Evolutionary Computation and Explainable AI: A Roadmap to Transparent Intelligent Systems

Anonymous Authors

***Abstract*—AI methods are finding an increasing number of applications, but their often black-box nature has raised concerns about accountability and trust. The field of explainable artificial intelligence (XAI) has emerged in response to the need for human understanding of AI models. Evolutionary computation (EC), as a family of powerful optimization and learning tools, has significant potential to contribute to XAI. In this paper, we provide an introduction to XAI and review various techniques in current use for explaining machine learning (ML) models. We then focus on how EC can be used in XAI, and review some XAI approaches which incorporate EC techniques. Additionally, we discuss the application of XAI principles within EC itself, examining how these principles can shed some light on the behavior and outcomes of EC algorithms in general, on the (automatic) configuration of these algorithms, and on the underlying problem landscapes that these algorithms optimize. Finally, we discuss some open challenges in XAI and opportunities for future research in this field using EC. Our aim is to demonstrate that EC is well-suited for addressing current problems in explainability and to encourage further exploration of these methods to contribute to the development of more transparent and trustworthy ML models and EC algorithms.**

***Index Terms*—Explainability, Interpretability, Evolutionary Computation, Machine Learning.**

I. Introduction

The use of AI has become increasingly widespread, and with it, there is a growing need to understand the reasoning behind the outputs and decisions it produces. Although AI methods can learn complex relationships in data or provide solutions to challenging problems, decisions based on the outputs of models can have real-world impacts. For example, the use of predictive models in applications such as medicine, hiring, and the justice system has raised concerns about the fairness and transparency of such models; the increasing use of large language models in commercial products has made avoiding harmful content ever more important; and the adoption of black-box optimization in areas such as scheduling and logistics [1] requires the trust of the users who are accountable if things go wrong. Therefore, it is essential not only to improve the models we create but also to understand and explain what led to their decisions. There are active lines of research into improving the fairness and safety of models, but in this survey, we focus on the latter problem: understanding, explaining, and increasing transparency in AI systems.

Recent advances in AI have drawn heavily on “black-box” approaches. Deep learning, ensemble models, and stochastic optimization algorithms may have clearly defined structures, but the processes leading to the decisions they make are often sufficiently complex to be opaque to humans. The field of explainable artificial intelligence (XAI) has emerged in response to this need [2]. XAI is an umbrella term that covers research on methods designed to provide human-understandable explanations of the decisions made/knowledge captured by AI models.

XAI research aims to develop methods to explain the decisions, predictions, or recommendations made by AI processes in terms that humans can understand. These explanations foster trust and improve a system’s robustness by highlighting potential biases and failures. They also provide researchers with insights to better understand, validate, and debug the system effectively. Beyond this, they also play a pivotal role in ensuring regulatory compliance and improving human-machine interactions, allowing users a better understanding of when they can rely on a model’s conclusions.

In the context of evolutionary computation (EC), in our view, two directions associated with XAI emerge. First, the application of XAI principles to decision-making within EC, and second, the use of EC to enhance explainability within ML. A body of work is developing in both areas and is gathering pace – in part due to events such as a workshop on EC and XAI held at GECCO in 2022 and 2023. The aim of this paper is to provide a critical review of research conducted at the intersection of EC and XAI. We provide a taxonomy of methods and highlight potential avenues of future work, expanding on the initial directions proposed in [3] and [4].

The remainder of this paper provides a discussion around these themes. First, in Section II, we introduce foundational concepts in XAI such as the nature of explanations and the distinctions between interpretability and explainability, and provide motivation for strengthening the link between XAI and EC. Then, in Section III, we discuss how EC can be used *for* XAI. while in Section IV we discuss how XAI can be *applied to* EC. We then discuss the ongoing challenges and potential opportunities in Section V. Lastly, we provide final thoughts and conclusions in Section VI.

II. Explainable AI

At its core, XAI aims to provide methods and tools for humans to understand the decision-making processes of AI systems. These tools provide insights in the form of explanations, shedding light on how such systems produce their outputs and solutions, highlighting significant features and interactions that influence the results, and revealing potential

Manuscript received June 8, 2024.

0000–0000/00$00.00 © 2024 IEEE

issues in their workings. Even though such a system may be too complex for a human to interpret directly, it can be considered explainable if it can be understood.

Explainability is important for several reasons. Perhaps the most crucial is **trust**. Trust in the workings of an AI algorithm directly influences users' willingness to adopt and adhere to the results and ensures that users not only accept but also can confidently and justifiably rely on the answers these models provide. In the case of optimization, this may mean convincing users that they can trust the solutions by knowing *what* makes that solution better than anything (or at least something) else, which might be seen as synonymous with knowing *why* the solution was chosen. In the case of a model created by ML, it may mean allowing users to understand the decisions that a model makes leading to the final output. It is also important to consider that such an explanation will likely be important in the future to provide an audit trail for the decisions underpinning an implemented solution, as legislation regulating the use of AI increases.

