various digital libraries and citation databases consistent and to aid in the cleaning and homogenisation of the retrieved data, a set of Python scripts were developed. For this purpose, we invoked the bibtexparser and pybliometrics packages. Next, the titles and abstracts of candidate papers are double-checked to ensure that they meet the following inclusion criteria. Firstly, the automation must be defined for a phase of the knowledge discovery process and the contribution must be focused on making a real progress in the field of AutoML, not as much on the specific

[1] https://www.core.edu.au/conference-portal (last accessed: March 29, 2022).

artificial intelligence methods applied. Additionally, the proposal must be independent of a specific dataset, and could be of a theoretical nature.

The application of these inclusion criteria reduces the number of selected papers to 404. Also, variants of a paper presenting similar results—or a subset of the results—without a clearly differentiated contribution are removed. Note that extended versions in journals prevail over conferences. The identification of these variants concludes with 392 primary studies, which were complemented by a snow-balling procedure consisting of checking cross-references and detecting papers potentially omitted by the automatic literature search. More precisely, each referenced paper is treated as a potential primary study as if it had been obtained through the search strings. Then, inclusion and exclusion criteria are applied as already described. To avoid the exponential growth of primary studies, we limit the snow-balling procedure to one level. This is, it is not applied to primary studies obtained through it. As shown in Fig. 1, 447 primary studies were finally selected for the data collection and review process.

3.3 Data collection process

Next, we conduct the review process by thoroughly analysing the primary studies. With this aim, these papers are randomly distributed among the three authors according to the following rules: (1) every paper should be analysed by at least two authors; (2) there should be a balance among the reviewers with respect to the number of papers, their source (i.e. journal, conference) and the search string they are related to. Following these rules, all authors fill a data extraction form for each paper. Then, filled forms concerning the same paper are compared to mitigate any possible error during the review process. In the event of discrepancies, the third author is contacted to reach a consensus form. It should be noted that all papers have been objectively reviewed under strict control conditions, and no other information is inferred during the extraction process. Therefore, all data being reflected in the extraction forms is explicitly collected from primary studies. Consequently, data may be missing for some categories in some of the primary studies analysed. Notice that the authors of the papers are not contacted in order to avoid any type of subjectivity or bias.

For each primary study, comprehensive information is collected for classification regarding the phases of the knowledge discovery process being automated, the accomplished tasks and the applied techniques. In addition, information on the experimental framework and the additional material available—mostly related to replicability—is analysed too. These categories are further divided into more specific properties aligned with the taxonomy developed in Sect. 4. A more detailed description on the data extraction form, including these properties and their possible values, can be found in the additional material. Together with this information, we also retrieve a brief summary of the primary study, as well as a brief overview of the future work reported by the authors. In terms of meta-information, each primary study is annotated with the following identification data: (a) author(s) name; (b) title; (c) year of publication; (d) type of publication, i.e. conference or journal; (e) name of publication and publisher; (f) volume, issue and pages; and (g) its DOI.

4 A taxonomy for AutoML

In response to RQ1, Fig. 2 shows our taxonomy, which reflects the main classification scheme obtained from the analysis of current AutoML approaches. Our taxonomy is constructed on

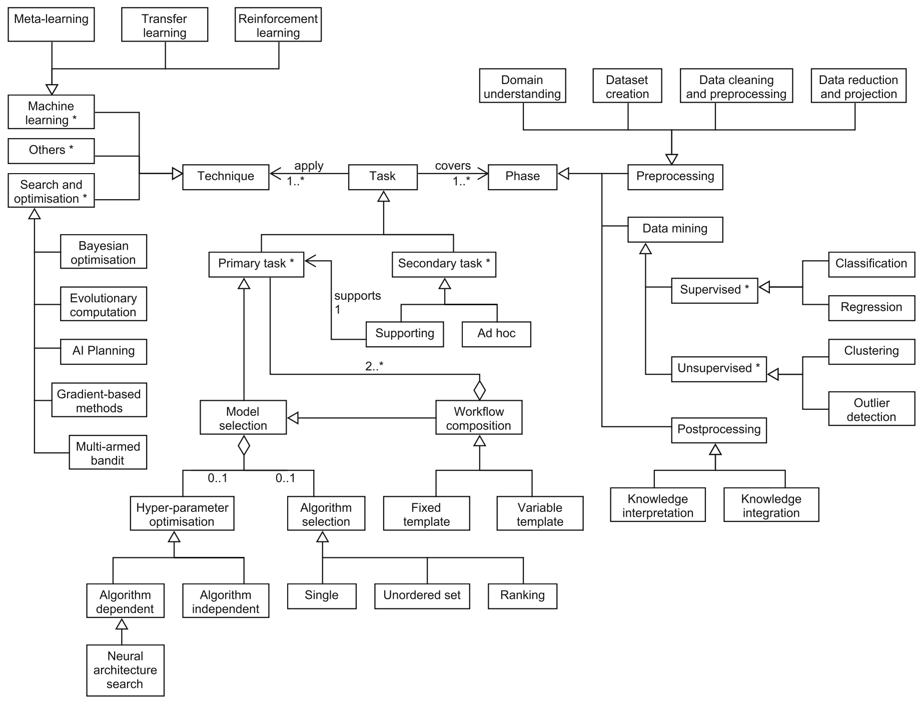


Fig. 2 Taxonomy for classifying terms in the field of AutoML

the basis of the three core AutoML terms: (1) the *phases* of the knowledge discovery process being automated; (2) the *tasks* conducted to perform such automation; and (3) the *techniques* carrying out these tasks. As explained in Sect. 3, a preliminary version of this taxonomy was first drafted from the pilot literature search and then iteratively refined along the review process as primary studies have been examined. The resulting taxonomy represents the current state of the art of the field. In Fig. 2, those terms followed by an asterisk symbol ($*$) denote extension points, i.e. those points in the classification scheme where it can be expected to extend the terminology as the number of studies in the area increases. It is worth noting that this taxonomy is not intended to provide an exhaustive list of terms of the area, but a categorisation scheme for AutoML proposals.

As pointed out in Sect. 2.1, there are different methodologies to conduct the knowledge discovery process. They all define a number of *phases* that are likely to be automated. Here, KDD, CRISP-DM and SEMMA have been considered in this taxonomy as the most general models from which to extract the different phases. Thus, we distinguish between *Preprocessing*, *Data mining* and *Postprocessing*. These three top categories are further divided into more specific terms. The preprocessing encompasses those phases that are performed prior to the generation of models, such as *Domain understanding*, *Dataset creation*, *Data cleaning and preprocessing* and *Data reduction and projection*. Regarding the data mining phase, both *Supervised*, i.e. *Classification* and *Regression*, and *Unsupervised* learning, i.e. *Clustering* and *Outlier detection*, are considered. Note that only those categories found in the literature are explicitly considered in the taxonomy, without limiting the possibility that future studies may include other types of proposals, such as pattern mining under *Unsupervised* approaches. Finally, postprocessing includes the *Knowledge interpretation* and the

Knowledge integration phases, which act on the knowledge discovered during the previous phases.

There is a set of *Task*s usually conducted in the field of AutoML, which are referred as *Primary task*s in our taxonomy. In contrast, *Secondary task*s comprise those problems that have been studied to a lesser extent in the field. Recurring primary tasks are *Algorithm selection* and *Hyper-parameter optimisation*, which aim at selecting the most accurate ML models, i.e. *Model selection*. Regarding AS, we differentiate between studies recommending a *Single* algorithm or several algorithms, either an *Unordered set* of recommendable algorithms or a *Ranking* that explicitly indicates the best ones. As for HPO, this category can be further decomposed into *Algorithm-dependent* and *Algorithm-independent* proposals, depending on whether they are designed to tune the hyper-parameters of a specific algorithm—or family of algorithms—or not. A representative example of the first group is *Neural architecture search*, which optimises both the architecture and the hyper-parameters of artificial neural networks. Furthermore, we define *Workflow composition* as a specific model selection problem, where different algorithms, and optionally their hyper-parameters values, are selected. This task can be further decomposed according to whether the generated workflows follow a *Fixed template* or a *Variable template*. It is worth noting that these tasks require a set of algorithms, so they are only applicable if a catalogue of alternative algorithms that address them is already available, for example, contrary to what happens in cases such as the knowledge interpretation. Focusing on the secondary tasks, we distinguish between *Supporting* and *Ad hoc* tasks. On the one hand, supporting tasks refer to those that are closely related to primary tasks and are usually aimed at assisting them. As an example, recommending whether it is worth investing resources in tuning the hyper-parameters of a ML algorithm may accelerate the HPO process. On the other hand, ad hoc tasks are those specifically designed for the phase being automated, e.g. generating a report explaining in natural language a ML model.

Finally, primary studies have shown a wide range of *Technique*s applied to conduct AutoML tasks. Notice that applying two or more techniques simultaneously to address a single task is a common practice. Techniques can be mostly categorised into three groups: *Search and Optimisation*, *Machine Learning* and *Others*. Regarding *Search and Optimisation*, there are various techniques available such as *Bayesian optimisation*, *Evolutionary computation*, *AI Planning*, *Gradient-based methods*, and *Multi-armed bandit* to conduct tasks such as hyper-parameter optimisation. Additionally, in *Machine Learning*, it is noteworthy how ML techniques are used to conduct specific AutoML tasks. For instance, *Meta-learning* is commonly used to develop models for AS. *Transfer learning* involves reusing knowledge gained from solving a problem to tackle a related one and is often used in conjunction with other techniques. *Reinforcement learning*, which focuses on how agents should act in an environment to maximise rewards, is typically employed to design neural network architectures. Lastly, the *Others* category encompasses techniques that are scarcely used in primary studies and do not fall under any of the previous categories. It should be noted that, due to space limitations, the taxonomy covers specific techniques within these categories based on how frequently they appear in current AutoML studies. We expect to expand these categories as new proposals emerge.

5 Quantitative analysis

In response to RQ2, this section performs a quantitative analysis of the primary studies. Firstly, the number of contributions per year and their distribution by type of publication

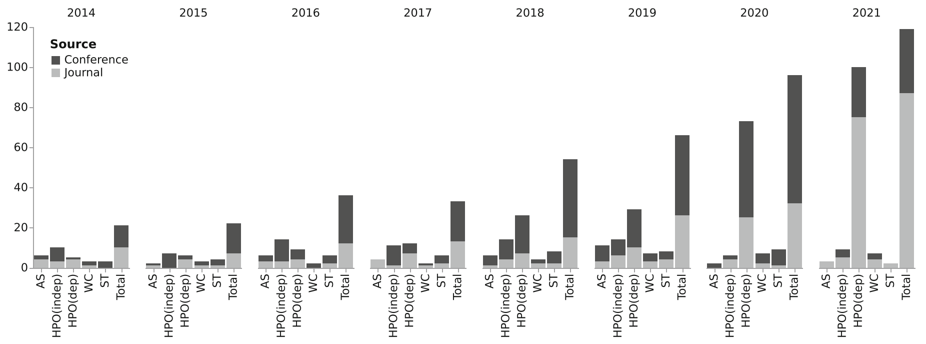


Fig. 3 Number of publications per year and task

(conference or journal) has been calculated. The idea is to identify current publication trends and check how AutoML evolves in different ways. Figure 3 shows the number of primary studies published in conferences and journals per year since the appearance of the term AutoML. For each year, the distribution of papers focused on the different AutoML tasks (see our taxonomy in Sect. 4) is depicted, where the last column shows the total amount of papers for this year. Notice that the category HPO has been decomposed into algorithm-independent tasks, *HPO(indep)*, and algorithm-dependent tasks, *HPO(dep)*. All HPO tasks together with algorithm selection (*AS*) and workflow composition (*WC*) sum those papers related to the model selection primary task. Papers on secondary tasks (*ST*) are also calculated. As can be observed, the number of primary studies published between 2014 and 2017 remains fairly steady. However, since 2018 the number of papers has increased significantly, mostly as the number of works related to *HPO(dep)* also grows. This is due to a large increase in the number of NAS proposals. Also, algorithm selection has traditionally attracted more attention than workflow composition. This is the case until 2020, when we can speculate that there could be a change in trend due to an improvement in computational resources, what offer the possibility of executing workflow composition proposals that include part of the AS and HPO tasks. In the particular case of algorithm-independent proposals, *HPO(indep)*, its number has been larger than for *HPO(dep)* until 2017. As for the studies focused on secondary tasks, its number is still limited even when it involves both supporting and ad hoc tasks. Finally, notice that the number of journal papers is notoriously greater than publications in conferences in 2021. However, we speculate that it could be due to the COVID-19 situation.

Regarding the type of publication, 55% of primary studies have been submitted to conferences, while 45% have been published in JCR journals. Also, observed that, following our search protocol, only top conferences (A* and A) are considered in the analysis. We can find 57 different conferences, including the *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (CVPR) (38 primary studies), the *International Conference on Machine Learning* (ICML) (20), the *International Joint Conference on Neural Networks* (IJCNN) (18), the *European Conference on Computer Vision* (ECCV) (16) and the *Genetic and Evolutionary Computation Conference* (GECCO) (15). As for the journals, the largest number of primary studies is published in *Neurocomputing* (19 papers), *IEEE Access* (17), *IEEE Transactions on Pattern Analysis and Machine Intelligence* (11) and *IEEE Transactions on Neural Networks and Learning Systems* (10). In general, no particular preference in terms of tasks or phases of the knowledge discovery process has been observed for one publication or the other. However, we observe that algorithm-dependent HPO approaches, which mainly include NAS-specific proposals, are commonly submitted to publications more focused on

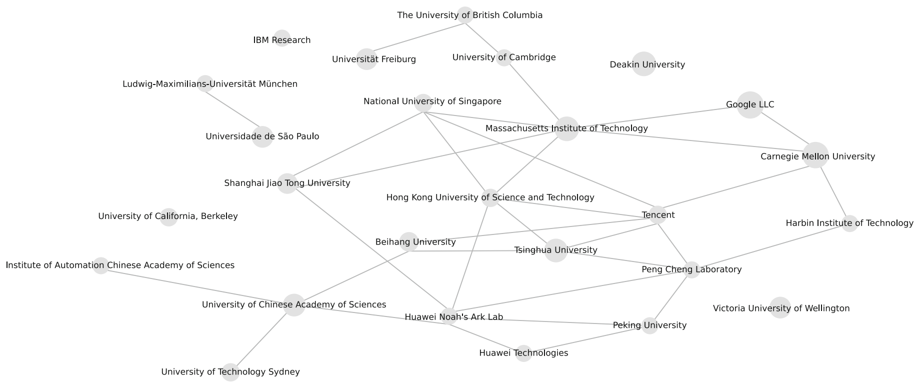

The University of British Columbia
IBM Research
Universität Freiburg
University of Cambridge
Deakin University
Ludwig-Maximilians-Universität München
National University of Singapore
Google LLC
Massachusetts Institute of Technology
Universidade de São Paulo
Carnegie Mellon University
Shanghai Jiao Tong University
Hong Kong University of Science and Technology
University of California, Berkeley
Tencent
Harbin Institute of Technology
Beihang University
Tsinghua University
Institute of Automation Chinese Academy of Sciences
Peng Cheng Laboratory
University of Chinese Academy of Sciences
Victoria University of Wellington
Huawei Noah's Ark Lab
Peking University
Huawei Technologies
University of Technology Sydney

Fig. 4 Affiliations and collaborations

artificial neural networks or their applications like CVPR (37 primary studies), *Neurocomputing* (15), ECCV (14) or IJCNN (11). Other conferences like GECCO have also accepted NAS-related papers (10), since they commonly apply evolutionary algorithms. Similarly, the largest number of WC studies is also found in GECCO (4) as well. As for algorithm-independent HPO approaches, ICML (13) and the *International Conference on Artificial Intelligence and Statistics* (AISTATS) (9) have attracted the majority of primary studies.

In average, each primary study in this area is co-authored by 4 researchers from 2 distinct institutions. Altogether, over 1000 authors have been identified, most of them having participated in just one primary study. As for international collaborations, the emergence of international networking groups such as the COSEAL (COnfiguration and SElection of ALgorithms) group[2] is noteworthy. Figure 4 shows the relationships among collaborating institutions. In this graph, nodes represent affiliations, while their size depicts the number of contributions. The edges indicate collaborations. However, due to space limitations and for the sake of readability, we do not show affiliations with less than 6 contributions. As can be seen, the institutions that have co-authored most primary studies are: *Google LLC* (16 papers in collaboration with others), *Carnegie Mellon University* (15), *Massachusetts Institute of Technology* (13), *Deakin University* (13), *Tsinghua University* (12) and *University of Chinese Academy of Sciences* (11). Of these, the *Beihang University* (20), *Tsinghua University* (18), *University of Chinese Academy of Sciences* (18) and *Tencent* (18) have been more proactive in collaborating with other institutions. Other affiliations with a large number of collaborations are *Massachusetts Institute of Technology* (16), *Peng Cheng Laboratory* (15), and *Hong Kong University of Science and Technology* (15).

6 Findings of the review process

In response to RQ3, the analysis presented next, which is founded on the proposed taxonomy in Sect. 4, covers the phases automated (Sect. 6.1), the tasks conducting such automation (Sect. 6.2), and the techniques applied (Sect. 6.3). We have also examined the experiments carried out in the primary studies, as well as their available additional materials (Sect. 6.4). Notice that, due to space limitations and for the sake of readability, Tables 2, 3 and 4 use

[2] COSEAL group: https://www.coseal.net/ (last accessed: 24/08/2022).

Table 2 Phases currently covered by AutoML

Category	%	Papers
Preprocessing (14%)		
Domain understanding	11%	[107, 111, 118, 168, 263, 287, 402]
Dataset creation	3%	[106, 268]
Data cleaning and preprocessing	48%	[59, 61, 62, 69, 97, 121, 122, 129, 140, 158, 204, 207, 273, 278, 285, 294, 297, 307, 311, 331, 333, 343, 356, 361, 375, 408, 428, 445, 450, 473]
Data reduction and projection	79%	[50, 57, 59, 61, 69, 74, 91, 93, 97, 119, 121, 122, 129, 135, 148, 158, 200, 203, 204, 206, 207, 212, 264, 273, 285, 293, 295, 297, 307, 311, 316, 317, 331, 333, 341, 343, 356, 361, 375, 377, 390, 393, 408, 428, 445, 450, 461, 473, 482, 484]
Data mining (93%)		
Classification	20%	[38, 47, 51, 57, 58, 61, 63, 67, 69, 77, 87, 95, 99, 109, 121, 122, 126, 129, 131, 135, 158, 164, 167, 185, 191, 193, 204, 207, 211, 212, 215, 222, 235, 265, 267, 269, 270, 272, 273, 283, 285, 294, 295, 297, 307, 311, 319, 321, 328, 331–333, 337, 338, 340, 342, 343, 345, 346, 352, 356, 361, 371, 375, 377, 379, 392, 395, 397, 401, 404, 408, 411, 427–429, 433, 445, 471, 473, 483, 484]
	71%	† [39–46, 48, 49, 52–56, 60, 64–66, 73, 75, 78–86, 88–94, 96, 98, 100–103, 105, 110, 112–117, 120, 123–125, 127, 130, 132–134, 136–139, 141–145, 147, 149–157, 160, 163, 165, 166, 169–172, 174–183, 186, 187, 189, 190, 192, 194, 195, 197–200, 202, 208–210, 214, 216–221, 223–231, 233, 234, 236, 238–247, 249–252, 254–262, 266, 271, 274–277, 279–282, 286, 288, 290–292, 296, 300–306, 308, 309, 313–315, 318, 320, 322, 323, 326, 329, 334–336, 339, 347–351, 353, 355, 357–360, 362–369, 372–374, 376, 378, 380–385, 387, 388, 391, 394, 396, 399, 400, 403, 405–407, 409, 410, 413–421, 423–426, 430–432, 434–444, 446–460, 462–470, 472, 474, 476–482]
Regression	6%	[61, 69, 71, 76, 95, 173, 205, 207, 211–213, 215, 232, 248, 273, 294, 297, 311, 344, 361, 370, 412, 473, 475]
	8%	† [70, 92, 104, 108, 146, 159, 161, 162, 165, 184, 188, 194, 199, 201, 214, 243, 247, 253, 284, 289, 299, 301, 304, 310, 312, 330, 354, 367, 386, 418, 438, 450]
Clustering	2%	[71, 128, 324, 325, 327, 398, 484]
	$< 1\%$	† [453, 457]
Outlier detection	$< 1\%$	[196, 237]
Postprocessing (1%)		
Knowledge interpretation	67%	[248, 389]
Knowledge integration	33%	[72]

Table 3 Tasks currently accomplished by AutoML

Category		%	Papers
P. tasks	*Algorithm selection (9%)*		
	Single algorithm	57%	[57, 58, 99, 140, 148, 162, 164, 176, 179, 196, 213, 248, 263, 267, 287, 316, 317, 342, 345, 352, 395, 398, 406]
	Set of algorithms	20%	[118, 140, 168, 205, 338, 344, 411]
	Ranking of algorithms	30%	[38, 47, 59, 95, 107, 128, 263, 287, 324, 325, 340, 397]
	Hyper-parameter optimisation (77%)		
	Alg. independent	25%	[42, 44, 56, 60, 73, 75, 83, 103, 108, 123, 133, 134, 141, 143, 144, 146, 151, 159–162, 165, 176, 178, 179, 182, 184, 185, 194, 195, 197, 198, 202, 208–210, 223, 225, 229, 231, 247, 248, 255, 260, 267, 274, 276, 296, 299, 300, 304, 306, 320, 322, 334, 338, 339, 342, 345, 350, 351, 353–355, 363, 364, 367, 373, 385, 406, 407, 414, 418–420, 431, 432, 434, 436, 437, 450, 455, 457, 464, 470]
	Alg. dependent	5%	[40, 63, 76, 167, 189, 251, 266, 275, 283, 289, 310, 327, 386, 393, 398, 401, 475]
	Neural architecture search	70%	[39, 41, 43, 45, 46, 48, 49, 51–55, 64–66, 70, 74, 78–82, 84–94, 96, 98, 100–102, 104, 105, 110, 112–117, 120, 124–127, 130–132, 136–139, 142, 145, 147, 149, 150, 152–157, 163, 166, 169, 170, 172, 173, 175, 177, 181, 183, 186–188, 190–193, 199–201, 214, 216–221, 224, 226–228, 230, 232–246, 249, 250, 252–254, 256–259, 261, 262, 265, 271, 272, 277, 279–282, 284, 286, 288, 290–292, 302, 303, 305, 308, 309, 312–315, 318, 319, 321, 323, 328, 330, 332, 336, 337, 348, 349, 357–360, 362, 365, 366, 368–372, 374, 377–382, 384, 387, 388, 391, 394, 396, 399, 400, 403–405, 409, 410, 412, 413, 415, 417, 421, 423–427, 429, 430, 435, 438–444, 446–449, 451, 452, 454, 456, 458–460, 462, 463, 465–469, 471, 472, 474, 476–483]
	Workflow composition (8%)		
	Fixed template	37%	[71, 129, 135, 212, 222, 294, 297, 343, 361, 375, 408, 445, 473]
	Variable template	63%	[57, 61, 69, 77, 97, 121, 122, 158, 204, 207, 211, 215, 273, 285, 295, 307, 311, 331, 333, 356, 433, 484]
S. tasks	*Supporting tasks (7%)*		
	Supporting AS	23%	[58, 109, 164, 278, 326, 346, 416]
	Supporting HPO	63%	[67, 171, 174, 180, 210, 260, 269, 270, 301, 329, 335, 339, 347, 376, 383, 391, 392, 416, 453]
	Supporting AWC	17%	[68, 207, 298, 422, 428]
	Ad hoc tasks (4%)		
	[50, 62, 72, 106, 111, 119, 203, 206, 248, 264, 268, 293, 341, 389, 390, 402, 461]		

Table 4 Techniques applied by AutoML primary studies

Category	%	Papers
Search and optimisation (SO) (74%)		
Evolutionary algorithms	35%	[40, 41, 44, 49, 51, 57, 60, 63, 74, 75, 77, 78, 81, 82, 86, 88, 90, 92, 96, 100, 102, 104, 105, 119, 121, 122, 125, 127, 144, 153, 156, 169, 172, 186, 193, 201, 210, 211, 214, 221, 230, 234, 240, 243–246, 249, 250, 252, 256–259, 262, 265, 271, 272, 279, 281, 284, 286, 288, 292, 307, 309, 311, 315, 319, 328, 332, 336, 337, 342, 343, 345, 348, 351, 358, 360, 366, 370–372, 378, 379, 381, 382, 384, 390, 391, 400, 401, 406, 409, 410, 421, 426, 429, 439, 440, 443, 444, 451, 454, 458, 460, 462, 465, 468, 473, 475, 478, 479, 484]
Sequential model-based opt	31%	[42, 44, 45, 48, 52, 56, 60, 61, 70, 71, 73, 96, 103, 108, 110, 115, 116, 123, 129, 135, 141, 143, 144, 146, 151, 159–162, 165, 176, 178, 182, 192, 194, 195, 197–199, 202, 204, 208–210, 212, 215, 222–225, 231, 239, 242, 247, 255, 267, 273, 274, 276, 294, 296, 297, 299, 300, 304, 306, 321, 322, 331, 333, 334, 336, 350, 354–356, 361–364, 367, 369, 373, 375, 376, 385, 398, 406, 414, 415, 418–420, 423, 431–434, 436, 455, 457, 470, 482]
Gradient methods	18%	[43, 64, 80, 85, 87, 94, 113, 117, 120, 134, 137, 149, 152, 155, 163, 166, 167, 173, 175, 177, 188, 191, 220, 241, 246, 248, 261, 266, 290, 291, 305, 310, 312–314, 320, 365, 380, 382, 386–388, 396, 403, 413, 417, 427, 435, 441, 445–447, 449, 459, 463, 466, 469, 472, 480]
Multi-armed bandit	7%	[44, 61, 79, 80, 96, 123, 124, 144, 176, 179, 184, 189, 229, 231, 247, 311, 339, 346, 352, 356, 361, 373, 482]
Swarm intelligence	6%	[76, 93, 126, 130, 131, 142, 183, 186, 192, 200, 218, 219, 235, 251, 283, 284, 289, 359, 377, 404, 405]
Random search	4%	[45, 54, 61, 96, 133, 144, 176, 179, 190, 228, 311, 424, 463]
Greedy search	3%	[69, 96, 106, 170, 181, 226, 248, 264, 374, 395, 463]
AI planning	2%	[62, 204, 207, 285, 295]
Other SO techniques	9%	[45, 46, 61, 98, 118, 119, 144, 150, 151, 154, 164, 185, 210, 216, 227, 254, 333, 338, 353, 398, 408, 415, 423, 438, 448, 453, 462, 464]
Machine learning (ML) (41%)		
Transfer learning	48%	[39, 43, 52, 54, 55, 78, 80, 82, 84, 85, 88, 90, 91, 94, 101, 112–114, 130, 137, 138, 144, 152, 153, 163, 165, 166, 175, 177, 180, 188, 217, 220, 224, 228, 233, 236, 238, 241, 243, 246, 247, 258, 282, 302, 303, 312, 314, 318, 321, 323, 332, 349, 360, 365, 368, 374, 380, 385, 387, 388, 399, 403, 409, 413, 415, 417, 421, 426, 427, 430, 435, 441, 446, 448, 449, 451, 455, 456, 459, 462, 463, 466, 468, 469, 472, 483]
Meta-learning	24%	[38, 58, 59, 67, 84, 95, 99, 120, 128, 129, 134, 140, 164, 168, 180, 196, 203, 205, 211, 213, 247, 267, 269, 270, 283, 287, 293, 312, 316, 317, 324, 325, 329, 338, 340, 344, 347, 356, 392, 397, 406, 411, 442, 445]

Table 4 continued

Category	%	Papers
Reinforcement learning	24%	[39, 53, 55, 65, 66, 83, 84, 89, 91, 114, 124, 138, 139, 145, 157, 158, 173, 187, 206, 217, 236–238, 253, 254, 277, 280, 282, 302, 303, 314, 318, 323, 375, 396, 399, 437, 465, 471, 474, 476, 477, 483]
Ensemble learning	3%	[51, 129, 131, 205, 342, 433, 482]
Other ML techniques	16%	[40, 50, 69, 89, 109, 111, 115, 132, 144, 148, 190, 261, 263, 275, 313, 327, 339, 357, 365, 376, 383, 389, 402, 405, 428, 440, 443, 446, 461, 466]
Others (12%)		
[47, 50, 58, 68, 69, 72, 75, 97, 101, 107, 112, 135, 136, 147, 152, 161, 171, 174, 207, 210, 224, 232, 236, 248, 260, 268, 278, 297, 298, 301, 313, 326, 330, 335, 339, 341, 363, 391, 393, 394, 398, 401, 407, 412, 416, 422, 425, 450, 452, 461, 467, 481]		

an identifier for each study whose respective reference can be found in the *Primary Studies* bibliography.

6.1 Automated phases

Table 2 shows the phases of the knowledge discovery process that have been automated by the primary studies. Notice that AutoML has not covered yet the automation of the entire process. Even so, a number of primary studies (13%) automate more than one phase. This is particularly common in—but not limited to—proposals that automate the preprocessing operations, usually together with the automation of the data mining phase. In general, not all phases are equally distributed, with the data mining phase receiving the most attention from the AutoML community with up to 93% of primary studies. Although to a lesser extent, AutoML proposals have also faced the automation of the preprocessing (14%) and postprocessing (1%) phases.

As for the preprocessing, the domain understanding phase is tackled by some proposals that support data scientists during data exploration [107, 111, 118, 168, 263, 287, 402]. Regarding the dataset creation problem, it has only been considered by a limited number of proposals [106, 268]. Observe that more than 81% out of the studies about preprocessing are focused on cleaning and transforming the resulting datasets. More specifically, 48% of studies refers to data cleaning and preprocessing, and typically deal with data scaling [331, 343, 375, 428, 450, 473], missing values handling [106, 129, 273, 445] or both [61, 69, 122, 158, 207, 285, 297, 333, 356, 361, 408]. Note that, with the exception of [59, 62, 74, 140, 278], these proposals are not exclusively dedicated to these particular activities, but they also cover the data mining phase. Data reduction and projection are also addressed (79%). In this case, some primary studies address the feature engineering [50, 119, 203, 206, 264, 293, 341, 390, 461]. A reduced number of studies are particularly devoted to feature selection [148, 316, 317]. As mentioned above, most of these primary studies automate some data mining phase too, such as classification [61, 122, 129, 285, 295, 297, 307, 331, 343, 375, 395, 408, 445, 450], regression [61, 207, 212, 273, 297, 450] and clustering [484]. The same happens with feature extraction, which is automated together with the data mining phase [61, 69, 74, 91, 93, 122, 129, 135, 158, 200, 297, 307, 311, 343, 356, 361, 375, 377, 408, 428, 450, 482, 484]. Just a few studies [59, 74, 393] are an exception in this case.

Turning specifically to the data mining phase, we should take into account whether it is an HPO proposal or not, since the former are usually independent of the data mining technique being conducted. Thus, these studies have been classified according to the data mining problem(s) being addressed in their experimental validation. They are shown in Table 2 as a separate list of references marked with the symbol †. Independently of the AutoML task carried out, we can observe that the automation of supervised learning, i.e. classification and regression, has attracted most of the attention (91% and 14%, respectively) compared to unsupervised techniques like clustering and outlier detection (2% and <1%). In fact, to the best of our knowledge, other unsupervised problems, such as pattern mining or association rule mining, have not been explored yet in this field. As motivated by Ferrari et al. [128], most problems addressed with unsupervised methods lack a previously known solution set. Regardless of this point, most studies automating data mining are related to HPO. However, they rarely give support to more than one phase. The same does not apply for the studies addressing other AutoML tasks. Among them, focusing on supervised learning, proposals automating classification or regression also consider some preprocessing methods (44% and 56%, respectively). More specifically, some automate the data reduction and projection phase before building the classification [57, 135, 212, 295, 484] and regression [212] models. The data cleaning and preprocessing has been considered too (83%), but always together with data reduction and projection, only with the exception of [294]. In this case, they have been applied before the classification [61, 69, 121, 122, 129, 158, 204, 207, 273, 285, 297, 307, 311, 331, 333, 343, 356, 361, 375, 408, 428, 445, 473] or regression [61, 69, 207, 273, 297, 311, 361, 473] algorithms. Regarding the unsupervised learning, no primary study has considered any preprocessing method with the only exception of [484], which automates the data reduction and projection before building the clusters.

Finally, the actions performed during the postprocessing phase are inherently human in nature, requiring a certain amount of experience, know-how and intuition. As a consequence, barely 1% of the primary studies automate this part of the knowledge discovery process. In particular, we found proposals dealing with the knowledge interpretation [248, 389] and knowledge integration [72] phases.

6.2 AutoML tasks

Automation can be conducted in different ways, depending on the operation or task performed (see Sect. 2.2). Following our taxonomy, Table 3 lists the studies that carry out some primary or secondary task(s). As can be observed, there is a large difference in the number of contributions referring to each task, with HPO attracting by far the most attention (77% of primary studies). In contrast, a smaller number of proposals conduct secondary tasks (11%), from which 7% are dedicated to support one of the primary tasks, i.e. AS, HPO or WC.

Regarding the primary AutoML tasks, 9% of the reviewed manuscripts perform algorithm selection, i.e. they recommend the best preprocessing or ML algorithms for a given dataset. Despite AS has been largely studied, recent proposals have started to address new problems like clustering [128, 324, 325, 398] or data stream forecasting [344]. Regardless of the automated phase, the most common strategy is to recommend a single algorithm (57%), but other approaches can also select an unordered set (20%) or a ranking (30%) of the best algorithms. Notice that some studies conduct more than one type of recommendation. On the other hand, HPO covers 77% out of the primary studies. According to our taxonomy, these proposals might be divided into two groups depending on whether they are independent of the algorithm whose hyper-parameters are tuned (25%) or not (75%). Also, NAS proposals, which represent the majority of studies (70%), have been differentiated from the rest of HPO algorithm-dependent approaches. It should be noted the existence of approaches optimising neural networks without considering their architecture [40, 189, 251, 266, 310, 386, 393, 475]. In addition, there are approaches optimising clustering algorithms [327, 398] and kernel-based algorithms, such as support vector machines [76, 283, 289, 343, 401], graph kernels methods [275], or conditional mean embeddings [167].

Regarding the primary tasks, workflow composition involves 8% of primary studies. Again, this task can be further decomposed into two categories depending on whether the workflow structure has been prefixed in advance (37%) or not (63%). Regardless of this sub-category, all these proposals choose between different alternative algorithms like decision tree or SVM in order to perform each step of the workflow, e.g. classification, with the exception of [71] that only optimises hyper-parameters. Most of these WC proposals optimise a workflow composed of one or more preprocessing methods and a data mining algorithm. In contrast, other approaches [57, 77, 433] only consider classification algorithms, thus defining a classifier ensemble as a type of workflow outcome. Notice that some proposals have not considered the optimisation of hyper-parameters [57, 69, 207, 484], even though some recognise it as an interesting future work [295].

Of the 11% of studies that refer to secondary tasks, 66% explore a supporting task. For those supporting AS, some proposals identify the best meta-features describing a dataset [58, 109, 326], build a portfolio of algorithms for a posterior selection [164], reduce the execution time of the task [346], or assess the quality of an algorithm [278]. As for the case of supporting HPO, these approaches have been focused on recommending whether it is worth investing resources in the task [269, 270, 347, 392], predicting whether a given configuration will lead to better results [329, 335, 391], identifying the most relevant hyper-parameter to be tuned [260, 339, 376], distributing the optimisation process in a cluster [453], or looking for speeding up the optimisation [67, 180]. Also, there are other approaches intended to assist NAS by shrinking the search space [174], predicting the performance of an ANN architecture [301, 383] or ensuring diversity during the search [171]. Finally, with respect to WC, some approaches aim at recommending the best knowledge discovery workflow [207, 428], speeding up the evaluation of workflows [298], improving interpretability [422], or generating a complete search space from a partially defined one by adding new preprocessing and ML algorithms [68]. On the other hand, ad hoc tasks suppose 34% of all primary studies covering a secondary task. Under this category, both preprocessing and postprocessing are considered. With respect to the former, there are approaches aimed at creating tabular datasets [106, 268], generating data visualisations from the datasets [111, 402], fixing data inconsistencies [62] and generating new features [50, 119, 203, 206, 264, 293, 341, 390, 461]. As for the postprocessing step, approaches explaining the resulting ML models [248, 389] or generating their source code [72] have been proposed as well.

6.3 Applied techniques and methods

Table 4 compiles the techniques that have been applied in primary studies to carry out their respective AutoML tasks of interest. These techniques are divided into two well-distinguished categories, search and optimisation (74%) and machine learning (41%), apart from other proposals that do not fall into any of these groups (12%). Also, it is worth noting that only the main techniques of each study are categorised.

Focusing on the search and optimisation category, evolutionary algorithms [3] are the most frequent techniques (35%). Among them, genetic algorithms [40, 60, 63, 74, 75, 78, 88, 96, 100, 102, 104, 125, 127, 151, 153, 169, 193, 201, 210, 214, 221, 230, 244, 249, 256–259, 265, 271, 284, 286, 292, 309, 311, 315, 336, 348, 358, 360, 366, 370, 378, 379, 384, 400, 401, 406, 439, 454, 458, 475], grammatical evolution [41, 49, 51, 105, 121, 122], evolution strategy [74, 144, 451, 479] and differential evolution [74, 332, 351] are the preferred alternatives. Also, it is common to find other approaches based on genetic programming: canonical genetic programming [57, 211, 307, 337, 390], Cartesian genetic programming [119, 371, 372], grammar-guided genetic programming [345] and strongly typed genetic programming [484]. Meta-heuristic methods like swarm intelligence (6%) and greedy search (3%) are also applied. Particularly, all the proposals based on swarm intelligence rely on particle swarm optimisation (PSO) algorithms, with the exception of [183, 192, 289] that are founded on the behaviour of ants, bats and bees, respectively. Also, note that variants for multi-objective optimisation of these methods have been applied, more specifically to genetic algorithms [60, 63, 78, 88, 153, 169, 244, 249, 256–259, 271, 311, 348, 400], genetic programming [307, 484] and PSO [93, 200, 283, 404], among others. Other methods like gradient-based methods (18%), multi-armed bandit (7%), random search (4%) and AI planning (2%) can be found among the primary studies. Finally, other search and optimisation techniques include bilevel programming [134], racing algorithms [464] or coordinate descent [98, 448], among others.

Sequential model-based optimisation [32] is the second leading technique (31%). These studies can be categorised according to their surrogate model and acquisition function. As for the surrogate model, Gaussian processes are used by most proposals (48 out of 94). Other common alternatives are artificial neural networks [116, 242, 321, 350, 362, 364, 367] and random forest [60, 70, 71, 129, 144, 204, 212, 267, 273, 297, 373, 398]. Regarding the acquisition function, most studies apply expected improvement (41 out of 94). In this case, the upper confidence bound [103, 194, 198, 202, 231, 299, 300, 385], lower confidence bound [60, 71, 304] and Thompson Sampling [162, 276, 373] are also applied. In general, sequential model-based optimisation methods evaluate a single hyper-parameter configuration at each iteration, although some approaches evaluate a batch of them [108, 110, 146, 194, 202, 299, 354, 369, 436]. Finally, it is worth noting the existence of studies that are specially committed to the definition of new surrogate models (8 out of 94) and/or acquisition functions (25 out of 94).

As for the machine learning category, transfer learning is the most recurrent technique (48%). Nevertheless, it is often used in conjunction with others techniques, such as gradient-based methods (34 out of 87), reinforcement learning (17) and evolutionary algorithms (17). Meta-learning, together with reinforcement learning, is the second leading technique (24%). These proposals can be divided according to the method that is used at the meta-level. Here, classification algorithms, such as decision trees [140, 205, 269, 270, 316, 317, 326], neural networks [99, 168, 213, 269, 287, 293, 317], random forest [95, 203, 269, 270, 317, 324, 326, 344], support vector machines [99, 196, 269, 270, 326], naive bayes [269, 270, 344], and k-nearest neighbors [67, 99, 267, 269, 270, 324, 411], are the preferred alternatives. Also, other regression algorithms, such as random forest [58, 59, 329], support vector regression [338], decision trees [329], Gaussian processes [270, 393] or XGBoost [397], are considered. In fact, Sanders et al. [347] consider both classification and regression algorithms during the learning. Although to a lesser extent, clustering algorithms [325, 442] have been used for the recommendation too. There are methods [38, 128, 340] that take advantage of the similarities between datasets—measured by a set of meta-features—without relying on the aforementioned data mining methods. Other recurrent ML approach is ensemble learning (3%), which is commonly applied to combine the best models found during AS [205, 342], NAS [51, 131, 482] or WC [129, 433]. Finally, others ML techniques (16%) include a variety of techniques like inductive logic programming [389] or multiple kernel learning [275], among others.

In Table 4, the category "*Others*" includes those proposals not applying neither SO nor ML techniques. It should be noted the existence of a wide range of techniques being applied by these proposals like collaborative filtering [135, 207], model checking [278], case-based reasoning [97], data envelopment analysis [401], conditional parallel coordinates [422] or fractional factorial design [260], among others. As the study of the area progresses, we envision that this category may be reformulated or broken down into other categories and integrated into the taxonomy.

6.4 Experimental framework and additional framework

Presenting an experimental framework to assess the validity of the proposed approach is a common practice in most primary studies (99%). In this section, we analyse the main components of the experimentation, including the type of case study, the comparative framework

Table 5 More relevant repositories in GitHub

Reference	#Commits	Last commit	Popularity
[307]	2368	06/01/2021	8497
[44]	11,877	31/03/2022	6157
[129]	2617	18/02/2022	6117
[144]	165	11/11/2019	1447
[368]	9	08/04/2019	391

or the use of statistical validation. Empirical studies, i.e. those setting a hypothesis and constructing a method to qualitatively or quantitatively validate it, and discussing findings that support such a hypothesis are a consolidated practice in the area (98%), in contrast to purely theoretical proposals (1%) or the use of sample executions (1%). A combination of a sample execution with some empirical validation can be found too [292].

In terms of the *case study* used for the experimental validation, there is a clear preference for those benchmarks and datasets obtained from the literature (98%). The UC Irvine Machine Learning[3] and the OpenML[4] are well-known and referenced repositories, as well as the CIFAR dataset[5] in the case of NAS proposals. In contrast, only 6% out of the primary studies generate their synthetic dataset. In fact, notice that, with the exception of [161, 196, 350, 357, 428], these studies combine their own datasets with others taken from the literature. Regardless of the datasets being used, the analysis of the *comparison framework* reveals that 93% of the primary studies compare their proposals against other approaches from the literature. In this vein, AutoML approaches are the preferred option (69%) for comparison. However, a large number of primary studies compare their results against other proposals not belonging to this specific area. For example, NAS approaches can be compared against hand-crafted architectures. Furthermore, we found that 47% out of studies are compared against variants of themselves.