Extending this theme is that of **validity**. EC methods (and optimizers in general) only optimize the function they have been given, and ML methods learn from the data they are provided. Explaining why a solution was chosen or how a prediction was made might help us know if it solves the actual problem, or if it just exploits an error or loophole in the problem's definition or a spurious relationship in the data. This not only can lead to surprising or even amusing results [5], but can also simply yield frustratingly incorrect solutions to a problem.

EC is stochastic and, as a result, some noise in the generated solutions is likely if not unavoidable. Different runs can produce similar solutions of equal quality but solutions can also feature artifacts that have no impact on their quality. Thus, another fundamental question is whether we can explain which characteristics of the solution are crucial: its **malleability**. In other words: *Which variables could be refined or amended for aesthetic or implementation purposes?*

Finally, when we define a problem, it is often hard to fully codify all the real-world goals of the system. For ML, this can lead to unwanted biases in the predictions if goals such as "fairness" are not explicitly coded in the cost function used for training. In optimization, subtle rules (for example, "I prefer not to work late on Fridays"; or "Joe likes to drive that route because it ends near his house") are typically used to judge solutions *after* the optimization is completed. We can generate lots of diverse solutions in order to "optimize" these goals post-hoc but we propose that, better still, an explanation could again reveal which characteristics are important for optimality, allowing one to refine the solutions and better *fit* the real-world problem. This also relates to one of the motivating factors behind interactive EC – we want something that is mathematically optimized, but also something that corresponds to the problem owner's hard-to-codify intuition. By incorporating XAI into interactive EC we could make it easier for the problem owner to interact with the optimizer [6].

More concretely, the types of questions we wish to answer with an explanation include [3]:

- Has the problem been formulated correctly?
- Are the patterns the model is drawing on to make its prediction the ones we expect?
- Why did the model make this prediction instead of a different one, and what would it take to make it change its prediction?
- Is the model biased and are the decisions made by the model fair?

A. What is an explanation?

It is difficult to define exactly what makes an explanation. Informally, an explanation aims to answer the question: "why?". Previous work has considered explanations to provide causal information [7], non-causal explanations [8], or as deductive arguments [9]. In this paper, we will consider an explanation to be an aid for a human to understand something about a model. The end goal of an explanation is to act as an interface between the model and the human, presenting information about the model in a way that is easier to understand. This means that the explanation does not need to capture the entire behavior of the model, but must communicate something important about it.

Explanations can be provided in many forms; examples include visualizations, numerical values, data instances, or text explanations [2]. They can also be provided as part of a dialogue between a human and an explainer [7], [10].

B. Explainability and Interpretability

The terms interpretability and explainability are often used interchangeably by researchers. In this paper, we distinguish between them as referring to two different but related aspects of attempting to understand a model [11], [12].

For our purposes, interpretability refers to whether a human can follow a model's decision-making process on its own. Such *interpretable* models do not require explanations because they are intrinsically self-explanatory. For example, models with a simple symbolic representation or small decision trees are generally considered interpretable. Note, however, that as the size of a model grows, it may become more difficult to follow the logic without external aid. Random forests and neural networks are examples of models that are theoretically interpretable at small scales but, due to their usually large size or ensemble behavior, are no longer considered to be interpretable. This aligns with Lipton's notion that interpretability is not one-size-fits-all [11]; instead, it covers a broad spectrum where complex models may be less transparent but achieve higher accuracy, while, conversely, simpler models are easier to understand but may sacrifice performance.

On the other hand, even if we cannot trace the exact logic, a model can still be considered *explainable* if a human-understandable explanation can be provided for what the model is doing or why a decision is made. Explanations do not need to capture the full behavior of the model, and in general, encompassing the full behavior in a single explanation is nearly impossible without creating an equally complex explanation. However, explanations can provide windows into particular aspects of the model's behavior. Some practical methods for providing explanations for particular aspects of

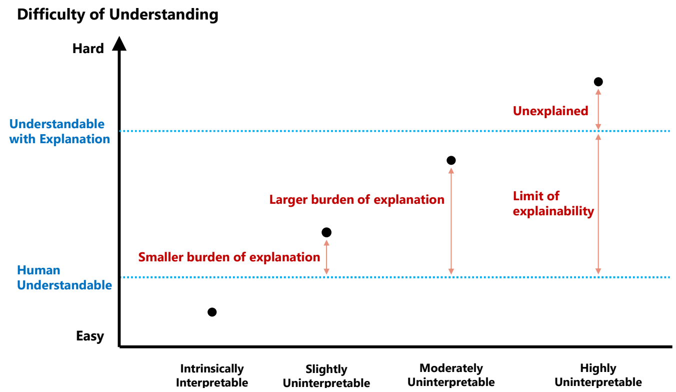

Fig. 1. As models and solutions become more difficult to understand, the amount of explanation required increases. Simple solutions (left) may not require any explanation at all, and are intrinsically interpretable. Others (middle two) may lie beyond the ability of a human to grasp easily, but can be understood with explanation. Finally, some models may remain opaque even with the current best efforts at explanation.

a model include evaluating feature importance, approximating the local or global behavior with a simpler model, or comparing the prediction to be explained with other similar inputs.