Finally, *statistical methods* allows producing trustworthy analyses and reliable results during the comparison. However, it is notorious that only 19% of primary studies are supported by statistical evidence. Pairwise comparison (14%) and multiple comparison (6%) analyses are the most frequent choices. Regarding the pairwise comparison, the t-test is the preferred option (30 out of 62), as well as the Mann–Whitney-U test (19) and the Wilcoxon signed-rank test (16). As for multiple comparison, the Friedman (20 out of 27) and Nemenyi tests (9) are the most recurrent. Some primary studies also combine both types of statistical tests [95, 150, 161, 196, 222, 270, 316, 317, 372, 390, 434]. Finally, it is worth noting that some manuscripts claim that they conduct some sort of statistical analysis, but they do not describe the methodology explicitly [98, 164, 260, 352, 396, 422].

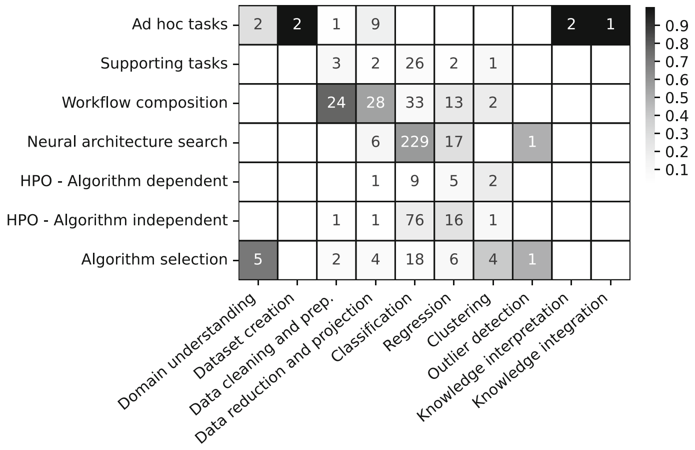
We have also inspected the additional material made available by the authors of the primary study, as a way to check their replicability and reproducibility. Software artefacts are publicly available in 33% of cases. From those, sharing the source code is a common practice (96%), as well as providing some manuals, installation instructions or software guides (77%). In this context, Table 5 shows the top-5 GitHub repositories deployed in the primary studies.[6] On the one hand, [307] and [129] are related to TPOT and auto-sklearn, respectively, which

[3] UCI Machine Learning Repository: https://archive.ics.uci.edu/ (last accessed: 30/08/2022).

[4] OpenML repository: https://www.openml.org/ (last accessed: 30/08/2022).

[5] CIFAR dataset: https://www.cs.toronto.edu/~kriz/cifar.html/ (last accessed: 30/08/2022).

[6] Github accessed on March 31, 2022. Popularity of a repository is measured in terms of its number of starts. Those repositories providing the source code of more than one project/paper are omitted.



	Domain understanding	Dataset creation	Data cleaning and prep.	Data reduction and projection	Classification	Regression	Clustering	Outlier detection	Knowledge interpretation	Knowledge integration
Ad hoc tasks	2	2	1	9					2	1
Supporting tasks			3	2	26	2	1			
Workflow composition			24	28	33	13	2			
Neural architecture search				6	229	17		1		
HPO - Algorithm dependent				1	9	5	2			
HPO - Algorithm independent			1	1	76	16	1			
Algorithm selection	5		2	4	18	6	4	1		

Fig. 5 Relation between phases and tasks

are approaches for workflow composition. On the other hand, [44] and [144] correspond to tools implementing different methods for hyper-parameter optimisation. Finally, [368] proposes a NAS method. It is worth noting that some tools [44, 129] are still under continuous development according to the frequency of commits.

Focusing on those studies providing software artefacts, in terms of their practicability and usability, it is worth noting that only 11% out of studies provide a software with an end-user friendly graphical user interface, as opposed to those proposals based on a command-line interface (73%). Moreover, in addition to software artefacts, some primary studies (36%) include other types of additional information that complement the manuscript. Among them, the most frequent are the incorporation of reports and attached documents (79%), datasets provided for replicability purposes (4%) or raw data of the experimental outcomes (19%). Finally, it is worth noting the existence of other types of supplementary material (17%) like videos [152, 161, 168, 226, 247, 267, 277, 280, 281, 304, 311, 403, 428, 449], online demo tools [111, 329] or interactive visualisation of results [350, 432].

7 Cross-category data analysis

To complement the analysis in response to RQ3, we have conducted a cross-analysis of the relationships between the different perspectives of our taxonomy: phases, tasks and techniques. This is intended to identify trends, gaps or potential limitations in the current application of AutoML. First, we contrast the data on automated phases with the tasks performed (see Sect. 7.1). Then, we examine possible relationships between tasks and the techniques applied (see Sect. 7.2).

7.1 Phases and tasks

The graphical representation in Fig. 5 illustrates the frequency with which specific tasks (rows) have been used to automate various phases (columns) of the knowledge discovery process. It is important to note that this figure presents a grid of all possible combinations of phases and tasks, which may include some that appear unlikely or even impossible. The relative frequency of application of each task for a phase is depicted following a coloured pattern, where black cells represent frequency equal to 1, i.e. only the referred task has been used in the context of the phase. Cells contain the number of primary studies found for each task-phase pair. As can be observed, if we group phases into preprocessing, data mining and postprocessing, there is no clear trend in terms of which tasks are applied in each category, although the marginal use of ad hoc tasks in the data mining phase can be highlighted, while their use is exclusive in the case of postprocessing.

Focusing on preprocessing, note that domain understanding—an inherently human activity—has been only automated by means of AS (5) and ad hoc (2) tasks, focused on selecting [107, 118, 168, 263, 287] or explicitly generating [111, 402] the best visualisation technique(s) according to the properties of the dataset. Ad hoc tasks are also defined to automate the creation of a tabular dataset with the attribute-value format required by most ML algorithms. In this vein, De Raedt et al. [106] generate a tabular dataset from a dataset formed by a collection of worksheets containing likely incomplete data. Similarly, Mahuele et al. [268] have used linked open data to build this type of datasets. A wider range of tasks have been conducted to automate data cleaning and preprocessing, as well as data reduction and projection. However, both phases coincide in the predominant use of workflow composition tasks and in the limited use or lack of use of HPO tasks. Regarding WC, algorithms handling missing values, normalising data or extracting and selecting features are commonly used for composing workflows. As for HPO, a number of primary studies [74, 93, 200, 377] are considered for data reduction using a NAS method to optimise the architecture of an autoencoder. Having model selection as a primary task, algorithm selection appears in both phases as well. In this case, studies are focused on recommending the best label noise filter [140], providing a feature selection method [148, 316, 317] and, more generally, the best preprocessing algorithm including feature extraction, normalisation, discretisation and missing values methods [59]. Also, although they are not a majority in their respective phases, some primary studies present proposals that perform secondary supporting tasks to evaluate the effectiveness of a cleansing function [278] or to recommend workflows involving both phases [207, 428]. Ad hoc tasks are proposed to fix data inconsistencies [62] and handle missing values in the context of data cleaning [106]. As for data reduction and projection, they are also considered for feature generation either by combining those existing within the original dataset [57, 74, 119, 203, 206, 264, 293, 390, 461] or by extracting them from other structured [341] or unstructured [50] external sources.

In the case of the data mining phase, it can be observed that, regardless of the particular task, classification concentrates the largest number of contributions (391), followed by regression (59). In both cases, the predominance of proposals that carry out NAS is noticeable (229 and 17, respectively). It is worth noting that 70% out of all these NAS proposals have been published since 2020. Also, the use of algorithm-independent proposals for HPO in both classification and regression is relevant. However, only [165, 194, 247, 304, 367, 418, 450] consider both phases in their experimental frameworks. For the rest of primary tasks, workflow composition has a relative importance, mostly for classification. In fact, with the exception of [71], the studies that automate regression also consider classification. Then, focusing on AS, it has covered the three subcategories described in our taxonomy, i.e. single [57, 58, 99, 162, 164, 176, 179, 213, 248, 267, 342, 345, 352, 395, 406], unordered set [205, 338, 344, 411] and ranking [38, 47, 95, 340, 397]. Regarding HPO, neural networks [40, 251, 266, 310, 386, 475] and SVM [63, 76, 283, 289, 401] hyper-parameters are commonly tuned. In the case of clustering and outlier detection, algorithm selection is the most studied task, which has been committed to recommend a single [196, 398] or a ranking [128,

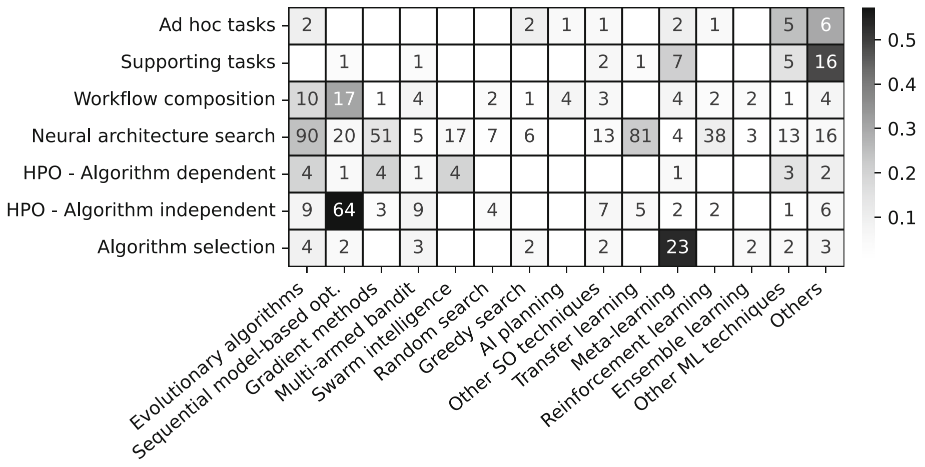


Fig. 6 Relationships between tasks and techniques

324, 325] of algorithms. Particularly, the only primary studies found for outlier detection are [196] and [237], which builds a SVM model to recommend the best outlier detection method between 14 alternatives and defines a NAS method for their detection, respectively. As for clustering, it has been also automated by HPO [327, 398, 457] and WC [71, 484] studies. Finally, there is no primary study presenting an ad hoc task in this phase, but we could find some supporting tasks for AS, HPO and WC, mainly for classification. In fact, the classification setting is also considered by the approaches addressing regression [207, 301] and clustering [453] problems.

As mentioned in Sect. 6.1, postprocessing activities have been only marginally automated, and always by means of ad hoc tasks. On the one hand, knowledge interpretation has been automated with the aim of explaining machine learning models by exploiting linked open data [389] and generating natural language descriptions [248]. On the other hand, Castro-Lopez et al. [72] defined an approach for knowledge integration that automatically generates the source code of a machine learning model.

7.2 Tasks and techniques

Similar to Sect. 7.1, Fig. 6 shows how often a certain task (rows) has been performed by applying a given technique (columns). Again, the coloured pattern represents the relative frequency of automation of each task by a technique (the darker, the closer to 1). As can be observed, evolutionary algorithms, sequential model-based optimisation (i.e. Bayesian optimisation), multi-armed bandit and meta-learning are the most recurrent techniques to address the largest number of tasks. Next, we discuss all techniques by each particular task.

Focusing on algorithm selection, meta-learning is the most common option (53%), regardless of the number of algorithms being recommended. Other techniques like evolutionary algorithms [57, 342, 345, 406] and greedy search [248, 395] have been also applied. In contrast, recommending an unordered set of algorithms has been performed only with meta-learning techniques [140, 205, 338, 344, 411], with the exception of [118]. It is worth noting the existence of other techniques to recommend either a single or a ranking of algorithms, such as sequential model-based optimisation [162, 406], multi-armed bandit [176, 179, 352], traditional classification [263] and clustering [148], or racing algorithms [164]. Finally, some approaches build ensembles with the best selected algorithms [205, 342].

Sequential model-based optimisation is the most frequently applied technique in algorithm-independent HPO (57%). Similarly to meta-learning in the context of AS, Bayesian optimisation has been established as the predominant technique for black-box optimisation, specially for the last years [19]. Following the algorithm-independent setting, other techniques considered here are evolutionary algorithms [44, 60, 75, 144, 210, 342, 345, 351, 406], multi-armed bandit methods [44, 123, 144, 184, 229, 231, 247, 339, 373], or gradient-based methods [134, 248, 320], among others. In contrast, Bayesian optimisation has been rarely applied for algorithm-dependent HPO [398]. Instead, evolutionary algorithms [40, 63, 401, 475], gradient-based method [167, 266, 310, 386] and swarm intelligence [76, 251, 283, 289] are the preferred. Focusing on NAS, the predominant technique is evolutionary algorithms (25%), as well as transfer learning (22%), gradient-based methods (14%) and reinforcement learning (10%). In the case of gradient-based methods, they have been applied mainly in recent years [25]. Also, in the particular case of transfer learning, it is worth noting that it is generally used jointly to other techniques with the aim of reusing weights of previously trained architectures.

Regarding *workflow composition*, search and optimisation techniques are the most commonly applied and, more specifically, Bayesian optimisation (31%) and evolutionary algorithms (18%). They have been used to generate knowledge discovery workflows regardless of whether their structure is fixed beforehand [71, 129, 135, 212, 222, 294, 297, 343, 361, 375, 473] or not [57, 61, 121, 122, 204, 211, 273, 307, 311, 331, 333, 356, 484]. However, within the category of search and optimisation techniques, AI planning has been only used to compose workflows whose structure is not preset [204, 207, 285, 295]. Since generating workflows of unbounded size significantly increases the search space, some proposals include mechanisms to overcome this issue, such as meta-learning to warm-start the optimisation process [129] or multi-objective evolutionary algorithms to prioritise workflows with fewer algorithms [307].

Finally, with regard to secondary tasks, we found that ad hoc tasks rely mostly on ML techniques (40%), while supporting tasks are frequently addressed with meta-learning (21%). More specifically, meta-learning has been used in AS to assess the predictive performance of meta-features [58] and also to generate an algorithm portfolio [164]. It has been also applied to predict whether it is worth providing additional resources on hyper-parameter optimisation through both classification [269, 270, 347, 392] and regression [329, 347] algorithms. Furthermore, meta-learning serves to improve the efficiency of the fine-tuning process by identifying the most relevant hyper-parameters [67] or defining a multi-fidelity framework [180]. With respect to WC, different techniques have been applied. Workflow recommendation has been addressed by means of collaborative filtering and auto-experiment [207] and a learning-to-rank method based on association rules [428], whereas conditional parallel coordinates have proven to make the process more interpretable [422]. In the case of ad hoc tasks, the automatic generation of features has been addressed by a large number of techniques like evolutionary algorithms [119, 390], meta-learning [203, 293] and reinforcement learning [206], among others. However, these tasks are commonly performed by techniques falling into the category "*Others*", such as link traversal and SPARQL queries for creating tabular datasets [268], or a custom compiler that transforms the predictive model markup language (PMML) definition of ML models into source code [72].

8 Open gaps, challenges and trends

Our findings make it clear that AutoML has drawn significant research attention during the last years. In response to Q4, we identify a number of open gaps and challenges, and speculate on related future trends.

8.1 Phases not covered

Section 6.1 reveals that AutoML has not covered the automation of the whole knowledge discovery process yet, which affects the practicality of the proposals in a realistic context. In fact, proposals have been unevenly distributed among the phases that require automation. Actually, this review has shown that 93% out of the primary studies are focused on the data mining phase. Moreover, most of them are committed to supervised methods, specially classification. This is seemingly motivated by the fact that there is an already labelled dataset and, consequently, it may be more practicable to measure and validate the performance of the automatically generated classifier. In contrast, unsupervised learning has been barely studied and problems like pattern mining are completely unexplored. Preprocessing is covered by 14% of studies, most of which are related to data cleaning and preprocessing, and data reduction and projection. Special mention should be made of those activities of the pre- and postprocessing phases that are inherently human and require intuition and know-how. For example, supporting the comprehensibility of the application domain or creating the target dataset remains a challenge for AutoML. Similarly, explaining and exploiting the acquired knowledge is still an area which requires further study, despite early efforts to explain the generated models in natural language [248] or generate their source code [72]. We speculate that in the near future there will be progress in involving humans in the automation of the different phases of the knowledge extraction process, e.g. with interactive algorithms.

8.2 Lack of interoperability

As noted in Sect. 6.2, most AutoML proposals are committed to the automation of a single phase of the knowledge discovery process. This is particularly common among studies addressing the AS and the HPO of the data mining phase. Similarly, WC proposals generally automate only the preprocessing and data mining phases, leaving aside the support for postprocessing and even some preprocessing steps like domain understanding and dataset creation. As noted above, those proposals that have addressed the automation of these phases have done so in a siloed fashion. At this point, the development of AutoML systems providing comprehensive support for the knowledge discovery process is computationally challenging. However, this holistic view seems necessary if we intend to provide real support for the work of the data scientist, and even support the democratisation of simple data science tasks. Therefore, we foresee that future proposals should consider possible interoperability between different AutoML approaches, thus enabling their reusability and replicability, improving their applicability and greatly facilitating the development and adaptation to the problem of complex workflows.

8.3 Role of the human

Analysing the AutoML definitions from the literature, we identify two potential human roles in AutoML: data scientists [129] and domain experts [307]. AutoML solutions should assist data scientists by automating and optimising those repetitive and time-consuming steps. On the other hand, to help domain experts it is necessary to go a step further, and support the automation of almost all—if not all—of the knowledge discovery process as a way to reduce the entry barrier. Regardless of the role of the participant, we found that most AutoML approaches operate as black-box methods, so that the human must simply rely on the generated models, regardless of how they can be interpreted by (or explained to) the human. Furthermore, with the proper information, the human could provide valuable inputs that could benefit the automation process. Even though, according to the primary studies, interactive approaches have been barely unexplored in the context of AutoML [161, 356], its usefulness has been demonstrated in the context of evolutionary computation [28] and machine learning [18]. Also, [13] proposed the human-guided machine learning (HGML), as an hybrid framework where an user interacts and guides an AutoML system to explore different problem settings models, features and datasets, according with their knowledge about the data available. In this regard, we speculate that putting the human in the loop in AutoML proposals will be an interesting line of future research. We also speculate that obtaining interpretable models will gain attention in the community, as will the use of both local and global explainable methods.

8.4 Empirical validation and replicability of proposals

Section 6.4 reveals the existence of some shortcomings in the experimental validation of a considerable number of studies. Firstly, we found that many approaches were only compared against some simple ML baseline and/or a variant or previous version of the same proposal. Ideally, AutoML proposals should be compared with other studies in the area addressing the same task, using the same or different technique, and ensuring fair experimental conditions (e.g. experimental environment set-up, tuning of methods, datasets, etc.). Furthermore, we found that only 19% of the primary studies support their results with an appropriate statistical framework. In terms of replicability, it is noteworthy that less than one-third of the studies provide the source code implementing the proposed method. Interestingly, we found some differences depending on tasks. For example, WC approaches tend to be better documented than proposals for HPO. Nevertheless, the top 5 most popular software repositories in the area are split between WC, HPO and NAS.

8.5 Future trends

Section 6.3 reveals that a large number of primary studies have formulated the automation of the knowledge discovery process as an optimisation problem. This is a common alternative for HPO and WC solutions. However, this can be prohibitively resource-intensive, particularly when dealing with large amounts of data (generating a single model can take hours or even days). This is a common lines of future work identified in the primary studies. Thus, we observed that some authors plan to address this issue by applying a variety of different mechanisms, such as parallelisation [61, 107, 229, 255, 300], early stopping [46, 210, 242, 355] or the reuse of knowledge from previous optimisation or learning procedures [297, 307, 351, 385, 386]. In a different field, other authors propose extending the scope of their approaches with respect to one of the AutoML dimensions (see Sect. 4). For example, a number of proposals automating the data mining phase plan to extend their work to give support to other data mining tasks apart from classification, such as regression [58, 129, 183], semi-supervised learning [129] and/or pattern learning [183]. Also, other authors pretend to cover the preprocessing [342] and postprocessing [345] phases. On the other hand, some authors working on AS [95, 397] and WC [57, 69] acknowledge the importance of tuning the hyper-parameters. Finally, although to a lesser extent, we can already find the first works focused on studying how to improve the experience of the human operating the AutoML system either through the design of graphical user interfaces [44, 317, 428] or through the application of interactive techniques [122, 287, 416].

9 Concluding remarks

The application of the most appropriate and valuable machine learning methods in a given situation often requires a considerable amount of technical knowledge, which prevents these techniques from being accessible and practicable for domain experts. At the same time, this learning process involves a number of repetitive and time-consuming tasks like hyper-parameter optimisation. AutoML has emerged to automate machine learning, thus reducing times and making it more practicable for both data scientists and domain experts. This paper presents the results of a systematic literature review of the field of AutoML. With this aim, we have followed a precise methodology, according to which we have selected and examined a total of 447 conference and journal papers as primary studies. It is worth noting that, because of the large number of studies, we had to impose some limitations on the scope of this manuscript. Specifically, we opted to limit the snow-balling procedure to a single level, which meant that we did not apply it to any papers obtained through this means. Additionally, we excluded papers that had not been published in conferences rated A* or A according to the CORE ranking system, or those published in journals not indexed in JCR. Based on a preliminary analysis of these studies, we have proposed a taxonomy for AutoML that allows categorising any AutoML approach into three different perspectives: the phase of the knowledge discovery process being automated, the task being performed, and the techniques applied to address the automation procedure.

All primary studies have been reviewed according to this three-dimensional taxonomy, in addition to considering several other aspects such as the experimental framework or the availability of additional material. As a result, we have observed that research in the area has mostly focused on the automation of the data mining phase and, to a lesser extent, also on preprocessing, particularly on data reduction and projection. In contrast, the postprocessing phase and other inherently human activities have been barely taken into consideration. It is worth noting that the number of publications in 2021 has nearly quintupled compared to 2014. A cross-dimensional analysis has allowed us to identify current gaps and trends from the AutoML literature, such as the lack of well-established tasks to automate the postprocessing or the shift from evolutionary algorithms to gradient-based methods as the most widely used technique for NAS. In terms of practicability, it is worth noting that 25% of the primary studies provide instructions on how to install (if required) and execute their proposals, less than 4% provide some graphical user interface, and only 32% make their source code publicly available. Also, our findings reveal that AutoML proposals tend to address such automation process as a black-box problem, limiting the interpretability of the resulting models or the explanation provided about the process to reach the generated solution.

Finally, from the observed evolution of the area, we can assume that AutoML is still an emerging area facing promising challenges, with a clear dominance of neural architecture search proposals in the last four years. Hopefully the proposed taxonomy herein will contribute to the classification of future proposals, and we envision that the results of this work will serve as a reference to researchers and practitioners interested in AutoML. For future research, we aim to maintain the taxonomy presented in this review up-to-date as new proposals emerge. In addition, we propose developing a repository that enables the querying of AutoML techniques based on their various dimensions of the taxonomy. Similarly, an examination of the most noteworthy AutoML tools, classified according to the taxonomy, would provide valuable insights.

Supplementary information
The complete systematic review protocol is provided separately as additional material, including detailed information about the search process, the selection of candidate and primary studies and the data extraction process. In addition, raw search outcomes and the results of the data extraction process are available. Apart from the material in this repository (https://doi.org/10.5281/zenodo.7861452), no further Electronic Supplementary Material is provided.

Acknowledgements Grant PID2020-115832GB-I00 funded by MICIN/AEI/10.13039/501100011033; Spanish Ministry of Education under the FPU program, Grant: FPU17/00799.

Author Contributions Conceptualisation: JRR, SV; Methodology: JRR; Literature search: RB; Data analysis: All authors; Writing - original draft preparation: RB; Writing - review and editing: JRR, SV; Funding acquisition: SV, JRR; Supervision: JRR, SV. All authors read and approved the final manuscript.

Funding Funding for open access publishing: Universidad de Córdoba/CBUA

Declarations

Conflicts of interest The authors have no relevant financial or non-financial interests to disclose.

References

1. Ali S, Smith K (2006) On learning algorithm selection for classification. Appl Soft Comput 6(2):119–138. https://doi.org/10.1016/j.asoc.2004.12.002
2. Azevedo A, Santos M (2008) KDD, SEMMA and CRISP-DM: a parallel overview. IADS-DM
3. Baeck T, Schwefel H (1996) Evolutionary computation: an overview. In: Proceedings of the IEEE conference on evolutionary computation, pp 20–29. https://doi.org/10.1109/ICEC.1996.542329
4. Bergstra J, Bengio Y (2012) Random search for hyper-parameter optimization. J Mach Learn Res 13:281–305
5. Bernstein A, Provost F, Hill S (2005) Toward intelligent assistance for a data mining process: an ontology-based approach for cost-sensitive classification. IEEE Trans Knowl Data Eng 17(4):503–518. https://doi.org/10.1109/TKDE.2005.67

6. Bilalli B, Abelló A, Aluja-Banet T et al (2016) Automated data pre-processing via meta-learning. In: International conference on modelling and data engineering, pp 194–208. https://doi.org/10.1007/978-3-319-45547-1_16
7. Chapman P, Clinton J, Kerber R et al (2000) CRISP-DM 1.0: step-by-step data mining guide. SPSS inc, p 16
8. Chen L, Collins M, Zhu Y et al (2018) Searching for efficient multi-scale architectures for dense image prediction. Adv. Neural Inf. Process. Syst. 31:8699–8710
9. Coleman S, Göb R, Manco G et al (2016) How can SMEs benefit from big data? Challenges and a path forward. Qual Reliab Eng Int 32(6):2151–2164. https://doi.org/10.1002/qre.2008
10. Elsken T, Metzen J, Hutter F (2019) Neural architecture search: a survey. J Mach Learn Res 20(55):1–21
11. Escalante H, Montes M, Sucar L (2009) Particle swarm model selection. J Mach Learn Res 10:405–440
12. Fayyad U, Piatetsky-Shapiro G, Smyth P (1996) Advances in knowledge discovery and data mining. chap from data mining to knowledge discovery: an overview, pp 1–34. https://doi.org/10.1609/aimag.v17i3.1230
13. Gil Y, Honaker J, Gupta S et al (2019) Towards human-guided machine learning. In: Proceedings of the 24th international conference on intelligent user interfaces, pp 614–624. https://doi.org/10.1145/3301275.3302324
14. Guyon I, Bennett K, Cawley G et al (2015) Design of the 2015 chalearn automl challenge. In: 2015 International joint conference on neural networks (IJCNN), pp 1–8. https://doi.org/10.1109/IJCNN.2015.7280767
15. He K, Zhang X, Ren S et al (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778. https://doi.org/10.1109/CVPR.2016.90
16. He X, Zhao K, Chu X (2021) AutoML: a survey of the state-of-the-art. Knowl Syst 212(106):622. https://doi.org/10.1016/j.knosys.2020.106622
17. Hinton G, Deng L, Yu D et al (2012) Deep neural networks for acoustic modeling in speech recognition. IEEE Signal Process Mag. https://doi.org/10.1109/MSP.2012.2205597
18. Holzinger A (2016) Interactive machine learning for health informatics: when do we need the human-in-the-loop? Brain Inf 3(2):119–131. https://doi.org/10.1007/s40708-016-0042-6
19. Hutter F, Lücke J, Schmidt-Thieme L (2015) Beyond manual tuning of hyperparameters. KI - Künstliche Intelligenz 29(4):329–337. https://doi.org/10.1007/s13218-015-0381-0
20. Hutter F, Kotthoff L, Vanschoren J (2019) Automatic machine learning: methods, systems, challenges
21. Jordan M, Mitchell T (2015) Machine learning: trends, perspectives, and prospects. Science 349(6245):255–260. https://doi.org/10.1126/science.aaa8415
22. Kitchenham B, Charters S (2007) Guidelines for performing systematic literature reviews in software engineering. In: EBSE 2007-001. Keele University and Durham University Joint Report https://www.elsevier.com/__data/promis_misc/525444systematicreviewsguide.pdf
23. Kotthoff L (2016) Algorithm selection for combinatorial search problems: a survey. In: Data mining and constraint programming, pp 149–190. https://doi.org/10.1609/aimag.v35i3.2460
24. Lee I, Shin Y (2020) Machine learning for enterprises: applications, algorithm selection, and challenges. Bus Horiz 63(2):157–170. https://doi.org/10.1016/j.bushor.2019.10.005
25. Liu H, Simonyan K, Yang Y (2019) DARTS: differentiable architecture search. In: International conference on learning representation. ICLR
26. Luo G (2016) A review of automatic selection methods for machine learning algorithms and hyper-parameter values. Netw Model Anal Health Inf Bioinform 5(1):18. https://doi.org/10.1007/s13721-016-0125-6
27. Patil T, Davenport T (2012) Data scientist: the sexiest job of the 21st century. Harvard Bus Rev 90(10):70–76
28. Ramirez A, Romero J, Ventura S (2018) Interactive multi-objective evolutionary optimization of software architectures. Inf Sci 463:92–109. https://doi.org/10.1016/j.ins.2018.06.034
29. Rice J (1976) The algorithm selection problem. Adv Comput 15:65–118. https://doi.org/10.1016/S0065-2458(08)60520-3
30. de Sá A, Pinto W, Oliveira L et al (2017) Recipe: a grammar-based framework for automatically evolving classification pipelines. Genet Program. https://doi.org/10.1007/978-3-319-55696-3_16
31. Serban F, Vanschoren J, Kietz J et al (2013) A survey of intelligent assistants for data analysis. ACM Comput Surv 45(3):31:1-31:35. https://doi.org/10.1145/2480741.2480748
32. Shahriari B, Swersky K, Wang Z et al (2016) Taking the human out of the loop: a review of bayesian optimization. Proc IEEE 104(1):148–175. https://doi.org/10.1109/JPROC.2015.2494218
33. Snoek J, Larochelle H, Adams R (2012) Practical bayesian optimization of machine learning algorithms. Adv Neural Inf Process Syst 25:2951–2959

34. Thornton C, Hutter F, Hoos H et al (2013) Auto-WEKA: combined selection and hyperparameter optimization of classification algorithms. In: Proceedings of the 19th ACM SIGKDD international conference on knowledge discovery and data mining, KDD '13, pp 847–855. https://doi.org/10.1145/2487575.2487629
35. Tripathy M, Panda A (2017) A study of algorithm selection in data mining using meta-learning. J Eng Sci Technol Rev. https://doi.org/10.25103/jestr.102.06
36. Wolpert D, Macready W (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82. https://doi.org/10.1109/4235.585893
37. Zakova M, Kremen P, Zelezny F et al (2011) Automating knowledge discovery workflow composition through ontology-based planning. IEEE Trans Autom Sci Eng 8(2):253–264. https://doi.org/10.1109/TASE.2010.2070838

Primary Studies (Sources Reviewed in the SLR)

38. Abdulrahman S, Brazdil P, van Rijn J et al (2018) Speeding up algorithm selection using average ranking and active testing by introducing runtime. Mach Learn 107:79–108. https://doi.org/10.1007/s10994-017-5687-8
39. Achararit P, Hanif MA, Putra RVW et al (2020) APNAS: accuracy-and-performance-aware neural architecture search for neural hardware accelerators. IEEE Access 8:16,5319-16,5334. https://doi.org/10.1109/ACCESS.2020.3022327
40. Agrawal S, Sarkar S, Alazab M et al (2021) Genetic CFL: hyperparameter optimization in clustered federated learning. Comput Intell Neurosci. https://doi.org/10.1155/2021/7156420
41. Ahmadizar F, Soltanian K, Akhlaghiantab F et al (2015) Artificial neural network development by means of a novel combination of grammatical evolution and genetic algorithm. Eng Appl Artif Intell 39:1–13. https://doi.org/10.1016/j.engappai.2014.11.003
42. Ahmed M, Vaswani S, Schmidt M (2019) Combining Bayesian optimization and Lipschitz optimization. Mach Learn. https://doi.org/10.1007/s10994-019-05833-y
43. Ahn P, Hong H, Kim J (2021) Differentiable architecture search based on coordinate descent. IEEE Access 9:48544–48554. https://doi.org/10.1109/ACCESS.2021.3068766
44. Akiba T, Sano S, Yanase T et al (2019) Optuna: a next-generation hyperparameter optimization framework. In: ACM SIGKDD international conference on knowledge discovery and data mining, pp 2623–2631. https://doi.org/10.1145/3292500.3330701
45. Akl A, El-Henawy I, Salah A et al (2019) Optimizing deep neural networks hyperparameter positions and values. J Intell Fuzzy Syst. https://doi.org/10.3233/JIFS-190033
46. Albelwi S, Mahmood A (2017) A framework for designing the architectures of deep convolutional neural networks. Entropy. https://doi.org/10.3390/e19060242
47. Ali R, Lee S, Chung T (2017) Accurate multi-criteria decision making methodology for recommending machine learning algorithm. Expert Syst Appl 71:257–278. https://doi.org/10.1016/j.eswa.2016.11.034
48. Ariafar S, Mariet Z, Brooks D et al (2021) Faster and more reliable tuning of neural networks: bayesian optimization with importance sampling. In: International Conference on Artificial Intelligence and Statistics, pp 3961–3969
49. Assunção F, Lourenço N, Machado P et al (2019) DENSER: deep evolutionary network structured representation. Genet Program Evol Mach 20:5–35. https://doi.org/10.1007/s10710-018-9339-y
50. Bafna A, Wiens J (2016) Automated feature learning: mining unstructured data for useful abstractions. In: IEEE international conference on data mining, pp 703–708. https://doi.org/10.1109/ICDM.2015.115
51. Baldominos A, Saez Y, Isasi P (2018) Evolutionary design of convolutional neural networks for human activity recognition in sensor-rich environments. Sensors (Switzerland). https://doi.org/10.3390/s18041288
52. Basha S, Vinakota S, Pulabaigari V et al (2021) AutoTune: automatically tuning convolutional neural networks for improved transfer learning. Neural Netw 133:112–122. https://doi.org/10.1016/j.neunet.2020.10.009
53. Bello I, Zoph B, Vasudevan V et al (2017) Neural optimizer search with Reinforcement learning. In: International conference on machine learning, pp 712–721
54. Bender G, Kindermans PJ, Zoph B et al (2018) Understanding and simplifying one-shot architecture search. In: International conference on machine learning, pp 883–893
55. Bender G, Liu H, Chen B et al (2020) Can weight sharing outperform random architecture search? An investigation with TuNAS. In: IEEE/CVF conference on computer vision and pattern Recognition, pp 14311–14320. https://doi.org/10.1109/CVPR42600.2020.01433

56. Berk J, Nguyen V, Gupta S et al (2019) Exploration enhanced expected improvement for bayesian optimization. In: European conference on machine learning and principles and practice of knowledge discovery in databases, pp 621–637. https://doi.org/10.1007/978-3-030-10928-8_37
57. Bi Y, Xue B, Zhang M (2019) An automated ensemble learning framework using genetic programming for image classification. In: Genetic programming and Evolvable Computing Conference, pp 365–373. https://doi.org/10.1145/3321707.3321750
58. Bilalli B, Abelló A, Aluja-Banet T (2017) On the predictive power of meta-features in OpenML. Int J Appl Math Comput Sci 27:697–712. https://doi.org/10.1515/amcs-2017-0048
59. Bilalli B, Abelló A, Aluja-Banet T et al (2018) PRESISTANT: data pre-processing assistant. In: International conference on advance information system engineering, pp 57–65. https://doi.org/10.1007/978-3-319-92901-9_6
60. Binder M, Moosbauer J, Thomas J, et al (2020) Multi-objective hyperparameter tuning and feature selection using filter ensembles. In: Genetic and evolutionary computing conference, pp 471–479. https://doi.org/10.1145/3377930.3389815
61. Binder M, Pfisterer F, Lang M et al (2021) mlr3pipelines-flexible machine learning pipelines in r. J Mach Learn Res 22:1–7
62. Boselli R, Cesarini M, Mercorio F et al (2017) An AI planning system for data cleaning. In: European conference on machine learning and knowledge discovery in databases, pp 349–353. https://doi.org/10.1007/978-3-319-71273-4_29
63. Bouraoui A, Jamoussi S, BenAyed Y (2018) A multi-objective genetic algorithm for simultaneous model and feature selection for support vector machines. Artif Intell Rev 50:261–281. https://doi.org/10.1007/s10462-017-9543-9
64. Bulat A, Martinez B, Tzimiropoulos G (2020) BATS: binary architecture search. In: European conference on computer vision, pp 309–325. https://doi.org/10.1007/978-3-030-58592-1_19
65. Cai H, Yang J, Zhang W et al (2018) Path-level network transformation for efficient architecture search. In: International conference on machine learning, pp 1069–1080
66. Cai H, Lin J, Lin Y et al (2020) AutoML for architecting efficient and specialized neural networks. IEEE Micro 40:75–82. https://doi.org/10.1109/MM.2019.2953153
67. Cai Z, Long Y, Shao L (2019) Classification complexity assessment for hyper-parameter optimization. Pattern Recognit Lett 125:396–403. https://doi.org/10.1016/j.patrec.2019.05.021
68. Cambronero J, Cito J, Rinard M (2020) AMS: Generating AutoML search spaces from weak specifications. In: ACM joint European software engineering conference and symposium on the foundations of software engineering, pp 763–774. https://doi.org/10.1145/3368089.3409700
69. Cambronero JP, Rinard MC (2019) AL: autogenerating supervised learning programs. ACM Program Lang. https://doi.org/10.1145/3360601
70. Camero A, Wang H, Alba E et al (2021) Bayesian neural architecture search using a training-free performance metric. Appl Soft Comput. https://doi.org/10.1016/j.asoc.2021.107356
71. Candelieri A, Archetti F (2019) Global optimization in machine learning: the design of a predictive analytics application. Soft Comput 23:2969–2977. https://doi.org/10.1007/s00500-018-3597-8
72. Castro-Lopez O, Vega-Lopez I (2019) Multi-target Compiler for the Deployment of Machine Learning Models. In: IEEE/ACM international symposium on code generation and optimization, pp 280–281. https://doi.org/10.1109/CGO.2019.8661199
73. Chandrashekaran A, Lane I (2017) Speeding up hyper-parameter optimization by extrapolation of learning curves using previous builds. In: European conference on machine learning and knowledge discovery in databases, pp 477–492. https://doi.org/10.1007/978-3-319-71249-9_29
74. Charte F, Rivera A, Martínez F et al (2020) EvoAAA: an evolutionary methodology for automated neural autoencoder architecture search. Integr Comput Eng 27:211–231. https://doi.org/10.3233/ICA-200619
75. Chatzimparmpas A, Martins R, Kucher K et al (2021) VisEvol: visual analytics to support hyperparameter search through evolutionary optimization. Comput Graph Forum 40(3):201–214. https://doi.org/10.1111/cgf.14300
76. Che J, Yang Y, Li L et al (2017) A modified support vector regression: integrated selection of training subset and model. Appl Soft Comput J 53:308–322. https://doi.org/10.1016/j.asoc.2016.12.053
77. Chen B, Wu H, Mo W et al (2018) Autostacker: a compositional evolutionary learning system. In: Genetic programming and evolvable computing conference, pp 402–409. https://doi.org/10.1145/3205455.3205586
78. Chen D, Shen H, Shen Y (2021) DcaNAS: efficient convolutional network design for desktop CPU platforms. Appl Intell 51(7):4353–4366. https://doi.org/10.1007/s10489-020-02133-0
79. Chen H, Zhang B, Xue S et al (2020a) Anti-bandit neural architecture search for model defense. In: European conference on computer vision, pp 70–85. https://doi.org/10.1007/978-3-030-58601-0_5

80. Chen H, Zhuo L, Zhang B et al (2021) Binarized neural architecture search for efficient object recognition. Int J Comput Vis 129:501–516. https://doi.org/10.1007/s11263-020-01379-y
81. Chen J, Gao J, Chen Y et al (2021c) GraphPAS: parallel architecture search for graph neural networks. In: International ACM SIGIR conference on research and development in information retrieval, pp 2182–2186. https://doi.org/10.1145/3404835.3463007
82. Chen M, Fu J, Ling H (2021d) One-shot neural ensemble architecture search by diversity-guided search space shrinking. In: IEEE/CVF conference on computer vision and pattern recognition, pp 16525–16534. https://doi.org/10.1109/CVPR46437.2021.01626
83. Chen S, Wu J, Liu X (2021) EMORL: effective multi-objective reinforcement learning method for hyperparameter optimization. Eng Appl Artif Intell. https://doi.org/10.1016/j.engappai.2021.104315
84. Chen X, Duan Y, Chen Z et al (2020b) CATCH: context-based meta reinforcement learning for transferrable architecture search. In: European conference on computer vision, pp 185–202. https://doi.org/10.1007/978-3-030-58529-7_12
85. Chen X, Xie L, Wu J et al (2021) Progressive DARTS: bridging the optimization gap for NAS in the wild. Int J Comput Vis 129:638–655. https://doi.org/10.1007/s11263-020-01396-x
86. Chen Y, Meng G, Zhang Q, et al (2019a) RENAS: reinforced evolutionary neural architecture search. In: IEEE/CVF computer vision and pattern recognition conference, pp 4782–4791. https://doi.org/10.1109/CVPR.2019.00492
87. Chen Y, Zhu K, Zhu L et al (2019) Automatic design of convolutional neural network for hyperspectral image classification. IEEE Trans Geosci Remote Sens 57:7048–7066. https://doi.org/10.1109/TGRS.2019.2910603
88. Chen Y, Gao R, Liu F et al (2021) ModuleNet: knowledge-inherited neural architecture search. IEEE Trans Cybern. https://doi.org/10.1109/TCYB.2021.3078573
89. Chen Y, Guo Y, Chen Q et al (2021h) Contrastive neural architecture search with neural architecture comparators. In: IEEE/CVF computer vision and pattern recognition conference, pp 9497–9506. https://doi.org/10.1109/CVPR46437.2021.00938
90. Chen Z, Li B (2020) Efficient evolution for neural architecture search. In: International joint conference on neural network. https://doi.org/10.1109/IJCNN48605.2020.9207545
91. Cheng ZQ, Wu X, Huang S et al (2018) Learning to transfer: generalizable attribute learning with multitask neural model search. In: ACM multimedia conference, pp 90–98. https://doi.org/10.1145/3240508.3240518
92. Chiu C, Zhan J (2019) An evolutionary approach to compact DAG neural network optimization. IEEE Access 7:178331–178341. https://doi.org/10.1109/ACCESS.2019.2954795
93. Chouikhi N, Ammar B, Hussain A et al (2019) Bi-level multi-objective evolution of a multi-layered echo-state network autoencoder for data representations. Neurocomputing 341:195–211. https://doi.org/10.1016/j.neucom.2019.03.012
94. Chu X, Zhou T, Zhang B et al (2020) Fair DARTS: eliminating unfair advantages in differentiable architecture search. In: European conference on computer vision, pp 465–480. https://doi.org/10.1007/978-3-030-58555-6_28
95. Cohen-Shapira N, Rokach L, Shapira B et al (2019) AutoGRD: model recommendation through graphical dataset representation. In: ACM international conference on information and knowledge management, pp 821–830. https://doi.org/10.1145/3357384.3357896
96. Cordeiro J, Raimundo A, Postolache O et al (2021) Neural architecture search for 1d CNNS. Different approaches tests and measurements. Sensors 8:9. https://doi.org/10.3390/s21237990
97. Corrales D, Ledezma A, Corrales J (2020) A case-based reasoning system for recommendation of data cleaning algorithms in classification and regression tasks. Appl Soft Comput J. https://doi.org/10.1016/j.asoc.2020.106180
98. Cortes C, Gonzalvo X, Kuznetsov V et al (2017) AdaNet: adaptive structural learning of artificial neural networks. In: International conference on machine learning, pp 1452–1466
99. Da Silva R, Canuto A, Barreto C et al (2021) Automatic recommendation method for classifier ensemble structure using meta-learning. IEEE Access 9:106254–106268. https://doi.org/10.1109/ACCESS.2021.3099689
100. Dai X, Zhang P, Wu B et al (2019a) ChamNet: towards efficient network design through platform-aware model adaptation. In: IEEE/CVF conference on computer vision and pattern recognition, pp 11390–11399. https://doi.org/10.1109/CVPR.2019.01166
101. Dai X, Chen D, Liu M et al (2020) DA-NAS: data adapted pruning for efficient neural architecture search. In: European conference on computer vision, pp 584–600. https://doi.org/10.1007/978-3-030-58583-9_35