As illustrated in Figure 1, as an AI system becomes more difficult to understand the effort required to explain them adequately becomes greater. Below a certain threshold, depending on the audience, a system is easy enough to understand that it can be considered to be intrinsically interpretable. Above this threshold, explanations can provide sufficient aid that a normally uninterpretable system can still be understood. For example, a model might be multidimensional and uninterpretable when looking at its equations alone, but with the aid of visualizations, feature importances, and local approximations we can grasp its general behavior. As we consider larger or more complex models, we require more and more of these explanations to be confident we understand how the model works. At a certain point, it becomes impractical to explain to a satisfactory level how a system works, or the explanations we have are insufficient to capture everything. For example, although explanations can provide some insight into a large language model, there are still many aspects of its behavior that remains unknown. These systems are ones which lie beyond our current threshold of "understandable with explanation". This also points towards two ways of addressing this problem: first, by reducing the complexity of the model or otherwise bringing down the difficulty of understanding, so that existing explanation techniques can be used; second, by improving our ability to explain so that we can explain more uninterpretable systems. We will discuss specific ways of doing so in the later sections.

C. Why EC and XAI?

EC is an approach to AI inspired by the principles of biological evolution that have found use in a wide variety of applications, including optimization, ML, engineering design, and artificial life. This field encompasses evolutionary algorithms such as genetic algorithms (GA), genetic programming (GP), and evolution strategies (ES), as well as, for extension, swarm intelligence algorithms such as particle swarm optimization. EC techniques often employ populations of solutions and operators that introduce variation and diversity to explore larger regions of the search space.

Evolutionary techniques have unique strengths that can offer potential solutions to current challenges in XAI [13], [14]. First of all, as detailed in later sections, EC has a long track record of successful applications to create symbolic or interpretable models (e.g., decision trees or rule systems). By constructing solutions using intrinsically interpretable components, EC-derived solutions can guarantee the interpretability of the evolved representation. EC can also be used to create interpretable approximations of other complex models, i.e., to produce explanations for their behavior.

Second, the inherent flexibility of evolutionary methods, such as the ability to perform derivative-free, black-box optimization, positions them as a versatile tool to perform optimization where other methods struggle. For instance, they can be applied in scenarios where access to an ML model is only available through an API that returns predictions and no other information about the model's confidence or internal logic. Such a scenario is becoming increasingly common, but evolutionary methods can still perform optimization in these cases, to probe the model for patterns in its behavior or generate counterfactuals or adversarial examples. This also enables the optimization of unusual or customized metrics (for example, for measuring interpretability) without constructing metrics that can be readily optimized through gradient descent. Additionally, this flexibility also paves the way for hybrid methods when combined with other algorithms or to build meta-optimizers.

One especially useful capability offered by EC is multi-objective optimization. Many problems in XAI are inherently multi-objective, requiring a balance between model faithfulness and human interpretability, or complexity of the explanation. In many cases we may also want a diverse range of explanations – for example, different people may find different explanations to be helpful, or single explanations may not fully explain all the relevant characteristics of the model. The use of diversity metrics as well as quality-diversity algorithms can allow us to generate a range of different explanations for the problem.

Conversely, we also believe XAI principles can offer useful insights to EC and are currently under-utilized. XAI can provide insights into the decision-making process of evolutionary algorithms, explaining why certain solutions were selected in the end. Not only is this invaluable for debugging and improving algorithms, but end users may want to understand the reasoning behind why a particular solution was chosen. This is especially important in fields where the outputs of any algorithm must be justified or understood by decision-makers without deep technical knowledge of EC, such as in engineering design or policy-making.

Furthermore, XAI principles can aid in the development of more interpretable and transparent fitness landscape analyses. Understanding the fitness landscape is critical for understanding the difficulty of finding a solution as well as the effectiveness of EC algorithms. XAI-inspired approaches can

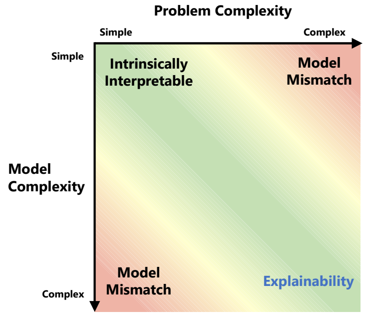

Fig. 2. The interaction between problem and model complexity. Simple models for simple problems are intrinsically interpretable, and do not require explainability techniques. On the other hand, applying simple models to complex problems may produce an interpretable model, but the model will not accurately solve the problem. Explainability is needed when we require a complex model for a complex problem, and we would like to understand the model.

improve our existing tools for visualizing and interpreting these landscapes, and understanding the landscape can serve as an explanation in and of itself.

III. EC FOR XAI

In this section, we discuss methods for XAI and the applications of evolutionary algorithms to this task. ML is a powerful tool for building a model of a system from data. As ML models have advanced, so has their complexity, often resulting in increased performance at the cost of interpretability. Explainability becomes a critical component to address this trade-off, ensuring that the models we rely on are not only effective but also understandable and trustworthy.

A. Explainability and Complexity

The interpretability of a model is inherently linked to its complexity [15]. Simpler models, such as linear regressions, are considered inherently interpretable due to their straightforward decision-making processes which humans can follow unaided. However, as model complexity increases, we lose interpretability and must rely on explainability instead. To clarify this relationship, we introduce a framework shown in Figure 2, mapping problem complexity against model complexity.

This begs the question: *What is the complexity of a problem?* We define problem complexity informally here, drawing parallels with the concept of computational complexity. We consider a problem's complexity to be the complexity of the model required to adequately capture its behavior up to the desired level of accuracy. The more complex the problem, the more complex the model must be to represent it faithfully. This also means that some problems may be simple if we are satisfied with a certain level of performance but may be complex if we aim to capture all the nuances and relationships in the data.

The complexity of a model (or a solution to an optimization problem) includes aspects such as the number of parameters, the depth of the structure, and the amount of computation required. This complexity can be measured by a model's description length [16], inspired by the concept of Kolmogorov complexity [17], or by parameterized complexity [15]. A model with a lengthy description, numerous parameters, or complex functions is considered more complex for our purposes. Although allowing for greater model complexity might increase a model's capacity to solve problems, this can lead to a loss of interpretability simply due to the size of the model. For example, a neural network with billions of parameters, a genetic program with a large number of instructions, or an extremely deep decision tree can produce accurate models for problems but be difficult to understand due to their size.

With these two axes in mind – that is, problem and model complexity – we can identify four main areas of concern, each of which represents a distinct combination of problem and model complexity. They are as follows:

- **Simple problem, simple model:** In this scenario, the desired behavior can be captured by a simple model. The model is accurate but also intrinsically interpretable, so there is no need for explainability in this case.
- **Simple problem, complex model:** When a complex model, such as a deep learning model, is used for a simple problem, the mismatch in complexity leads to a model that is excessively complex for the task at hand. While the model may perform well, it is difficult to interpret and offers no advantages for the complexity over a simple model which may perform equally well for the simple problem. In this case, the issue is not one of explainability but a result of the mismatch between the problem requirements and the model used. Rather than aiming to explain the complex model, a more fitting approach would be to use a model of the appropriate complexity and avoid the issue altogether.
- **Complex problem, simple model:** Conversely, applying a simple model such as linear regression to a complex problem can result in a model that is interpretable but with inadequate performance. Such a model may fail to capture the characteristics of the data to the degree of accuracy required. While the model remains interpretable, the issue is again a mismatch between the problem and model, as the simple model cannot accurately model the data.
- **Complex problem, complex model:** We argue that this quadrant is the main area of concern for XAI, i.e., the area where explainability is most relevant. In these cases, a complex model is necessary to capture the nuances of the data, but this complexity renders the model opaque and uninterpretable. Explainability methods enable users to navigate and understand complex models, building trust in the model, even when the model cannot be understood wholly on its own.

B. Types of Explanations

A wide variety of explanations focus on different aspects of the modeling process. In this survey, we take a problem-focused approach and consider the ML pipeline from data to trained model (Figure 3). we structure our categorization around the stage of the ML pipeline where they can be

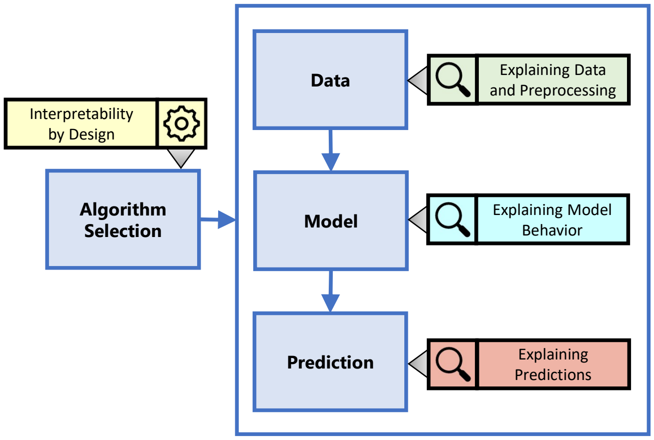

Fig. 3. Overview of the process of building an ML model, showing areas where explanations (magnifying glasses) are often applied. Examples of methods in each category are described in Section II. Also shown is the intrinsic interpretability approach (cogwheel), where models are designed to be interpretable from the start. All these methods can be used together to form a complete picture of a model's behavior.

applied. This is meant to highlight that explainability is not only applied at the end, to a fully trained model, but that it is a tool that can be used to understand the entire model-building process. By tying the categorization to the stages of the pipeline, we also aim to provide practitioners with a clear roadmap on areas where explainability can be applied.