102. Dai X, Wan A, Zhang P et al (2021) FbNetv3: joint architecture-recipe search using predictor pretraining. In: IEEE/CVF IEEE conference on computer vision and pattern recognition, pp 16271–16280. https://doi.org/10.1109/CVPR46437.2021.01601
103. Dai Z, Yu H, Low BKH et al (2019b) Bayesian optimization meets bayesian optimal stopping. In: International conference on machine learning, pp 1496–1506
104. Dale M (2018) Neuroevolution of hierarchical reservoir computers. Genet Evol Comput Conf. https://doi.org/10.1145/3205455.3205520
105. De Campos L, De Oliveira R, Roisenberg M (2016) Optimization of neural networks through grammatical evolution and a genetic algorithm. Expert Syst Appl 56:368–384. https://doi.org/10.1016/j.eswa.2016.03.012
106. De Raedt L, Blockeel H, Kolb S et al (2018) Elements of an automatic data scientist. In: International symposium on intelligent data analysis, pp 3–14. https://doi.org/10.1007/978-3-030-01768-2_1
107. Demiralp C, Haas P, Parthasarathy S et al (2017) Foresight: recommending visual insights. In: International conference on very large data bases, pp 1937–1940. https://doi.org/10.14778/3137765.3137813
108. Desautels T, Krause A, Burdick J (2014) Parallelizing exploration-exploitation tradeoffs in gaussian process bandit optimization. J Mach Learn Res 15:3873–3923
109. Dias L, Miranda P, Nascimento A et al (2021) ImageDataset2Vec: an image dataset embedding for algorithm selection. Expert Syst Appl. https://doi.org/10.1016/j.eswa.2021.115053
110. Diaz G, Fokoue-Nkoutche A, Nannicini G et al (2017) An effective algorithm for hyperparameter optimization of neural networks. IBM J Res Dev. https://doi.org/10.1147/JRD.2017.2709578
111. Dibia V, Demiralp C (2019) Data2Vis: automatic generation of data visualizations using sequence-to-sequence recurrent neural networks. IEEE Comput Graph Appl 39:33–46. https://doi.org/10.1109/MCG.2019.2924636
112. Ding M, Lian X, Yang L et al (2021a) HR-NAS: searching efficient high-resolution neural architectures with lightweight transformers. In: IEEE/CVF conference on computer vision and pattern recognition, pp 2981–2991. https://doi.org/10.1109/CVPR46437.2021.00300
113. Ding Y, Yao Q, Zhao H et al (2021b) DiffMG: differentiable meta graph search for heterogeneous graph neural networks. In: ACM SIGKDD conference on knowledge discovery and data mining, pp 279–288. https://doi.org/10.1145/3447548.3467447
114. Ding Z, Chen Y, Li N et al (2021) BNAS: efficient neural architecture search using broad scalable architecture. IEEE Trans Neural Netw Learn Syst. https://doi.org/10.1109/TNNLS.2021.3067028
115. Domhan T, Springenberg J, Hutter F (2015) Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves. In: International joint conference on artificial intelligence, pp 3460–3468
116. Dong JD, Cheng AC, Juan DC et al (2018) DPP-net: device-aware progressive search for pareto-optimal neural architectures. In: European conference on computer vision, pp 540–555. https://doi.org/10.1007/978-3-030-01252-6_32
117. Dong X, Yang Y (2019) Searching for a robust neural architecture in four GPU hours. In: IEEE/CVF conference on computer vision and pattern recognition, pp 1761–1770. https://doi.org/10.1109/CVPR.2019.00186
118. Ehsan H, Sharaf M, Chrysanthis P (2016) MuVE: efficient multi-objective view recommendation for visual data exploration. In: IEEE international conference on data engineering, pp 731–742. https://doi.org/10.1109/ICDE.2016.7498285
119. Elola A, Del Ser J, Bilbao M et al (2017) Hybridizing cartesian genetic programming and harmony search for adaptive feature construction in supervised learning problems. Appl Soft Comput J 52:760–770. https://doi.org/10.1016/j.asoc.2016.09.049
120. Elsken T, Staffler B, Metzen J et al (2020) Meta-learning of neural architectures for few-shot learning. In: IEEE/CVF conference on computer vision and pattern recognition, pp 12362–12372. https://doi.org/10.1109/CVPR42600.2020.01238
121. Estevez-Velarde S, Gutiérrez Y, Montoyo A et al (2019) AutoML strategy based on grammatical evolution: a case study about knowledge discovery from text. In: Annual meeting of the association for computational linguistics, pp 4356–4365. https://doi.org/10.18653/v1/P19-1428
122. Estevez-Velarde S, Gutiérrez Y, Almeida-Cruz Y et al (2021) General-purpose hierarchical optimisation of machine learning pipelines with grammatical evolution. Inf Sci 543:58–71. https://doi.org/10.1016/j.ins.2020.07.035
123. Falkner S, Klein A, Hutter F (2018) BOHB: robust and efficient hyperparameter optimization at scale. In: International conference on machine learning, pp 2323–2341
124. Fan Y, Zhou G, Shen J et al (2021) Toward gradient bandit-based selection of candidate architectures in AutoGAN. Soft Comput 25(6):4367–4378. https://doi.org/10.1007/s00500-020-05446-x

125. Fatyanosa T, Aritsugi M (2021) An automatic convolutional neural network optimization using a diversity-guided genetic algorithm. IEEE Access 9:91410–91426. https://doi.org/10.1109/ACCESS.2021.3091729
126. Fernandes Junior F, Yen G (2019) Particle swarm optimization of deep neural networks architectures for image classification. Swarm Evol Comput 49:62–74. https://doi.org/10.1016/j.swevo.2019.05.010
127. Fernando C, Banarse D, Reynolds M et al (2016) Convolution by evolution: differentiable pattern producing networks. In: Genetic and evolvable computing conference, pp 109–116. https://doi.org/10.1145/2908812.2908890
128. Ferrari D, De Castro L (2015) Clustering algorithm selection by meta-learning systems: a new distance-based problem characterization and ranking combination methods. Inf Sci 301:181–194. https://doi.org/10.1016/j.ins.2014.12.044
129. Feurer M, Klein A, Eggensperger K et al (2015) Efficient and robust automated machine learning. In: Annual conference on neural information processing systems, pp 2962–2970
130. Fielding B, Zhang L (2020) Evolving deep denseblock architecture ensembles for image classification. Electronics (Switzerland) 9:1–31. https://doi.org/10.3390/electronics9111880
131. Fielding B, Lawrence T, Zhang L (2019) Evolving and ensembling deep CNN architectures for image classification. In: International joint conference on neural networks. https://doi.org/10.1109/IJCNN.2019.8852369
132. Figueredo M, Paraiso E, Nievola J (2016) A constructive algorithm for neural networks inspired on decision trees and evolutionary algorithms. In: International annual conference on neural information processing systems, pp 1120–1127. https://doi.org/10.1109/IJCNN.2016.7727323
133. Florea AC, Andonie R (2019) Weighted random search for hyperparameter optimization. Int J Comput Commun Control 14:154–169. https://doi.org/10.15837/ijccc.2019.2.3514
134. Franceschi L, Frasconi P, Salzo S et al (2018) Bilevel programming for hyperparameter optimization and meta-learning. In: International conference on machine learning, pp 2537–2548
135. Fusi N, Sheth R, Elibol M (2018) Probabilistic matrix factorization for automated machine learning. In: Annual conference on neural information processing systems, pp 3348–3357
136. Gallicchio C, Micheli A, Pedrelli L (2018) Design of deep echo state networks. Neural Netw 108:33–47. https://doi.org/10.1016/j.neunet.2018.08.002
137. Gao C, Chen Y, Liu S et al (2020a) AdversarialNAS: adversarial neural architecture search for GANs. In: IEEE/CVF conference on computer vision and pattern recognition, pp 5679–5688. https://doi.org/10.1109/CVPR42600.2020.00572
138. Gao Y, Bai H, Jie Z et al (2020b) MTL-NAS: task-agnostic neural architecture search towards general-purpose multi-task learning. In: IEEE/CVF conference on computer vision and pattern recognition, pp 11540–11549. https://doi.org/10.1109/CVPR42600.2020.01156
139. Gao Y, Yang H, Zhang P et al (2020c) Graph neural architecture search. In: International Joint conferences on artificial intelligence, pp 1403–1409. https://doi.org/10.24963/ijcai.2020/195
140. Garcia L, Lorena A, Matwin S et al (2016) Ensembles of label noise filters: a ranking approach. Data Min Knowl Discov 30:1192–1216. https://doi.org/10.1007/s10618-016-0475-9
141. Gardner J, Kusner M, Xu Z et al (2014) Bayesian optimization with inequality constraints. In: International Conference on Machine Learning, pp 2581–2591
142. Garro B, Vázquez R (2015) Designing artificial neural networks using particle swarm optimization algorithms. Comput Intell Neurosci. https://doi.org/10.1155/2015/369298
143. Gelbart M, Snoek J, Adams R (2014) Bayesian optimization with unknown constraints. In: Conference on uncertainity in artificial intelligence, pp 250–259
144. Golovin D, Solnik B, Moitra S et al (2017) Google vizier: a service for black-box optimization. In: ACM SIGKDD international conference on knowledge discovery and data mining, pp 1487–1496. https://doi.org/10.1145/3097983.3098043
145. Gong C, Jiang Z, Wang D et al (2019) Mixed precision neural architecture search for energy efficient deep learning. In: IEEE/ACM international conference on computer design, pp 1–7. https://doi.org/10.1109/ICCAD45719.2019.8942147
146. González J, Dai Z, Hennig P et al (2016) Batch bayesian optimization via local penalization. In: International conference on artificial intelligence and statistics, pp 648–657
147. Gordon A, Eban E, Nachum O et al (2018) MorphNet: fast and simple resource-constrained structure learning of deep networks. In: IEEE/CVF conference on computer vision and pattern recognition, pp 1586–1595. https://doi.org/10.1109/CVPR.2018.00171
148. Goswami S, Chakrabarti A, Chakraborty B (2016) A proposal for recommendation of feature selection algorithm based on data set characteristics. J Univ Comput Sci 22:760–781. https://doi.org/10.3217/jucs-022-06-0760

149. Green S, Vineyard C, Helinski R et al (2020) RAPDARTS: resource-aware progressive differentiable architecture search. In: International joint conference on neural network. https://doi.org/10.1109/IJCNN48605.2020.9206969
150. Gulcu A, Kus Z (2020) Hyper-parameter selection in convolutional neural networks using microcanonical optimization algorithm. IEEE Access 8:52528–52540. https://doi.org/10.1109/ACCESS.2020.2981141
151. Guo B, Hu J, Wu W et al (2019) The Tabu genetic algorithm: a novel method for hyper-parameter optimization of learning algorithms. Electronics (Switzerland). https://doi.org/10.3390/electronics8050579
152. Guo M, Yang Y, Xu R et al (2020a) When NAS meets robustness: search of robust architectures against adversarial attacks. In: IEEE/CVF conference on computer vision and pattern recognition, pp 628–637. https://doi.org/10.1109/CVPR42600.2020.00071
153. Guo R, Lin C, Li C et al (2020b) Powering one-shot topological NAS with stabilized share-parameter proxy. In: European conference on computer vision, pp 625–641. https://doi.org/10.1007/978-3-030-58568-6_37
154. Gupta T, Raza K (2020) Optimizing deep feedforward neural network architecture: a Tabu search based approach. Neural Process Lett 51:2855–2870. https://doi.org/10.1007/s11063-020-10234-7
155. He C, Ye H, Shen L et al (2020) MiLeNAS: efficient neural architecture search via mixed-level reformulation. In: IEEE/CVF conference on computer vision and pattern recognition, pp 11990–11999. https://doi.org/10.1109/CVPR42600.2020.01201
156. He C, Tan H, Huang S et al (2021) Efficient evolutionary neural architecture search by modular inheritable crossover. Swarm Evol Comput. https://doi.org/10.1016/j.swevo.2021.100894
157. He Y, Lin J, Liu Z et al (2018) AMC: AutoML for model compression and acceleration on mobile devices. In: European conference on computer vision, pp 815–832. https://doi.org/10.1007/978-3-030-01234-2_48
158. Heffetz Y, Vainshtein R, Katz G et al (2020) DeepLine: AutoML tool for pipelines generation using deep reinforcement learning and hierarchical actions filtering. In: ACM SIGKDD international conference on knowledge discovery and data mining, pp 2103–2113. https://doi.org/10.1145/3394486.3403261
159. Hernández-Lobato J, Hoffman M, Ghahramani Z (2014) Predictive entropy search for efficient global optimization of black-box functions. In: Annual conference on neural information processing systems, pp 918–926
160. Hernández-Lobato J, Gelbart M, Adams R et al (2016) A general framework for constrained Bayesian optimization using information-based search. J Mach Learn Res 17:28
161. Higuchi K, Sano S, Igarashi T (2021) Interactive hyperparameter optimization with paintable timelines. In: ACM designing interactive systems nowhere and everywhere, pp 1518–1528. https://doi.org/10.1145/3461778.3462077
162. Hoffman M, Shahriari B, De Freitas N (2014) On correlation and budget constraints in model-based bandit optimization with application to automatic machine learning. In: International conference on artificial intelligence and statistics, pp 365–374
163. Hong W, Li G, Zhang W et al (2020) DropNAS: grouped operation dropout for differentiable architecture search. In: International Joint conferences on artificial intelligence, pp 2326–2332. https://doi.org/10.24963/ijcai.2020/322
164. Horn D, Schork K, Wagner T (2016) Multi-objective selection of algorithm portfolios: experimental validation. In: International conference in parallel problem solving from nature, pp 421–430. https://doi.org/10.1007/978-3-319-45823-6_39
165. Horvath S, Klein A, Richtarik P et al (2021) Hyperparameter transfer learning with adaptive complexity. In: International conference on artificial intelligence and statistics, pp 1378–1386
166. Hosseini R, Yang X, Xie P (2021) DsRNA: differentiable search of robust neural architectures. In: IEEE/CVF IEEE conference on computer vision and pattern recognition, pp 6192–6201. https://doi.org/10.1109/CVPR46437.2021.00613
167. Hsu K, Nock R, Ramos F (2019) Hyperparameter learning for conditional kernel mean embeddings with Rademacher complexity bounds. In: European conference in machine learning and knowledge discovery in databases, pp 227–242. https://doi.org/10.1007/978-3-030-10928-8_14
168. Hu K, Bakker M, Li S et al (2019a) VizML: a machine learning approach to visualization recommendation. In: Conference on human factors in computing systems. https://doi.org/10.1145/3290605.3300358
169. Hu S, Cheng R, He C et al (2021) Accelerating multi-objective neural architecture search by random-weight evaluation. Complex Intell Syst. https://doi.org/10.1007/s40747-021-00594-5
170. Hu W, Jin J, Liu TY et al (2020) Automatically design convolutional neural networks by optimization with submodularity and supermodularity. IEEE Trans Neural Netw Learn Syst 31:3215–3229. https://doi.org/10.1109/TNNLS.2019.2939157

171. Hu W, Li M, Yuan C et al (2020b) Diversity in neural architecture search. In: International joint conference on neural network. https://doi.org/10.1109/IJCNN48605.2020.9206793
172. Hu W, Zhou A, Zhang G (2020c) A classification surrogate model based evolutionary algorithm for neural network structure learning. In: International joint conference on neural network. https://doi.org/10.1109/IJCNN48605.2020.9207558
173. Hu Y, Jiang X, Liu X et al (2020d) NAS-count: counting-by-density with neural architecture search. In: European conference on computer vision, pp 747–766. https://doi.org/10.1007/978-3-030-58542-6_45
174. Hu Y, Liang Y, Guo Z et al (2020e) Angle-based search space shrinking for neural architecture search. In: European conference on computer vision, pp 119–134. https://doi.org/10.1007/978-3-030-58529-7_8
175. Hu Y, Wu X, He R (2020f) TF-NAS: rethinking three search freedoms of latency-constrained differentiable neural architecture search. In: European conference on computer vision, pp 123–139. https://doi.org/10.1007/978-3-030-58555-6_8
176. Hu Y, Liu X, Li S et al (2021) Cascaded algorithm selection with extreme-region UCB bandit. IEEE Trans Pattern Anal Mach Intell. https://doi.org/10.1109/TPAMI.2021.3094844
177. Hu Y, Wang X, Li L et al (2021) Improving one-shot NAS with shrinking-and-expanding supernet. Pattern Recognit. https://doi.org/10.1016/j.patcog.2021.108025
178. Hu YQ, Yu Y, Zhou ZH (2018) Experienced optimization with reusable directional model for hyper-parameter search. In: International Joint conferences on artificial intelligence, pp 2276–2282. https://doi.org/10.24963/ijcai.2018/315
179. Hu YQ, Yu Y, Liao JD (2019b) Cascaded algorithm-selection and hyper-parameter optimization with extreme-region upper confidence bound bandit. In: Joint conferences on artificial intelligence, pp 2528–2534. https://doi.org/10.24963/ijcai.2019/351
180. Hu YQ, Yu Y, Tu WW et al (2019c) Multi-fidelity automatic hyper-parameter tuning via transfer series expansion. In: AAAI conferences on artificial intelligence, pp 3846–3853. https://doi.org/10.1609/aaai.v33i01.33013846
181. Huang S, Li X, Cheng ZQ et al (2018) GNAS: a greedy neural architecture search method for multi-attribute learning. In: ACM multimedia conference, pp 2049–2057. https://doi.org/10.1145/3240508.3240588
182. Ilievski I, Akhtar T, Feng J et al (2017) Efficient hyperparameter optimization of deep learning algorithms using deterministic RBF surrogates. In: AAAI conferences on artificial intelligence, pp 822–829
183. Jaddi N, Abdullah S, Hamdan A (2015) Optimization of neural network model using modified bat-inspired algorithm. Appl Soft Comput J 37:71–86. https://doi.org/10.1016/j.asoc.2015.08.002
184. Jamieson K, Talwalkar A (2016) Non-stochastic best arm identification and hyperparameter optimization. In: International conference on artificial intelligence and statistics, pp 240–248
185. Ji Y, Chen Y, Fu H et al (2017) An EnKF-based scheme to optimize hyper-parameters and features for SVM classifier. Pattern Recognit 62:202–213. https://doi.org/10.1016/j.patcog.2016.08.014
186. Jiang J, Han F, Ling Q et al (2020) Efficient network architecture search via multiobjective particle swarm optimization based on decomposition. Neural Netw 123:305–316. https://doi.org/10.1016/j.neunet.2019.12.005
187. Jiang W, Yang L, Sha EM et al (2020) Hardware/software co-exploration of neural architectures. IEEE Trans Comput Des Integr Circuits Syst 39:4805–4815. https://doi.org/10.1109/TCAD.2020.2986127
188. Jie R, Gao J (2021) Differentiable neural architecture search for high-dimensional time series forecasting. IEEE Access 9:20922–20932. https://doi.org/10.1109/ACCESS.2021.3055555
189. Jie R, Gao J, Vasnev A et al (2020) HyperTube: a framework for population-based online hyperparameter optimization with resource constraints. IEEE Access 8:69038–69057. https://doi.org/10.1109/ACCESS.2020.2986456
190. Jing K, Xu J, Zhang Z (2021) A neural architecture generator for efficient search space. Neurocomputing. https://doi.org/10.1016/j.neucom.2021.10.118
191. Jing W, Ren Q, Zhou J et al (2020) AutoRSISC: automatic design of neural architecture for remote sensing image scene classification. Pattern Recognit Lett 140:186–192. https://doi.org/10.1016/j.patrec.2020.09.034
192. Jlassi S, Jdey I, Ltifi H (2021) Bayesian hyperparameter optimization of deep neural network algorithms based on ant colony optimization. In: International conference on document analysis and recognition, pp 585–594. https://doi.org/10.1007/978-3-030-86334-0_38
193. Johnson F, Valderrama A, Valle C et al (2020) Automating configuration of convolutional neural network hyperparameters using genetic algorithm. IEEE Access 8:156139–156152. https://doi.org/10.1109/ACCESS.2020.3019245
194. Joy T, Rana S, Gupta S et al (2020) Batch Bayesian optimization using multi-scale search. Knowl Syst. https://doi.org/10.1016/j.knosys.2019.06.026