In the next sections, we will address each area in turn. First, as an introduction to each category, we will describe some examples of conventional approaches. This overview is not meant to be exhaustive but rather to provide a primer on the current popular approaches. For a more extensive survey of current methods in XAI, we direct the reader to recent reviews [18], [19] on the topic. We will then survey EC-based approaches in each area, providing an overview of the current state-of-the-art with respect to combining EC and XAI.

C. Interpretability by Design

A growing concern on the importance of having ML models that are interpretable by design, rather than explainable post-hoc, has been recently advocated by many researchers in the XAI field [2], [12], arguing that whenever appropriate and possible, one should opt for models that are inherently explainable, or interpretable (white-box models). The main argument to support this statement is that post-hoc explanations often provide only local approximations of ML models, hence being limited: 1) because they rarely capture the whole (global) decision-making process of the model; and 2) because being approximations (essentially, they model other models) they can possibly introduce errors and thus not reflect the original decision-making process of the main model. For these reasons, "interpretable by design" models, based on some form of knowledge representation, should be preferred.

Unlike traditional ML approaches for the generation of models using such knowledge representations, which mostly use greedy heuristics, EC methods leverage the global optimization capabilities of evolutionary search. Learning Classifier Systems (LCS) are a notable instance of EC methods applied to ML. Within LCS, some methods apply batch learning [20], [21] while others use online learning [22]–[26] using either reinforcement learning (RL) [22] or supervised learning [27]. Most LCS approaches are applied to the generation of rule-based ML models, although other representations, such as decision trees [28] or hyper-ellipsoids [29], have also been explored. GP methods also have a long history of their application to symbolic regression [30].

Model complexity has been addressed in a variety of ways. For the EC methods generating variable-length models, the broad range of standard techniques to deal with the bloat effect [31] can be used to promote the generation of more compact models. For instance, specific fitness functions can be used to promote the generation of compact rule sets based on the minimum description length (MDL) principle [32], while other methods achieve this through multi-objective optimization [33]. Moreover, rule-based ML models can be simplified through the use of rule/rule set editing operators, which can be hybridized with the standard global search of EC methods (i.e., a memetic algorithm) [34], [35] or used as post-processing operators [36]–[38]. Sparsity in neural networks can be promoted through regularization, or through evolutionary pruning [39], [40].

Similar attempts at combining RL and EC have tried to obtain interpretable policies for RL tasks by combining decision trees induced by GP or Grammatical Evolution with RL acting on the leaves while the policy interacts with the environment [41]–[46], also through quality-diversity approaches [47], [48] and in multi-agent settings [49].

Some other works in this area have explicitly focused on addressing the interpretability question in white-box models. The balance between accuracy and interpretability has been explored in the context of genetic fuzzy systems [50]. In this regard, some recent studies have proposed machine-learned quantifiable measures of interpretability [6], while others [51] have emphasized the importance of focusing on low-complexity models, especially in the context of GP. Another important aspect in ML, *fairness*, has been addressed in [52], where explicit fairness constraints have been introduced in GP to obtain fair classifiers.

Visualization techniques in the shape of heatmaps have been used to represent the sets of classification rules generated by LCS [53]. This technique was particularly effective when combined with hierarchical clustering to reorder a dataset's rows (instances) and columns (features), as this enabled an effective global view of how the problem domain was partitioned across the classification rules and what features were relevant for each partition. Alternatively, 3D visualization approaches have also been shown to be a very effective tool to represent complete rule sets generated by LCS [54], by using different axes to represent attributes, levels of generality of the rules in which these attributes were involved, and estimated attribute importance.

D. Explaining Data and Preprocessing

We begin with a discussion of methods that can be used to explain the data. These are methods that aim to understand the structure of the data, for example through clustering, even though they may not become part of the final model. Although this category is often omitted in discussions of XAI, it is worth mentioning it as part of the overall pipeline. Every ML model begins with the data, and any pattern learned by the model is derived from the data, so it would be remiss not to discuss this as a component of the whole process. We should remark that methods under this category do not necessarily explain the model itself, but as said aim to explain the underlying data on which the model was trained, focusing on understanding the data distribution and its characteristics.

Techniques such as exploratory analysis, data visualization, and dimensionality reduction can be used to better understand the patterns in the underlying data that the model might learn, as well as identify any potential biases. Examples of these techniques include Principal Component Analysis (PCA) [55], [56] and t-Distributed Stochastic Neighbor Embedding (t-SNE) [57], which reduce the dimensionality of data to allow for easy visualization.

In addition, methods such as clustering and outlier detection can help identify patterns or anomalies in the data that may impact the model's performance and aid in feature selection and engineering. These include methods such as k-means clustering and DBSCAN [58]. These explanations can help identify data quality issues, biases, and preprocessing requirements, as well as build trust.

1) Dimensionality reduction: EC can be used to explain data by means of dimensionality reduction and visualization. One approach is GP-tSNE [59], which adapts the classic t-SNE [57] algorithm to use evolved trees to provide an interpretable mapping from the original data points to the embedded points. Similarly, Schofield and Lensen [60] use tree-GP to produce an interpretable mapping for Uniform Manifold Approximation and Projection (UMAP). By producing an explicit mapping function rather than simply the embedded points, we can not only make the process more transparent but also reuse the mapping on new data.

In some cases, we may want to use the lower-dimensional representation for prediction as well as visualization. This is useful for interpretability as it allows us to visualize exactly the same data representation the model sees. Therefore, another approach is to construct features that are both amenable to visualization and well-suited for downstream tasks. Icke and Rosenberg [61] proposed a multi-objective GP algorithm to optimize three objective measures desirable for constructed features – classifiability, visual interpretability and semantic interpretability. Similarly, Cano et al. [62] developed a method using multi-objective GP to construct features for visualization and downstream analysis, optimizing for six classification and visualization metrics. The classification metrics (accuracy, AUC, and Cohen's kappa rate) aim to improve the performance of the downstream classifier, while the visualization metrics (C-index, Davies-Bouldin index, and Dunn's index) aim to improve the clustering and separability of the features.

Moreover, GP has been effectively used for the ML task of manifold learning [63], i.e., the creation of (ideally) reduced data representations for high-dimensional datasets to facilitate the work of downstream ML algorithms. Often, this task is solved by black-box algorithms that perform a mapping from an original space to a reduced one without a clear explanation of how this mapping is designed. On the other hand, GP trees offer an interpretable alternative for this task with white-box transformation operations.

2) Feature selection and feature engineering: Feature selection is a common preprocessing step in which a relevant subset of features is selected from the original dataset. The latter is used to improve the model's performance and interpretability by narrowing down the features the model can draw on. As an explanation, feature selection shares some similarities with feature importance, which identifies the features a model is drawing on but, instead, restricts the model explicitly so it can only draw on the chosen features.

Genetic algorithms are a straightforward and effective approach to feature selection, with a natural representation in the form of strings of 1s and 0s, making them a popular choice for feature selection [64]–[66]. GP can also be used for feature selection since the inclusion of features in a tree or linear genetic program is intrinsically evolved with the program [67]–[69]. For an in-depth review of GP methods, we refer the reader to [70]. Swarm intelligence methods, such as particle swarm optimization, have been applied to feature selection as well [71]. For a more detailed review of these methods, we direct the reader to [72]. In addition to selecting features for a model, feature selection can also be used to improve data understanding by integrating it with techniques such as clustering [73].

Feature engineering, also referred to as feature construction, is a related approach that involves building higher-level condensed features out of basic features. GP can be used to evolve these higher-level features for downstream tasks such as classification and regression [74]–[77]. This approach can also help improve a model's interpretability since it can reduce a large number of low-level features to a smaller number of higher-level features which may be easier to understand for humans. Moreover, it removes some of the modeling needs from the black-box, replacing them with an a-priori (preprocessing) transparent step, thereby reducing the amount of explanation needed.

These methods also share many similarities with dimensionality reduction techniques and, in some cases, can fall under both categories. Uriot et al. [78] compared a variety of multi-tree GP algorithms for dimensionality reduction, as well as a tree-based autoencoder architecture. In the multi-tree representation, each individual in the population is a collection of trees each of which maps the input to one feature in the latent dimension. In order to reconstruct the input for the autoencoder, a multi-tree decoder is simultaneously evolved with one tree per input dimension. Their results showed that GP-based dimensionality reduction was on par with the conventional methods they tested (PCA, LLE, and Isomap).

E. Explaining Model Behavior

Once we have the trained model, it may still be difficult to understand how it works, even in cases where it is transparent and we can inspect the internal mechanisms. Consider, for example, a trained neural network. Even if in principle we are allowed to inspect each weight and internal operation, the model as a whole is still hard to understand. This is where we turn to explanations to bridge the gap. Methods in this category attempt to explain the internal function of the model, for example, by inspecting the structure of a tree or the weights in a neural network. These approaches can either attempt to explain the entire model or to understand smaller components of the model.