195. Joy T, Rana S, Gupta S et al (2020) Fast hyperparameter tuning using Bayesian optimization with directional derivatives. Knowl Syst. https://doi.org/10.1016/j.knosys.2020.106247
196. Kandanaarachchi S, Muñoz M, Hyndman R et al (2019) On normalization and algorithm selection for unsupervised outlier detection. Data Min Knowl Discov. https://doi.org/10.1007/s10618-019-00661-z
197. Kandasamy K, Schneider J, Póczos B (2015) High dimensional Bayesian optimisation and bandits via additive models. In: International conference on machine learning, pp 295–304
198. Kandasamy K, Dasarathy G, Schneider J et al (2017) Multi-fidelity Bayesian optimisation with continuous approximations. In: International conference on machine learning, pp 2861–2878
199. Kandasamy K, Neiswanger W, Schneider J et al (2018) Neural architecture search with Bayesian optimisation and optimal transport. In: Annual conference on neural information processing systems, pp 2016–2025
200. Kanwal S, Younas I, Bashir M (2021) Evolving convolutional autoencoders using multi-objective particle swarm optimization. Comput Electr Eng. https://doi.org/10.1016/j.compeleceng.2021.107108
201. Kapanova K, Dimov I, Sellier J (2018) A genetic approach to automatic neural network architecture optimization. Neural Comput and Appl 29:1481–1492. https://doi.org/10.1007/s00521-016-2510-6
202. Kathuria T, Deshpande A, Kohli P (2016) Batched Gaussian process bandit optimization via determinantal point processes. In: Annual conference on neural information processing systems, pp 4213–4221
203. Katz G, Shin E, Song D (2017) ExploreKit: automatic feature generation and selection. In: IEEE international Conference on Data Mining, pp 979–984. https://doi.org/10.1109/ICDM.2016.176
204. Katz M, Ram P, Sohrabi S et al (2020) Exploring context-free languages via planning: the case for automating machine learning. In: International conference on automated planning and scheduling, pp 403–411. https://doi.org/10.1609/icaps.v30i1.6686
205. Khiari J, Moreira-Matias L, Shaker A et al (2019) MetaBags: bagged meta-decision trees for regression. In: European conference in machine learning and knowledge discovery in databases, pp 637–652. https://doi.org/10.1007/978-3-030-10925-7_39
206. Khurana U, Samulowitz H, Turaga D (2018) Feature engineering for predictive modeling using reinforcement learning. In: AAAI conferences on artificial intelligence, pp 3407–3414. https://doi.org/10.1609/aaai.v32i1.11678
207. Kietz JU, Serban F, Fischer S et al (2014) "Semantics inside!" but let's not tell the data miners: intelligent support for data mining. In: International conference in the semantic web: trends and challenges, pp 706–720. https://doi.org/10.1007/978-3-319-07443-6_47
208. Kim Y, Chung M (2019) An approach to hyperparameter optimization for the objective function in machine learning. Electronics (Switzerland). https://doi.org/10.3390/electronics8111267
209. Klein A, Falkner S, Bartels S et al (2017) Fast Bayesian optimization of machine learning hyperparameters on large datasets. In: International conference on artificial intelligence and statistics
210. Koch P, Wujek B, Golovidov O et al (2018) Autotune: a derivative-free optimization framework for hyperparameter tuning. In: ACM SIGKDD conference on knowledge discovery and data mining, pp 443–452. https://doi.org/10.1145/3219819.3219837
211. Kordík P, Černý J, Frýda T (2018) Discovering predictive ensembles for transfer learning and meta-learning. Mach Learn 107:177–207. https://doi.org/10.1007/s10994-017-5682-0
212. Kotthoff L, Thornton C, Hoos H et al (2017) Auto-WEKA 2.0: automatic model selection and hyperparameter optimization in WEKA. J Mach Learn Res 18:1–5
213. Kuck M, Crone S, Freitag M (2016) Meta-learning with neural networks and landmarking for forecasting model selection an empirical evaluation of different feature sets applied to industry data. In: International joint conference on neural networks, pp 1499–1506. https://doi.org/10.1109/IJCNN.2016.7727376
214. Kumar P, Batra S, Raman B (2021) Deep neural network hyper-parameter tuning through twofold genetic approach. Soft Comput 25(13):8747–8771. https://doi.org/10.1007/s00500-021-05770-w
215. Lacoste A, Larochelle H, Marchand M et al (2014) Sequential model-based ensemble optimization. In: Conference on uncertainity in artificial intelligence, pp 440–448
216. Lakhmiri D, Digabel S, Tribes C (2021) HyperNOMAD: hyperparameter optimization of deep neural networks using mesh adaptive direct search. ACM Trans Math Softw. https://doi.org/10.1145/3450975
217. Laube K, Zell A (2019) ShuffleNASNets: efficient CNN models through modified efficient neural architecture search. In: International joint conference on neural networks. https://doi.org/10.1109/IJCNN.2019.8852294
218. Lawrence T, Zhang L, Lim C et al (2021) Particle swarm optimization for automatically evolving convolutional neural networks for image classification. IEEE Access 9:14369–14386. https://doi.org/10.1109/ACCESS.2021.3052489

219. Lawrence T, Zhang L, Rogage K et al (2021) Evolving deep architecture generation with residual connections for image classification using particle swarm optimization. Sensors. https://doi.org/10.3390/s21237936
220. Lee J, Rhim J, Kang D et al (2021) S3NAS: fast hardware-aware neural architecture search methodology. IEEE Trans Comput Des Integr Circuits Syst. https://doi.org/10.1109/TCAD.2021.3134843
221. Lee S, Kim J, Kang H et al (2021) Genetic algorithm based deep learning neural network structure and hyperparameter optimization. Appl Sci (Switzerland) 11(2):1–12. https://doi.org/10.3390/app11020744
222. Lévesque JC, Gagné C, Sabourin R (2016) Bayesian hyperparameter optimization for ensemble learning. In: Conference on uncertainity in artificial intelligence, pp 437–446
223. Li C, Gupta S, Rana S et al (2017) High dimensional Bayesian optimization using dropout. In: International joint conferences on artificial intelligence, pp 2096–2102. https://doi.org/10.24963/ijcai.2017/291
224. Li C, Fan X, Geng Y et al (2020) ENAS oriented layer adaptive data scheduling strategy for resource limited hardware. Neurocomputing 381:29–39. https://doi.org/10.1016/j.neucom.2019.11.005
225. Li CL, Kandasamy K, Póczos B et al (2016) High dimensional Bayesian optimization via restricted projection pursuit models. In: International conference on artificial intelligence and statistics, pp 884–892
226. Li G, Qian G, Delgadillo I et al (2020b) SGAS: sequential greedy architecture search. In: IEEE/CVF conference on computer vision and pattern recognition, pp 1617–1627. https://doi.org/10.1109/CVPR42600.2020.00169
227. Li G, Mandal S, Ogras U et al (2021) FLASH: fast neural architecture search with hardware optimization. ACM Trans Embed Comput Syst. https://doi.org/10.1145/3476994
228. Li L, Talwalkar A (2019) Random search and reproducibility for neural architecture search. In: Conference on uncertainity in artificial intelligence
229. Li L, Jamieson K, DeSalvo G et al (2018) Hyperband: a novel bandit-based approach to hyperparameter optimization. J Mach Learn Res 18:1–52
230. Li S, Sun Y, Yen G et al (2021) Automatic design of convolutional neural network architectures under resource constraints. IEEE Trans Neural Netw Learn Syst. https://doi.org/10.1109/TNNLS.2021.3123105
231. Li T, Zhong J, Liu J, et al (2018b) Ease.ml: towards multitenant resource sharing for machine learning workloads. In: International conference on very large data bases, pp 607–620. https://doi.org/10.1145/3177732.3177737
232. Li T, Zhang J, Bao K et al (2020c) AutoST: efficient neural architecture search for spatio-temporal prediction. In: ACM SIGKDD international conference on knowledge discovery and data mining, pp 794–802. https://doi.org/10.1145/3394486.3403122
233. Li X, Lin C, Li C et al (2020d) Improving one-shot NAS by suppressing the posterior fading. In: IEEE/CVF conference on computer vision and pattern recognition, pp 13833–13842. https://doi.org/10.1109/CVPR42600.2020.01385
234. Li Y, King I (2020) AutoGraph: automated graph neural network. In: International conference on neural information processing, pp 189–201. https://doi.org/10.1007/978-3-030-63833-7_16
235. Li Y, Xiao J, Chen Y et al (2019) Evolving deep convolutional neural networks by quantum behaved particle swarm optimization with binary encoding for image classification. Neurocomputing 362:156–165. https://doi.org/10.1016/j.neucom.2019.07.026
236. Li Y, Jin X, Mei J et al (2020e) Neural architecture search for lightweight non-local networks. In: IEEE/CVF conference on computer vision and pattern recognition, pp 10294–10303. https://doi.org/10.1109/CVPR42600.2020.01031
237. Li Y, Chen Z, Zha D et al (2021c) AutoOD: neural architecture search for outlier detection. In: IEEE international conference on data engineering, pp 2117–2122. https://doi.org/10.1109/ICDE51399.2021.00210
238. Li Y, Dong M, Xu Y et al (2021) Neural architecture tuning with policy adaptation. Neurocomputing. https://doi.org/10.1016/j.neucom.2021.10.095
239. Li Z, Xi T, Deng J et al (2020f) GP-NAS: Gaussian process based neural architecture search. In: IEEE/CVF conference on computer vision and pattern recognition, pp 11930–11939. https://doi.org/10.1109/CVPR42600.2020.01195
240. Liang J, Meyerson E, Hodjat B et al (2019) Evolutionary neural automl for deep learning. In: Genetic and evolvable computing conference, pp 401–409. https://doi.org/10.1145/3321707.3321721
241. Liang Y, Lu L, Jin Y et al (2021) An efficient hardware design for accelerating sparse CNNs with NAS-based models. IEEE Trans Comput Des Integr Circuits Syst 41(3):597–613. https://doi.org/10.1109/TCAD.2021.3066563
242. Liu C, Zoph B, Neumann M et al (2018a) Progressive neural architecture search. In: European conference on computer vision, pp 19–35. https://doi.org/10.1007/978-3-030-01246-5_2

243. Liu H, Nicolae B, Di S et al (2021a) Accelerating DNN architecture search at scale using selective weight transfer. In: IEEE international conference on cluster computing, pp 82–93. https://doi.org/10.1109/Cluster48925.2021.00051
244. Liu J, Jin Y (2021) Multi-objective search of robust neural architectures against multiple types of adversarial attacks. Neurocomputing 453:73–84. https://doi.org/10.1016/j.neucom.2021.04.111
245. Liu J, Gong M, Miao Q et al (2018) Structure learning for deep neural networks based on multiobjective optimization. IEEE Trans Neural Netw Learn Syst 29:2450–2463. https://doi.org/10.1109/TNNLS.2017.2695223
246. Liu J, Zhou S, Wu Y et al (2021) Block proposal neural architecture search. IEEE Trans Image Process 30:15–25. https://doi.org/10.1109/TIP.2020.3028288
247. Liu Y, Wang X, Xu X et al (2021c) Meta hyperparameter optimization with adversarial proxy subsets sampling. In: Conference on information and knowledge management, pp 1109–1118. https://doi.org/10.1145/3459637.3482368
248. Lloyd J, Duvenaud D, Grosse R et al (2014) Automatic construction and natural-language description of nonparametric regression models. In: AAAI conference on artificial intelligence, pp 1242–1250
249. Loni M, Sinaei S, Zoljodi A et al (2020) DeepMaker: a multi-objective optimization framework for deep neural networks in embedded systems. Microprocess Microsyst. https://doi.org/10.1016/j.micpro.2020.102989
250. Lorenzo P, Nalepa J (2018) Memetic evolution of deep neural networks. In: Genetic and evolvable computing conference, pp 505–512. https://doi.org/10.1145/3205455.3205631
251. Lorenzo P, Nalepa J, Kawulok M et al (2017) Particle swarm optimization for hyper-parameter selection in deep neural networks. In: Genetic and evolvable computing conference, pp 481–488. https://doi.org/10.1145/3071178.3071208
252. Louati H, Bechikh S, Louati A et al (2021) Deep convolutional neural network architecture design as a bi-level optimization problem. Neurocomputing 439:44–62. https://doi.org/10.1016/j.neucom.2021.01.094
253. Lu H, Du M, He X et al (2021) An adaptive neural architecture search design for collaborative edge-cloud computing. IEEE Netw 35(5):83–89. https://doi.org/10.1109/MNET.201.2100069
254. Lu X, Huang H, Dong W et al (2020a) Beyond network pruning: a joint search-and-training approach. In: International joint conference on artificial intelligence, pp 2583–2590. https://doi.org/10.24963/ijcai.2020/358
255. Lu Z, Chen L, Chiang CK et al (2019a) Hyper-parameter tuning under a budget constraint. In: International joint conference on artificial intelligence, pp 5744–5750. https://doi.org/10.24963/ijcai.2019/796
256. Lu Z, Whalen I, Boddeti V et al (2019b) NSGA-Net: neural architecture search using multi-objective genetic algorithm. In: Genetic and evolvable computing conference, pp 419–427. https://doi.org/10.1145/3321707.3321729
257. Lu Z, Deb K, Goodman E et al (2020b) NSGANetV2: evolutionary multi-objective surrogate-assisted neural architecture search. In: European conference on computer vision, pp 35–51. https://doi.org/10.1007/978-3-030-58452-8_3
258. Lu Z, Sreekumar G, Goodman E et al (2021) Neural architecture transfer. IEEE Trans Pattern Anal Mach Intell 43(9):2971–2989. https://doi.org/10.1109/TPAMI.2021.3052758
259. Lu Z, Whalen I, Dhebar Y et al (2021) Multiobjective evolutionary design of deep convolutional neural networks for image classification. IEEE Trans Evol Comput 25(2):277–291. https://doi.org/10.1109/TEVC.2020.3024708
260. Lujan-Moreno G, Howard P, Rojas O et al (2018) Design of experiments and response surface methodology to tune machine learning hyperparameters, with a random forest case-study. Expert Syst Appl 109:195–205. https://doi.org/10.1016/j.eswa.2018.05.024
261. Luo R, Tian F, Qin T et al (2018a) Neural architecture optimization. In: Annual conference on neural information processing systems, pp 7816–7827
262. Luo X, Liu D, Huai S et al (2021) Designing efficient DNNs via hardware-aware neural architecture search and beyond. IEEE Trans Comput Des Integr Circuits Syst. https://doi.org/10.1109/TCAD.2021.3100249
263. Luo Y, Qin X, Tang N et al (2018b) Deepeye: towards automatic data visualization. In: IEEE international conference on data engineering, pp 101–112. https://doi.org/10.1109/ICDE.2018.00019
264. Luo Y, Yao Q, Wang M et al (2019) Autocross: automatic feature crossing for tabular data in real-world applications. In: ACM SIGKDD international conference on knowledge discovery and data mining, pp 1936–1945. https://doi.org/10.1145/3292500.3330679
265. Ma B, Li X, Xia Y et al (2020) Autonomous deep learning: a genetic DCNN designer for image classification. Neurocomputing 379:152–161. https://doi.org/10.1016/j.neucom.2019.10.007

266. Maclaurin D, Duvenaud D, Adams R (2015) Gradient-based hyperparameter optimization through reversible learning. In: International conference on machine learning, ICML'15, pp 2113–2122
267. Maher M, Sakr S (2019) SmartML: a meta learning-based framework for automated selection and hyperparameter tuning for machine learning algorithms. In: International conference on extending database technology, pp 554–557. https://doi.org/10.5441/002/edbt.2019.54
268. Mahule R, Vyas O (2016) Leveraging linked open data information extraction for data mining applications. Turkish J Electr Eng Comput Sci 24:4874–4884. https://doi.org/10.3906/elk-1412-28
269. Mantovani R, Rossi A, Vanschoren J et al (2015) To tune or not to tune: recommending when to adjust SVM hyper-parameters via meta-learning. In: International joint conference on neural networks. https://doi.org/10.1109/IJCNN.2015.7280644
270. Mantovani R, Rossi A, Alcobaça E et al (2019) A meta-learning recommender system for hyperparameter tuning: predicting when tuning improves SVM classifiers. Inf Sci 501:193–221. https://doi.org/10.1016/j.ins.2019.06.005
271. Marchisio A, Massa A, Mrazek V et al (2020) NASCaps: a framework for neural architecture search to optimize the accuracy and hardware efficiency of convolutional capsule networks. In: IEEE/ACM international conference on computer design. https://doi.org/10.1145/3400302.3415731
272. Martín A, Lara-Cabrera R, Fuentes-Hurtado F et al (2018) EvoDeep: a new evolutionary approach for automatic deep neural networks parametrisation. J Parallel Distrib Comput 117:180–191. https://doi.org/10.1016/j.jpdc.2017.09.006
273. Martin Salvador M, Budka M, Gabrys B (2019) Automatic composition and optimization of multicomponent predictive systems with an extended auto-WEKA. IEEE Trans Autom Sci Eng 16:946–959. https://doi.org/10.1109/TASE.2018.2876430
274. Martinez-Cantin R (2019) Funneled Bayesian optimization for design, tuning and control of autonomous systems. IEEE Trans Cybern 49:1489–1500. https://doi.org/10.1109/TCYB.2018.2805695
275. Massimo C, Navarin N, Sperduti A (2016) Hyper-parameter tuning for graph kernels via multiple kernel learning. In: International conference on neural information processing, pp 214–223. https://doi.org/10.1007/978-3-319-46672-9_25
276. McInerney J (2017) An empirical bayes approach to optimizing machine learning algorithms. In: Annual conference on neural information processing systems, pp 2713–2722
277. Mendis H, Kang CK, Hsiu PC (2021) Intermittent-aware neural architecture search. ACM Trans Embed Comput Syst. https://doi.org/10.1145/3476995
278. Mezzanzanica M, Boselli R, Cesarini M et al (2015) A model-based evaluation of data quality activities in KDD. Inf Process Manag 51:144–166. https://doi.org/10.1016/j.ipm.2014.07.007
279. Miao F, Yao L, Zhao X (2021) Evolving convolutional neural networks by symbiotic organisms search algorithm for image classification. Appl Soft Comput. https://doi.org/10.1016/j.asoc.2021.107537
280. Mills K, Han F, Salameh M et al (2021a) L2NAS: learning to optimize neural architectures via continuous-action reinforcement learning. In: ACM international conference on information and knowledge management, pp 1284–1293. https://doi.org/10.1145/3459637.3482360
281. Mills K, Han F, Zhang J et al (2021b) Profiling neural blocks and design spaces for mobile neural architecture search. In: ACM international conference on information and knowledge management, pp 4026–4035. https://doi.org/10.1145/3459637.3481944
282. Mills K, Salameh M, Niu D et al (2021) Exploring neural architecture search space via deep deterministic sampling. IEEE Access 9:110962–110974. https://doi.org/10.1109/ACCESS.2021.3101975
283. Miranda P, Prudêncio R, de Carvalho A et al (2014) A hybrid meta-learning architecture for multiobjective optimization of SVM parameters. Neurocomputing 143:27–43. https://doi.org/10.1016/j.neucom.2014.06.026
284. Mohammadi R, Fatemi Ghomi S, Zeinali F (2014) A new hybrid evolutionary based RBF networks method for forecasting time series: a case study of forecasting emergency supply demand time series. Eng Appl Artif Intell 36:204–214. https://doi.org/10.1016/j.engappai.2014.07.022
285. Mohr F, Wever M, Hüllermeier E (2018) ML-Plan: automated machine learning via hierarchical planning. Mach Learn 107:1495–1515. https://doi.org/10.1007/s10994-018-5735-z
286. Montecino D, Perez C, Bowyer K (2021) Two-level genetic algorithm for evolving convolutional neural networks for pattern recognition. IEEE Access 9:126856–126872. https://doi.org/10.1109/ACCESS.2021.3111175
287. Muhammad T, Halim Z (2016) Employing artificial neural networks for constructing metadata-based model to automatically select an appropriate data visualization technique. Appl Soft Comput J 49:365–384. https://doi.org/10.1016/j.asoc.2016.08.039
288. Muravev A, Raitoharju J, Gabbouj M (2021) Neural architecture search by estimation of network structure distributions. IEEE Access 9:15304–15319. https://doi.org/10.1109/ACCESS.2021.3052996

289. Mustaffa Z, Yusof Y (2014) LSSVM parameters tuning with enhanced artificial bee colony. Int Arab J Inf Technol 11:236–242
290. Nakai K, Matsubara T, Uehara K (2020) Att-DARTS: differentiable neural architecture search for attention. In: International joint conference on neural network. https://doi.org/10.1109/IJCNN48605.2020.9207447
291. Nakai K, Matsubara T, Uehara K (2021) Neural architecture search for convolutional neural networks with attention. IEICE Trans Inf Syst E104D(2):312–321. https://doi.org/10.1587/transinf.2020EDP7111
292. Nambiar V, Khalil-Hani M, Marsono M et al (2014) Optimization of structure and system latency in evolvable block-based neural networks using genetic algorithm. Neurocomputing 145:285–302. https://doi.org/10.1016/j.neucom.2014.05.033
293. Nargesian F, Samulowitz H, Khurana U et al (2017) Learning feature engineering for classification. In: International conference on artificial intelligence, pp 2529–2535. https://doi.org/10.24963/ijcai.2017/352
294. Nguyen DA, Kong J, Wang H et al (2021a) Improved automated CASH optimization with tree parzen estimators for class imbalance problems. In: IEEE international conference on data science and advanced analytics, pp 1–9. https://doi.org/10.1109/DSAA53316.2021.9564147
295. Nguyen P, Hilario M, Kalousis A (2014) Using meta-mining to support data mining workflow planning and optimization. J Artif Intell Res 51:605–644. https://doi.org/10.1613/jair.4377
296. Nguyen T, Gupta S, Rana S, et al (2017a) Stable bayesian optimization. In: Pacific-Asia Conf. on Knowl. Discov. and Data Min., pp 578–591, https://doi.org/10.1007/978-3-319-57529-2_45
297. Nguyen T, Musial K, Gabrys B (2021) AutoWeka4MCPS-AVATAR: accelerating automated machine learning pipeline composition and optimisation. Expert Syst Appl 185(115):643. https://doi.org/10.1016/j.eswa.2021.115643
298. Nguyen TD, Maszczyk T, Musial K et al (2020) AVATAR-machine learning pipeline evaluation using surrogate model. In: International conference on intelligent data analysis, pp 352–365. https://doi.org/10.1007/978-3-030-44584-3_28
299. Nguyen V, Rana S, Gupta S et al (2017b) Budgeted batch Bayesian optimization. In: IEEE international conference on data mining, pp 1107–1112. https://doi.org/10.1109/ICDM.2016.52
300. Nguyen V, Gupta S, Rana S et al (2019) Filtering Bayesian optimization approach in weakly specified search space. Knowl Inf Syst 60:385–413. https://doi.org/10.1007/s10115-018-1238-2
301. Ning X, Zheng Y, Zhao T, et al (2020) A Generic Graph-Based Neural Architecture Encoding Scheme for Predictor-Based NAS. In: Eur. Conf. on Comput. Vis., pp 189–204, https://doi.org/10.1007/978-3-030-58601-0_12
302. Ning X, Ge G, Li W, et al (2021) FTT-NAS: Discovering Fault-tolerant Convolutional Neural Architecture. ACM Trans on Des Autom of Electronic Syst 26(6). https://doi.org/10.1145/3460288
303. Niu S, Wu J, Zhang Y et al (2021) Disturbance-immune weight sharing for neural architecture search. Neural Netw 144:553–564. https://doi.org/10.1016/j.neunet.2021.09.002
304. Nomura M, Saito Y (2021) Efficient hyperparameter optimization under multi-source covariate shift. In: ACM international conference on knowledge management, pp 1376–1385. https://doi.org/10.1145/3459637.3482336
305. Noy A, Nayman N, Ridnik T et al (2020) ASAP: architecture search, anneal and prune. In: International conference on artificial intelligence and statistics, pp 493–503
306. Oh C, Gavves E, Welling M (2018) BOCK: Bayesian optimization with cylindrical kernels. In: International conference on machine learning, pp 6201–6210
307. Olson RS, Bartley N, Urbanowicz RJ, Moore JH (2016) Evaluation of a tree-based pipeline optimization tool for automating data science. In: Genetic and evolvable computing conference, pp 485–492. https://doi.org/10.1145/2908812.2908918
308. O'Neill D, Xue B, Zhang M (2020) Neural architecture search for sparse DenseNets with dynamic compression. In: Genetic and evolvable computing conference, pp 386–394. https://doi.org/10.1145/3377930.3390178
309. O'Neill D, Xue B, Zhang M (2021) Evolutionary neural architecture search for high-dimensional skip-connection structures on DenseNet style networks. IEEE Trans Evol Comput 25(6):1118–1132. https://doi.org/10.1109/TEVC.2021.3083315
310. Öztürk M, Cankaya I, Ipekçi D (2020) Optimizing echo state network through a novel fisher maximization based stochastic gradient descent. Neurocomputing 415:215–224. https://doi.org/10.1016/j.neucom.2020.07.034
311. Gijsbers P, Vanschoren J (2021) GAMA: a general automated machine learning assistant. In: European conference in machine learning and knowledge discovery in databases, pp 560–564. https://doi.org/10.1007/978-3-030-67670-4_39

312. Pan Z, Ke S, Yang X et al (2021a) AutoSTG: neural architecture search for predictions of spatio-temporal graph. In: World wide web conference, pp 1846–1855. https://doi.org/10.1145/3442381.3449816
313. Pan Z, Zeng J, Cheng R et al (2021) PNAS: a privacy preserving framework for neural architecture search services. Inf Sci 573:370–381. https://doi.org/10.1016/j.ins.2021.05.073
314. Pang D, Le X, Guan X (2021) RL-DARTS: differentiable neural architecture search via reinforcement-learning-based meta-optimizer. Knowl Syst. https://doi.org/10.1016/j.knosys.2021.107585
315. Park KM, Shin D, Chi SD (2021) Modified neural architecture search using the chromosome non-disjunction. Appl Sci (Switzerland). https://doi.org/10.3390/app11188628
316. Parmezan A, Lee H, Wu F (2017) Metalearning for choosing feature selection algorithms in data mining: proposal of a new framework. Expert Syst Appl 75:1–24. https://doi.org/10.1016/j.eswa.2017.01.013
317. Parmezan A, Lee H, Spolaôr N et al (2021) Automatic recommendation of feature selection algorithms based on dataset characteristics. Expert Syst Appl. https://doi.org/10.1016/j.eswa.2021.115589
318. Pasunuru R, Bansal M (2020) Continual and multi-task architecture search. In: Annual meeting of the association for computational linguistics, pp 1911–1922. https://doi.org/10.18653/v1/P19-1185
319. Pauletto L, Amini MR, Babbar R et al (2020) Neural architecture search for extreme multi-label text classification. In: International conference on neural information processing, pp 282–293. https://doi.org/10.1007/978-3-030-63836-8_24
320. Pedregosa F (2016) Hyperparameter optimization with approximate gradient. In: International conference on machine learning, pp 1150–1159
321. Perez-Rua J, Vielzeuf V, Pateux S et al (2019) MFAS: multimodal fusion architecture search. In: IEEE/CVF conference on computer vision and pattern recognition, pp 6959–6968. https://doi.org/10.1109/CVPR.2019.00713
322. Perrone V, Jenatton R, Seeger M et al (2018) Scalable hyperparameter transfer learning. In: Annual conference on neural information processing systems, pp 6845–6855
323. Pham H, Guan M, Zoph B et al (2018) Efficient neural architecture search via parameter sharing. In: International conference on machine learning, pp 6522–6531
324. Pimentel B, de Carvalho A (2019) A new data characterization for selecting clustering algorithms using meta-learning. Inf Sci 477:203–219. https://doi.org/10.1016/j.ins.2018.10.043
325. Pimentel B, De Carvalho A (2019) Unsupervised meta-learning for clustering algorithm recommendation. In: International joint conference on neural networks. https://doi.org/10.1109/IJCNN.2019.8851989
326. Pinto F, Soares C, Mendes-Moreira J (2016) Towards automatic generation of metafeatures. In: Pacific-Asia conference on advance in conference on knowledge discovery and data mining, pp 215–226. https://doi.org/10.1007/978-3-319-31753-3_18
327. Pourrajabi M, Zimek A, Moulavi D et al (2014) Model selection for semi-supervised clustering. In: International conference on extending database technology, pp 331–342. https://doi.org/10.5441/002/edbt.2014.31
328. Prellberg J, Kramer O (2018) Lamarckian evolution of convolutional neural networks. In: International conference on parallel problem solving from nature, pp 424–435. https://doi.org/10.1007/978-3-319-99259-4_34
329. Probst P, Boulesteix A, Bischl B (2019) Tunability: importance of hyperparameters of machine learning algorithms. J Mach Learn Res 20:1–32
330. Qiao J, Li F, Han H et al (2017) Growing echo-state network with multiple subreservoirs. IEEE Trans Neural Netw Learn Syst 28:391–404. https://doi.org/10.1109/TNNLS.2016.2514275
331. Quemy A (2020) Two-stage optimization for machine learning workflow. Inf Syst. https://doi.org/10.1016/j.is.2019.101483
332. Rakhshani H, Ismail Fawaz H, Idoumghar L et al (2020) Neural architecture search for time series classification. In: International joint conference on neural networks. https://doi.org/10.1109/IJCNN48605.2020.9206721
333. Rakotoarison H, Schoenauer M, Sebag M (2019) Automated machine learning with Monte-Carlo tree search. In: International joint conference on artificial intelligence, pp 3296–3303. https://doi.org/10.24963/ijcai.2019/457
334. Rana S, Li C, Gupta S et al (2017) High dimensional Bayesian optimization with elastic Gaussian process. In: International conference on machine learning, pp 4407–4415
335. Rasley J, He Y, Yan F et al (2017) HyperDrive: exploring hyperparameters with POP scheduling. In: ACM/IFIP/USENIX middleware Conference, pp 1–13. https://doi.org/10.1145/3135974.3135994
336. Rawat W, Wang Z (2019) Hybrid stochastic GA-Bayesian search for deep convolutional neural network model selection. J Univ Comput Sci 25:647–666. https://doi.org/10.3217/jucs-025-06-0647
337. Real E, Moore S, Selle A et al (2017) Large-scale evolution of image classifiers. In: International conference on machine learning, pp 4429–4446

338. Reif M, Shafait F, Goldstein M et al (2014) Automatic classifier selection for non-experts. Pattern Anal Appl 17:83–96. https://doi.org/10.1007/s10044-012-0280-z
339. van Rijn J, Hutter F (2018) Hyperparameter importance across datasets. In: ACM SIGKDD international conference on knowledge discovery and data mining, pp 2367–2376. https://doi.org/10.1145/3219819.3220058
340. van Rijn J, Abdulrahman S, Brazdil P et al (2015) Fast algorithm selection using learning curves. In: International symposium on intelligent data analysis, pp 298–309. https://doi.org/10.1007/978-3-319-24465-5_26
341. Ristoski P (2015) Towards linked open data enabled data mining: strategies for feature generation, propositionalization, selection, and consolidation. In: European Semantic Web Conference, pp 772–782. https://doi.org/10.1007/978-3-319-18818-8_50
342. Rosales-Pérez A, Gonzalez J, Coello Coello C et al (2014) Multi-objective model type selection. Neurocomputing 146:83–94. https://doi.org/10.1016/j.neucom.2014.05.077
343. Rosales-Pérez A, Gonzalez J, Coello Coello C et al (2015) Surrogate-assisted multi-objective model selection for support vector machines. Neurocomputing 150:163–172. https://doi.org/10.1016/j.neucom.2014.08.075
344. Rossi A, de Carvalho A, Soares C et al (2014) MetaStream: a meta-learning based method for periodic algorithm selection in time-changing data. Neurocomputing 127:52–64. https://doi.org/10.1016/j.neucom.2013.05.048
345. de Sá A, Freitas A, Pappa G (2018) Automated selection and configuration of multi-label classification algorithms with grammar-based genetic programming. In: International conference on parallel problem solving from nature, pp 308–320. https://doi.org/10.1007/978-3-319-99259-4_25
346. Sabharwal A, Samulowitz H, Tesauro G (2016) Selecting near-optimal learners via incremental data allocation. In: AAAI conferences on artificial intelligence, pp 2007–2015. https://doi.org/10.1609/aaai.v30i1.10316
347. Sanders S, Giraud-Carrier C (2017) Informing the use of hyperparameter optimization through metalearning. In: IEEE international conference on data mining, pp 1051–1056. https://doi.org/10.1109/ICDM.2017.137
348. Sapra D, Pimentel A (2021) Designing convolutional neural networks with constrained evolutionary piecemeal training. Appl Intell. https://doi.org/10.1007/s10489-021-02679-7
349. Saxena S, Verbeek J (2016) Convolutional neural fabrics. In: Annual conference on neural networks information processing system, pp 4060–4068
350. Schilling N, Wistuba M, Drumond L et al (2015) Hyperparameter optimization with factorized multilayer perceptrons. In: European conference in machine learning and knowledge discovery in databases, pp 87–103. https://doi.org/10.1007/978-3-319-23525-7_6
351. Schmidt M, Safarani S, Gastinger J et al (2019) On the performance of differential evolution for hyperparameter tuning. In: International joint conference on neural networks. https://doi.org/10.1109/IJCNN.2019.8851978
352. Schmidt M, Gastinger J, Nicolas S et al (2020) HAMLET—a learning curve-enabled multi-armed bandit for algorithm selection. In: International joint conference on neural networks. https://doi.org/10.1109/IJCNN48605.2020.9207233
353. Sen R, Kandasamy K, Shakkottai S (2018) Multi-fidelity black-box optimization with hierarchical partitions. In: International conference on machine learning, pp 7212–7224
354. Shah A, Ghahramani Z (2015) Parallel predictive entropy search for batch global optimization of expensive objective functions. In: Annual conference on neural information processing systems, pp 3330–3338
355. Shahriari B, Bouchard-Côté A, de Freitas N (2016) Unbounded bayesian optimization via regularization. In: International conference on artificial intelligence and statistics, pp 1168–1176
356. Shang Z, Zgraggen E, Buratti B et al (2019) Democratizing data science through interactive curation of ML pipelines. In: International conference on management of data, pp 1171–1188. https://doi.org/10.1145/3299869.3319863
357. da Silva A, de Oliveira W, Ludermir T (2016) Weightless neural network parameters and architecture selection in a quantum computer. Neurocomputing 183:13–22. https://doi.org/10.1016/j.neucom.2015.05.139
358. Singh L, Paul S (2021) Hybrid evolutionary network architecture search for convolution class of deep neural networks with applications. Expert Syst. https://doi.org/10.1111/exsy.12690
359. Singh P, Chaudhury S, Panigrahi B (2021) Hybrid MPSO-CNN: multi-level particle swarm optimized hyperparameters of convolutional neural network. Swarm Evol Comput. https://doi.org/10.1016/j.swevo.2021.100863

360. Sinha N, Chen KW (2021) Evolving neural architecture using one shot model. In: Genetic and evolvable computing conference, pp 910–918. https://doi.org/10.1145/3449639.3459275
361. Smith M, Sala C, Kanter J et al (2020) The machine learning bazaar: harnessing the ML ecosystem for effective system development. In: ACM SIGMOD international conference on management of data, pp 785–800. https://doi.org/10.1145/3318464.3386146
362. Smithson S, Yang G, Gross W et al (2016) Neural networks designing neural networks: multi-objective hyper-parameter optimization. In: IEEE/ACM international conference on computer design. https://doi.org/10.1145/2966986.2967058
363. Snoek J, Swersky K, Zemel R et al (2014) Input warping for Bayesian optimization of non-stationary functions. In: International conference on machine learning, pp 3654–3662
364. Snoek J, Ripped O, Swersky K et al (2015) Scalable Bayesian optimization using deep neural networks. In: International conference on machine learning, pp 2161–2170
365. Song H (2021) A method for gradient differentiable network architecture search by selecting and clustering candidate operations. Appl Sci (Switzerland). https://doi.org/10.3390/app112311436
366. Soniya Paul S, Singh L (2020) Application and need-based architecture design of deep neural networks. Int J Pattern Recognit Artif Intell. https://doi.org/10.1142/S021800142052014X
367. Springenberg J, Klein A, Falkner S et al (2016) Bayesian optimization with Robust Bayesian neural networks. In: Annual conference on neural information processing systems, pp 4141–4149
368. Stamoulis D, Ding R, Wang D et al (2020) Single-path NAS: designing hardware-efficient ConvNets in less than 4 hours. In: European conference in machine learning and knowledge discovery in databases, pp 481–497. https://doi.org/10.1007/978-3-030-46147-8_29
369. Stein B, Wang H, Back T (2019) Automatic configuration of deep neural networks with parallel efficient global optimization. In: International joint conference on neural network. https://doi.org/10.1109/IJCNN.2019.8851720
370. Stojanovic B, Milivojevic M, Milivojevic N et al (2016) A self-tuning system for dam behavior modeling based on evolving artificial neural networks. Adv Eng Softw 97:85–95. https://doi.org/10.1016/j.advengsoft.2016.02.010
371. Suganuma M, Shirakawa S, Nagao T (2017) A genetic programming approach to designing convolutional neural network architectures. In: Genetic and evolvable computing conference, pp 497–504. https://doi.org/10.1145/3071178.3071229
372. Suganuma M, Kobayashi M, Shirakawa S et al (2020) Evolution of deep convolutional neural networks using cartesian genetic programming. Evol Comput 28:141–163. https://doi.org/10.1162/evco_a_00253
373. Sui G, Yu Y (2020) Bayesian contextual bandits for hyper parameter optimization. IEEE Access 8:42971–42979. https://doi.org/10.1109/ACCESS.2020.2977129
374. Sun M, Dou H, Yan J (2020a) Efficient transfer learning via joint adaptation of network architecture and weight. In: European conference on computer vision, pp 463–480. https://doi.org/10.1007/978-3-030-58601-0_28
375. Sun X, Lin J, Bischl B (2020b) ReinBo: machine learning pipeline conditional hierarchy search and configuration with bayesian optimization embedded reinforcement learning. In: Joint European conference on machine learning and principle and problem of knowledge discovery in databases, pp 68–84. https://doi.org/10.1007/978-3-030-43823-4_7
376. Sun Y, Gong H, Li Y et al (2019) Hyperparameter importance analysis based on N-rrelieff algorithm. Int J Comput Commun Control 14:557–573. https://doi.org/10.15837/ijccc.2019.4.3593
377. Sun Y, Xue B, Zhang M et al (2019) A particle swarm optimization-based flexible convolutional autoencoder for image classification. IEEE Trans Neural Netw Learn Syst 30:2295–2309. https://doi.org/10.1109/TNNLS.2018.2881143
378. Sun Y, Xue B, Zhang M et al (2020) Completely automated CNN architecture design based on blocks. IEEE Trans Neural Netw Learn Syst 31:1242–1254. https://doi.org/10.1109/TNNLS.2019.2919608
379. Sun Y, Xue B, Zhang M et al (2020) Automatically designing CNN architectures using the genetic algorithm for image classification. IEEE Trans Cybern 50:3840–3854. https://doi.org/10.1109/TCYB.2020.2983860
380. Sun Z, Hu Y, Yang L et al (2021) STC-NAS: fast neural architecture search with source-target consistency. Neurocomputing. https://doi.org/10.1016/j.neucom.2021.11.082
381. Szwarcman D, Civitarese D, Vellasco M (2019) Quantum-inspired neural architecture search. In: International joint conference on neural network. https://doi.org/10.1109/IJCNN.2019.8852453
382. Tan H, Cheng R, Huang S et al (2021) RelativeNAS: relative neural architecture search via slow–fast learning. IEEE Trans Neural Netw Learn Syst. https://doi.org/10.1109/TNNLS.2021.3096658
383. Tang Y, Wang Y, Xu Y et al (2020) A semi-supervised assessor of neural architectures. In: IEEE/CVF conference on computer vision and pattern recognition, pp 1807–1816. https://doi.org/10.1109/CVPR42600.2020.00188

384. Termritthikun C, Jamtsho Y, Ieamsaard J et al (2021) EEEA-Net: an early exit evolutionary neural architecture search. Eng Appl Artif Intell. https://doi.org/10.1016/j.engappai.2021.104397
385. Theckel Joy T, Rana S, Gupta S et al (2019) A flexible transfer learning framework for Bayesian optimization with convergence guarantee. Expert Syst Appl 115:656–672. https://doi.org/10.1016/j.eswa.2018.08.023
386. Thiede L, Parlitz U (2019) Gradient based hyperparameter optimization in echo state networks. Neural Netw 115:23–29. https://doi.org/10.1016/j.neunet.2019.02.001
387. Tian Y, Liu C, Xie L et al (2021) Discretization-aware architecture search. Pattern Recognit. https://doi.org/10.1016/j.patcog.2021.108186
388. Tian Y, Shen L, Shen L et al (2021) AlphaGAN: fully differentiable architecture search for generative adversarial networks. IEEE Trans Pattern Anal Mach Intell. https://doi.org/10.1109/TPAMI.2021.3099829
389. Tiddi I, D'Aquin M, Motta E (2014) Dedalo: looking for clusters explanations in a labyrinth of linked data. In: International conference in the semantic web: trends and challenges, pp 333–348. https://doi.org/10.1007/978-3-319-07443-6_23
390. Tran B, Xue B, Zhang M (2016) Genetic programming for feature construction and selection in classification on high-dimensional data. Memetic Comput 8:3–15. https://doi.org/10.1007/s12293-015-0173-y
391. Tran LT, Ali M, Bae SH (2021) A feature fusion based indicator for training-free neural architecture search. IEEE Access 9:133914–133923. https://doi.org/10.1109/ACCESS.2021.3115911
392. Tran N, Schneider JG, Weber I et al (2020) Hyper-parameter optimization in classification: to-do or not-to-do. Pattern Recognit. https://doi.org/10.1016/j.patcog.2020.107245
393. Tu K, Ma J, Cui P et al (2019) Autone: hyperparameter optimization for massive network embedding. In: ACM SIGKDD international conference on knowledge discovery and data mining, pp 216–225. https://doi.org/10.1145/3292500.3330848
394. Turner J, Crowley E, O'Boyle M (2021) Neural architecture search as program transformation exploration. In: International conference on arch support for programming languages and operating systems, pp 915–927. https://doi.org/10.1145/3445814.3446753
395. Ukil A, Sahu I, Puri C et al (2018) AutoModeling: integrated approach for automated model generation by ensemble selection of feature subset and classifier. In: International joint conference on neural network. https://doi.org/10.1109/IJCNN.2018.8489730
396. Vahdat A, Mallya A, Liu MY et al (2020) UNAS: differentiable architecture search meets reinforcement learning. In: IEEE/CVF conference on computer vision and pattern recognition, pp 11263–11272. https://doi.org/10.1109/CVPR42600.2020.01128
397. Vainshtein R, Greenstein-Messica A, Katz G et al (2018) A hybrid approach for automatic model recommendation. In: ACM international conference on information and knowledge management, pp 1623–1626. https://doi.org/10.1145/3269206.3269299
398. Van Craenendonck T, Blockeel H (2017) Constraint-based clustering selection. Mach Learn 106:1497–1521. https://doi.org/10.1007/s10994-017-5643-7
399. Veniat T, Denoyer L (2018) Learning time/memory-efficient deep architectures with budgeted super networks. In: IEEE/CVF conference on computer vision and pattern recognition, pp 3492–3500. https://doi.org/10.1109/CVPR.2018.00368
400. Vidnerová P, Neruda R (2020) Multi-objective evolution for deep neural network architecture search. In: International conference on neural information processing, pp 270–281. https://doi.org/10.1007/978-3-030-63836-8_23
401. Vijaya Saradhi V, Girish K (2014) Effective parameter tuning of SVMs using radius/margin bound through data envelopment analysis. Neural Process Lett 41:125–138. https://doi.org/10.1007/s11063-014-9338-9
402. Völcker C, Molina A, Neumann J et al (2020) DeepNotebooks: deep probabilistic models construct python notebooks for reporting datasets. In: Joint European conference on machine learning and principles and practice of knowledge discovery in databases, pp 28–43. https://doi.org/10.1007/978-3-030-43823-4_3
403. Wan A, Dai X, Zhang P et al (2020) FBNetV2: differentiable neural architecture search for spatial and channel dimensions. In: IEEE/CVF conference on computer vision and pattern recognition, pp 12962–12971. https://doi.org/10.1109/CVPR42600.2020.01298
404. Wang B, Sun Y, Xue B et al (2019a) Evolving deep neural networks by multi-objective particle swarm optimization for image classification. In: Genetic and evolvable computing conference, pp 490–498. https://doi.org/10.1145/3321707.3321735