1) Feature importance: Global feature importance aims to explain the dependence of a model on each feature it uses. For example, feature importance returns a score that represents the significance of each feature to the model. This helps identify which features impact the model's predictions most and provides insights into how the model is making decisions. This type of explanation can also be used to verify whether the model is behaving as expected – for example, by checking whether it is using the same features a human would to solve the problem. In the case of a computer vision model, this type of explanation can be used to determine if the features used to classify a particular image as a cat make sense or if the model is using spurious patterns in the data, such as identifying the cat based on its surroundings. This type of explanation can also aid in optimizing models and performing feature selection by identifying less important features. Some models, such as decision trees and tree ensembles like random forests, provide built-in feature importance measures [79]. For models without built-in feature importance measures, more general methods such as partial dependence plots [80] and permutation feature importance [79], [81] can be used to determine which features have the largest impact. Evolutionary computation can be used to go further and measure the strength of higher-order interactions between features by evolving groups of features [82].

2) Global model approximations: This approach, also known as model extraction or a global surrogate model, aims to approximate a black-box model with a more interpretable model. This idea is closely related to knowledge distillation [83], [84] in deep learning, but rather than simply making the model smaller, we also want to make it more interpretable. This is done by training or evolving a secondary model, which both approximates the original model and is more interpretable. An example of this approach was proposed by Lakkaraju et al. [85]. Their approach approximates the behavior of the model using a small number of decision sets, providing an interpretable proxy for the entire model.

Evolutionary computation methods such as genetic programming are well-suited for this approach as they can guarantee interpretable models while optimizing for one or more objectives. Evans et al. [86] propose a model extraction method using multi-objective GP to construct decision trees that accurately represent a given black-box classifier while being more interpretable. This method aims to simultaneously maximize the ability of the tree to reconstruct (replicate) the predictions of a black-box model and maximize interpretability by minimizing the decision tree's complexity. The reconstruction ability is measured by the weighted F1 score over cross-validation, and the complexity of the decision tree is measured by the number of splitting points in the tree. The overall evolutionary process uses a modified version of NSGA-II [87]. In their experiments on a range of classification problems, the authors found that the accuracy remained commensurate with other model extraction methods (namely Bayesian rule lists, logistic regression, and two types of decision trees) while significantly reducing the complexity of the models produced.

3) Domain-specific knowledge extraction from machine learning models: Finally, domain-specific studies have also been performed. For instance, the classification rules evolved by EC methods have been analyzed in the domain of protein structure prediction [88]. Furthermore, biological functional networks (i.e., graphs) can be inferred by mining the structure of ensembles of rule sets evolved by EC methods [89]. A topological analysis of such networks led to the experimentally-verified discovery of the function of several genes (in the biological sense of the word) for the *Arabidopsis Thaliana* plant organism [90]. Knowledge representations for rules can be constrained in a variety of ways, which shape the data patterns captured by the sets of classification rules using such representations. This potentially leads to the extraction of different knowledge from the same data depending on the chosen representation, as was studied for molecular biology datasets [91]. In the field of neuro-evolution, EC methods have instead been used to discover interpretable plasticity rules [92]–[94] or to produce self-interpretable agents [95], i.e., agents that (through self-attention) access a smaller fraction of the input, for which interpretable policies are possible.

4) Explaining neural networks: Thus far, the methods we have covered are general and can be used with a variety of models. However, given the popularity of deep learning methods, we would be remiss not to discuss methods specifically tailored to explaining these models. The rise in popularity of deep learning, combined with the inherent black-box nature of neural networks and their large number of parameters, makes explaining them challenging but increasingly important.

For image classification, the large number of input features (pixels) poses a significant problem for many explanation methods. As such, it is necessary to reduce the dimension first, for example by clustering similar pixels into "superpixels". Wang et al. [96] propose using a multi-objective genetic algorithm to identify superpixels of importance for the final prediction and using this set of superpixels as an explanation. The genetic algorithm uses NSGA-II to optimize for the least number of superpixels used while maximizing the model's confidence in its prediction.

Methods have been developed for explaining the internals of deep learning models [97]. As an example of one such method, Interpretable Lens Variable Models [98] train an invertible mapping from a complex internal representation inside a neural network (i.e., the latent space in a generative or discriminative model) to a simpler, interpretable one.

More recently, the field of "mechanistic interpretability" has gained popularity, aiming to understand the internal operations and mechanisms of neural networks. This field attempts to reverse-engineer and describe the algorithm performed by the layers of a neural network. One promising thread of work has been the Circuits approach, which discovered curve detectors in vision models [99] and interpretable circuits in small transformer models [100].

As an example of this approach, mechanistic interpretability has been used to explain the "grokking" phenomenon seen when training neural networks [101], by which some networks learn the training data quickly but only generalize well after a long period of further training in which little appears to happen. In this work, Nanda et al. show that, when training a transformer model to perform modular addition, the network first memorizes the training data directly before eventually learning a general algorithm for the problem. They are able to describe the exact algorithm used by the network, by inspecting the activations and ablating specific components.