405. Wang B, Xue B, Zhang M (2021) Surrogate-assisted particle swarm optimization for evolving variable-length transferable blocks for image classification. IEEE Trans Neural Netw Learn Syst. https://doi.org/10.1109/TNNLS.2021.3054400
406. Wang C, Wang H, Mu T et al (2020a) Auto-model: utilizing research papers and HPO techniques to deal with the CASH problem. In: IEEE international conference on data engineering, pp 1906–1909. https://doi.org/10.1109/ICDE48307.2020.00200
407. Wang C, Wang H, Zhou C et al (2021) Experience thinking: constrained hyperparameter optimization based on knowledge and pruning. Knowl Syst. https://doi.org/10.1016/j.knosys.2020.106602
408. Wang D, Andres J, Weisz J (2021c) Autods: towards human-centered automation of data science. In: Conference on human factors in computing systems. https://doi.org/10.1145/3411764.3445526
409. Wang D, Li M, Gong C et al (2021d) AttentiveNAS: improving neural architecture search via attentive sampling. In: IEEE/CVF conference on computer vision and pattern recognition, pp 6414–6423. https://doi.org/10.1109/CVPR46437.2021.00635
410. Wang E, Xu S, Chen CM et al (2021) Neural-architecture-search-based multiobjective cognitive automation system. IEEE Syst J 15(2):2918–2925. https://doi.org/10.1109/JSYST.2020.3002428
411. Wang G, Song Q, Zhang X et al (2014) A generic multilabel learning-based classification algorithm recommendation method. ACM Trans Knowl Discov Data. https://doi.org/10.1145/2629474
412. Wang G, Ben Sassi M, Grosu R (2017) ZIZO: a novel zoom-in-zoom-out search algorithm for the global parameters of echo-state networks. Can J Electr Comput Eng 40:210–216. https://doi.org/10.1109/CJECE.2017.2703093
413. Wang H, Yang R, Huang D et al (2021) iDARTS: improving DARTS by node normalization and decorrelation discretization. IEEE Trans Neural Netw Learn Syst. https://doi.org/10.1109/TNNLS.2021.3105698
414. Wang L, Feng M, Zhou B et al (2015) Efficient hyper-parameter optimization for NLP applications. In: Conference on empirical methods in natural language processing, pp 2112–2117. https://doi.org/10.18653/v1/d15-1253
415. Wang L, Xie S, Li T et al (2021) Sample-efficient neural architecture search by learning actions for Monte Carlo tree search. IEEE Trans Pattern Anal Mach Intell. https://doi.org/10.1109/TPAMI.2021.3071343
416. Wang Q, Ming Y, Jin Z et al (2019b) AtmSeer: increasing transparency and controllability in automated machine learning. In: CHI conference on human factors in computer system. https://doi.org/10.1145/3290605.3300911
417. Wang X, Xue C, Yan J et al (2020b) MergeNAS: merge operations into one for differentiable architecture search. In: International joint conferences on artificial intelligence, pp 3065–3072. https://doi.org/10.24963/ijcai.2020/424
418. Wang Z, Jegelka S (2017) Max-value entropy search for efficient Bayesian optimization. In: International conference on machine learning, pp 5530–5543
419. Wang Z, Hutter F, Zoghi M et al (2016) Bayesian optimization in a billion dimensions via random embeddings. J Artif Intell Res 55:361–367. https://doi.org/10.1613/jair.4806
420. Wang Z, Zhou B, Jegelka S (2016b) Optimization as estimation with Gaussian processes in bandit settings. In: International conference on artificial intelligence and statistics, pp 1022–1031
421. Wang Z, Lu D, Wang H et al (2021) Evolutionary convolutional neural network optimization with cross-tasks transfer strategy. Electronics (Switzerland). https://doi.org/10.3390/electronics10151857
422. Weidele D, Weisz J, Oduor E et al (2020) AutoAIViz: opening the blackbox of automated artificial intelligence with conditional parallel coordinates. In: ACM international conference on intelligent user interfaces, pp 308–312. https://doi.org/10.1145/3377325.3377538
423. Welchowski T, Schmid M (2016) A framework for parameter estimation and model selection in kernel deep stacking networks. Artif Intell Med 70:31–40. https://doi.org/10.1016/j.artmed.2016.04.002
424. Wen W, Liu H, Chen Y et al (2020a) Neural predictor for neural architecture search. In: European conference on computer vision, pp 660–676. https://doi.org/10.1007/978-3-030-58526-6_39
425. Wen W, Yan F, Chen Y, et al (2020b) AutoGrow: automatic layer growing in deep convolutional networks. In: ACM SIGKDD international conference on knowledge discovery and data mining, pp 833–841. https://doi.org/10.1145/3394486.3403126
426. Wen YW, Peng SH, Ting CK (2021) Two-stage evolutionary neural architecture search for transfer learning. IEEE Trans Evol Comput 25(5):928–940. https://doi.org/10.1109/TEVC.2021.3097937
427. Weng Y, Zhou T, Liu L et al (2019) Automatic convolutional neural architecture search for image classification under different scenes. IEEE Access 7:38495–38506. https://doi.org/10.1109/ACCESS.2019.2906369

428. Werneck R, de Almeida W, Stein B et al (2018) Kuaa: a unified framework for design, deployment, execution, and recommendation of machine learning experiments. Futur Gener Comput Syst 78:59–76. https://doi.org/10.1016/j.future.2017.06.013
429. Wistuba M (2019) Deep learning architecture search by neuro-cell-based evolution with function-preserving mutations. In: European conference on machine learning and principles and practice of knowledge discovery in databases, pp 243–258. https://doi.org/10.1007/978-3-030-10928-8_15
430. Wistuba M (2021) XferNAS: transfer neural architecture search. In: European conference in machine learning and knowledge discovery in databases, pp 247–262. https://doi.org/10.1007/978-3-030-67664-3_15
431. Wistuba M, Schilling N, Schmidt-Thieme L (2015) Hyperparameter search space pruning—a new component for sequential model-based hyperparameter optimization. In: European conference on machine learning and principles and practice of knowledge discovery in databases, pp 104–119. https://doi.org/10.1007/978-3-319-23525-7_7
432. Wistuba M, Schilling N, Schmidt-Thieme L (2016) Sequential model-free hyperparameter tuning. In: IEEE international conference on data mining, pp 1033–1038. https://doi.org/10.1109/ICDM.2015.20
433. Wistuba M, Schilling N, Schmidt-Thieme L (2017) Automatic frankensteining: creating complex ensembles autonomously. In: SIAM international conference on data mining, pp 741–749. https://doi.org/10.1137/1.9781611974973.83
434. Wistuba M, Schilling N, Schmidt-Thieme L (2018) Scalable Gaussian process-based transfer surrogates for hyperparameter optimization. Mach Learn 107:43–78. https://doi.org/10.1007/s10994-017-5684-y
435. Wu B, Dai X, Zhang P et al (2019) FBNet: hardware-aware efficient ConvNet design via differentiable neural architecture search. In: IEEE/CVF conference on computer vision and pattern recognition, pp 10726–10734. https://doi.org/10.1109/CVPR.2019.01099
436. Wu J, Frazier P (2016) The parallel knowledge gradient method for batch Bayesian optimization. In: Annual conference on neural information processing systems, pp 3134–3142
437. Wu J, Chen S, Liu X (2020) Efficient hyperparameter optimization through model-based reinforcement learning. Neurocomputing 409:381–393. https://doi.org/10.1016/j.neucom.2020.06.064
438. Xie H, Zhang L, Lim C (2020) Evolving CNN-LSTM models for time series prediction using enhanced grey wolf optimizer. IEEE Access 8:161519–161541. https://doi.org/10.1109/ACCESS.2020.3021527
439. Xie L, Yuille A (2017) Genetic CNN. In: IEEE international conference on computer vision, pp 1388–1397. https://doi.org/10.1109/ICCV.2017.154
440. Xu Y, Wang Y, Han K et al (2021a) ReNAS: relativistic evaluation of neural architecture search. In: IEEE/CVF conference on computer vision and pattern recognition, pp 4409–4418. https://doi.org/10.1109/CVPR46437.2021.00439
441. Xu Y, Xie L, Dai W et al (2021) Partially-connected neural architecture search for reduced computational redundancy. IEEE Trans Pattern Anal Mach Intell 43(9):2953–2970. https://doi.org/10.1109/TPAMI.2021.3059510
442. Xue C, Yan J, Yan R et al (2019) Transferable AutoML by model sharing over grouped datasets. In: IEEE/CVF conference on computer vision and pattern recognition, pp 8994–9003. https://doi.org/10.1109/CVPR.2019.00921
443. Xue S, Chen H, Xie C et al (2021) Fast and unsupervised neural architecture evolution for visual representation learning. IEEE Comput Intell Mag 16(3):22–32. https://doi.org/10.1109/MCI.2021.3084394
444. Xue Y, Jiang P, Neri F et al (2021) A multi-objective evolutionary approach based on graph-in-graph for neural architecture search of convolutional neural networks. Int J Neural Syst. https://doi.org/10.1142/S0129065721500350
445. Yakovlev A, Moghadam HF, Moharrer A et al (2020) Oracle AutoML: a fast and predictive AutoML pipeline. In: International conference on very large data bases, pp 3166–3180. https://doi.org/10.14778/3415478.3415542
446. Yan C, Chang X, Li Z et al (2021) ZeroNAS: differentiable generative adversarial networks search for zero-shot learning. IEEE Trans Pattern Anal Mach Intell. https://doi.org/10.1109/TPAMI.2021.3127346
447. Yan Z, Dai X, Zhang P et al (2021b) FP-NAS: fast probabilistic neural architecture search. In: IEEE/CVF conference on computer vision and pattern recognition, pp 15134–15143. https://doi.org/10.1109/CVPR46437.2021.01489
448. Yang TJ, Liao YL, Sze V (2021a) NetAdaptV2: efficient neural architecture search with fast super-network training and architecture optimization. In: IEEE/CVF conference on computer vision and pattern recognition, pp 2402–2411. https://doi.org/10.1109/CVPR46437.2021.00243
449. Yang Y, You S, Li H et al (2021b) Towards improving the consistency, efficiency, and flexibility of differentiable neural architecture search. In: IEEE/CVF conference on computer vision and pattern recognition, pp 6663–6672. https://doi.org/10.1109/CVPR46437.2021.00660

450. Yang Z, Zhang A (2021) Hyperparameter optimization via sequential uniform designs. J Mach Learn Res 22:6592–6638
451. Yang Z, Wang Y, Chen X et al (2020) CARS: continuous evolution for efficient neural architecture search. In: IEEE/CVF conference on computer vision and pattern recognition, pp 1826–1835. https://doi.org/10.1109/CVPR42600.2020.00190
452. Yang Z, Wang Y, Chen X et al (2021c) HourNAS: extremely fast neural architecture search through an hourglass lens. In: IEEE/CVF conference on computer vision and pattern recognition, pp 10891–10901. https://doi.org/10.1109/CVPR46437.2021.01075
453. Yao Y, Cao J, Ma Z (2018) A cost-effective deadline-constrained scheduling strategy for a hyperparameter optimization workflow for machine learning algorithms. In: International conference on system computing, pp 870–878. https://doi.org/10.1007/978-3-030-03596-9_62
454. Ye Q, Sun Y, Zhang J et al (2021) A distributed framework for EA-based NAS. IEEE Trans Parallel Distrib Syst 32(7):1753–1764. https://doi.org/10.1109/TPDS.2020.3046774
455. Yogatama D, Mann G (2014) Efficient transfer learning method for automatic hyperparameter tuning. In: International conference on artificial intelligence and statistics, pp 1077–1085
456. You S, Huang T, Yang M et al (2020) Greedynas: towards fast one-shot NAS with greedy supernet. In: IEEE/CVF conference on computer vision and pattern recognition, pp 1996–2005. https://doi.org/10.1109/CVPR42600.2020.00207
457. Yu Y, Qian H, Hu YQ (2016) Derivative-free optimization via classification. In: AAAI conference on artificial intelligence, pp 2286–2292
458. Zainel Q, Khorsheed M, Darwish S et al (2021) An optimized convolutional neural network architecture based on evolutionary ensemble learning. Comput Mater Contin 69(3):3813–3828. https://doi.org/10.32604/cmc.2021.014759
459. Zhang B, Zhou G (2021) Control the number of skip-connects to improve robustness of the NAS algorithm. IET Comput Vis 15(5):356–365. https://doi.org/10.1049/cvi2.12036
460. Zhang H, Jin Y, Cheng R et al (2021) Efficient evolutionary search of attention convolutional networks via sampled training and node inheritance. IEEE Trans Evol Comput 25(2):371–385. https://doi.org/10.1109/TEVC.2020.3040272
461. Zhang J, Fogelman-Soulié F, Largeron C (2018a) Towards automatic complex feature engineering. In: International conference on web information system engineering, pp 312–322. https://doi.org/10.1007/978-3-030-02925-8_22
462. Zhang M, Li H, Pan S et al (2020) One-shot neural architecture search via novelty driven sampling. In: International joint conference on artificial intelligence, pp 3188–3194. https://doi.org/10.24963/ijcai.2020/441
463. Zhang M, Li H, Pan S et al (2021) One-shot neural architecture search: maximising diversity to overcome catastrophic forgetting. IEEE Trans Pattern Anal Mach Intell 43(9):2921–2935. https://doi.org/10.1109/TPAMI.2020.3035351
464. Zhang T, Georgiopoulos M, Anagnostopoulos G (2016) Multi-objective model selection via racing. IEEE Trans Cybern 46:1863–1876. https://doi.org/10.1109/TCYB.2015.2456187
465. Zhang T, Lei C, Zhang Z et al (2021) AS-NAS: adaptive scalable neural architecture search with reinforced evolutionary algorithm for deep learning. IEEE Trans Evol Comput 25(5):830–841. https://doi.org/10.1109/TEVC.2021.3061466
466. Zhang T, Waqas M, Shen H et al (2021) A neural network architecture optimizer based on DARTS and generative adversarial learning. Inf Sci 581:448–468. https://doi.org/10.1016/j.ins.2021.09.041
467. Zhang X, Chen X, Yao L et al (2019a) Deep neural network hyperparameter optimization with orthogonal array tuning. In: International conference on neural information processing, pp 287—-295. https://doi.org/10.1007/978-3-030-36808-1_31
468. Zhang X, Hou P, Zhang X et al (2021e) Neural architecture search with random labels. In: IEEE/CVF conference on computer vision and pattern recognition, pp 10902–10911. https://doi.org/10.1109/CVPR46437.2021.01076
469. Zhang X, Huang Z, Wang N et al (2021) You only search once: single shot neural architecture search via direct sparse optimization. IEEE Trans Pattern Anal Mach Intell 43(9):2891–2904. https://doi.org/10.1109/TPAMI.2020.3020300
470. Zhang Y, Dai Z, Low B (2019b) Bayesian optimization with binary auxiliary information. In: Conference on uncertainty in artificial intelligence
471. Zhang Z, Han J, Qian K et al (2018b) Evolving learning for analysing mood-related infant vocalisation. In: Annual international speech communication, pp 142–146. https://doi.org/10.21437/Interspeech.2018-1914

472. Zhao Z, Kang Y, Hou A et al (2021) SP-DARTS: synchronous progressive differentiable neural architecture search for image classification. IEICE Trans Inf Syst E104D(8):1232–1238. https://doi.org/10.1587/transinf.2020BDP0009
473. Zheng X, Zhang Y, Hong S et al (2021) Evolving fully automated machine learning via life-long knowledge anchors. IEEE Trans Pattern Anal Mach Intell 43(9):3091–3107. https://doi.org/10.1109/TPAMI.2021.3069250
474. Zhong G, Jiao W, Gao W et al (2020) Automatic design of deep networks with neural blocks. Cogn Comput. https://doi.org/10.1007/s12559-019-09677-5
475. Zhong S, Xie X, Lin L et al (2017) Genetic algorithm optimized double-reservoir echo state network for multi-regime time series prediction. Neurocomputing 238:191–204. https://doi.org/10.1016/j.neucom.2017.01.053
476. Zhong Z, Yan J, Wu W et al (2018) Practical block-wise neural network architecture generation. In: IEEE/CVF conference on computer vision and pattern recognition, pp 2423–2432. https://doi.org/10.1109/CVPR.2018.00257
477. Zhong Z, Yang Z, Deng B et al (2021) BlockQNN: efficient block-wise neural network architecture generation. IEEE Trans Pattern Anal Mach Intell 43(7):2314–2328. https://doi.org/10.1109/TPAMI.2020.2969193
478. Zhou D, Zhou X, Zhang W et al (2020a) EcoNAS: finding proxies for economical neural architecture search. In: IEEE/CVF conference on computer vision and pattern recognition, pp 11393–11401. https://doi.org/10.1109/CVPR42600.2020.01141
479. Zhou Y, Jin Y, Ding J (2020) Surrogate-assisted evolutionary search of spiking neural architectures in liquid state machines. Neurocomputing 406:12–23. https://doi.org/10.1016/j.neucom.2020.04.079
480. Zhu B, Al-Ars Z, Hofstee H (2020) NASB: neural architecture search for binary convolutional neural networks. In: International joint conference on neural network. https://doi.org/10.1109/IJCNN48605.2020.9207674
481. Zhuo L, Zhang B, Chen H et al (2020) CP-NAS: child–parent neural architecture search for 1-bit CNNs. In: International joint conference on artificial intelligence, pp 1033–1039. https://doi.org/10.24963/ijcai.2020/144
482. Zimmer L, Lindauer M, Hutter F (2021) Auto-Pytorch: multi-fidelity MetaLearning for efficient and robust AutoDL. IEEE Trans Pattern Anal Mach Intell 43(9):3079–3090. https://doi.org/10.1109/TPAMI.2021.3067763
483. Zoph B, Vasudevan V, Shlens J et al (2018) Learning transferable architectures for scalable image recognition. In: IEEE/CVF conference on computer vision and pattern recognition, pp 8697–8710. https://doi.org/10.1109/CVPR.2018.00907
484. Zutty J, Long D, Adams H et al (2015) Multiple objective vector-based genetic programming using human-derived primitives. In: Genetic and evolvable computing conference, pp 1127–1134. https://doi.org/10.1145/2739480.2754694

Rafael Barbudo is a Ph.D. Student of Computer Science at the University of Córdoba, Spain. He received both his B.S. and M.S. in Information and Computer Science at the University of Córdoba. His current research is focused on the area of automated machine learning (AutoML). Specifically, it is dedicated to the development of tools that assist both domain experts and data scientists in the process of knowledge extraction by optimizing and automating those repetitive and time-consuming phases.

Sebastián Ventura is a Full Professor in the Department of Computer Science and Numerical Analysis at the University of Córdoba, where he leads the Knowledge Discovery and Intelligent Systems Research Laboratory. He has published three books and about 300 papers in journals and scientific conferences and edited about ten books and multiple special issues in international journals in his area of expertise. He currently holds different positions at the editorial board of multiple top-ranked journals and serves as Editor in Chief at the Progress in Artificial Intelligence journal. His main research interests are in the fields of data science, computational intelligence, and their applications.

José Raúl Romero is an Associate Professor of Computer Science at the University of Córdoba, Spain. He received his PhD from the University of Málaga (Spain). He has published more than 110 papers in journals and scientific conferences and edited one volume in his area of expertise. His research interests focus on democratization of data science, particularly on the development of new methods for the automation of machine learning, human-centered machine learning, and explicability of artificial intelligence. He is an active researcher on artificial intelligence for software engineering.