This is a space that is ripe for innovation in the evolutionary computation community. Evolutionary methods have been used to explore the decision boundary [102] and prompt space [103] of language models, attempting to map out the space of inputs along relevant dimensions. Another approach is to directly interpret the network, such as by using symbolic regression to find a expression that matches the gradients of the network [104].

F. Explaining Predictions

This type of approach aims to explain a specific prediction made by a model. As such, the explanation only needs to capture the behavior of the model with respect to the prediction in question, rather than the model as a whole.

1) Local explanations: Instead of creating an interpretable model to approximate the global performance of a black-box model, which may not be possible, these approaches only attempt to approximate the local behavior using an evolutionary algorithm. One notable method in this category is Local Interpretable Model-agnostic Explanations (LIME) [105]. LIME constructs an explanation by generating a collection of instances in the vicinity of the input to be explained, each accompanied by the model's prediction. It then fits a linear model to this new dataset, serving as a local surrogate that approximates the original model's behavior in that specific region. While this explanation does not necessarily reflect the global behavior of the model, it is locally faithful and can be used to understand the behavior of the model around that point.

Ferreira et al. [106] proposed Genetic Programming Explainer (GPX), a GP-based method that fits a local explanation model for a given input example. Similar to LIME, when given a sample input to be explained, the method samples a set of neighboring data points around the input and fits a local explanation. However, rather than a linear model, GPX uses a GP to evolve symbolic expression trees that best capture the behavior of the pre-trained black-box model over the neighboring data points. The authors tested this approach on both classification and regression datasets and reported that the GP-based approach captured the model's behavior better than LIME, as the assumption of linear local behavior was not always valid, and also outperformed a decision tree used as an explainer for the same neighbor set.

On the other hand, Guidotti et al. [107] proposed a method called Local Rule-based Explanations (LORE), which applies an evolutionary algorithm to neighborhood generation rather than evolving the explanation itself. Specifically, a genetic algorithm generates a set of points near the prediction to be explained, which are either classified the same as or differently from the original prediction while being nearby. A decision tree is then used to fit the local behavior of the black-box model. The use of a genetic algorithm here ensures a dense sampling of points in the local neighborhood that lie on both sides of the decision boundary.

Feature importance can also be provided for individual predictions. Shapley additive explanations (SHAP) [108] attempts to provide a universal method for assessing feature importance that can be applied to most ML models. This is based on the Shapley value, a concept from cooperative game theory that assigns a value to each player in a game based on their contribution to the overall outcome. In the context of ML, the "players" are the features in the data, and the "game" is the prediction task. The Shapley value scores each feature based on its contribution to each prediction. The exact calculation of Shapley values is usually computationally impractical, as it involves evaluating every possible combination of features. However, SHAP proposes approximating these values using sampling and regression, making the estimation of feature importance computationally feasible. This method is widely used in the field of XAI.

2) Counterfactuals: Counterfactual explanations are another type of explanation that provides insight through a hypothetical example where the model would have made a different decision. For instance, "the model would have approved the loan if the income were $5000 higher" is a counterfactual explanation that identifies how the input should change in order to change the model's result [109]. This form of explanation is intuitive and can be performed on a model in a black-box manner without access to the internal logic of the model. Another notable advantage of counterfactual explanations is that they afford users actionable steps or *recourse* to achieve a desired result [110]. They are also inherently faithful to the model's behavior since they are grounded in actual model evaluations. However, because they consist of single instances or data points, they only provide limited insight into the model's global behavior.

Diverse Counterfactual Explanations (DiCE) [111] is an method of constructing counterfactual predictions. The aim of this method is to produce counterfactuals that are valid (produce a different result when fed into the model), proximal (are similar to the input), and diverse (different from each other). Diversity is desirable here as it increases the likelihood of finding a useful explanation and provides a more complete picture of the model's behavior. DiCE generates a diverse set of counterfactual examples using a diversity metric based on determinantal point processes [112], a probabilistic model that can solve subset selection problems under diversity constraints. This diversity constraint forces the various examples apart, while an additional proximity constraint forces the examples to lie close to the original input. The method also attempts to make the counterfactual examples differ from the input in as few features as possible (feature sparsity).

Evolutionary computation is well suited to this task as a black-box, possibly multi-objective optimizer, as it allows us to find counterfactuals without knowing the internal workings of the model while also optimizing for multiple desirable criteria in the counterfactuals.

CERTIFAI [113] generates a population of counterfactual explanations using a model-agnostic genetic algorithm. The initial population is generated by sampling instances that lie on the other side of the decision boundary of the model (i.e., are classified differently from the instance to be explained). Then, the genetic algorithm optimizes the population to minimize the distance (for some notion of distance, depending on the type of data) from each counterfactual instance to the input instance. The population is then analyzed for (1) robustness, which increases if the best counterfactual examples found are farther away from the input, and (2) fairness, which is measured by comparing robustness across different values of a particular feature.

GeCo [114] uses a genetic algorithm with feasibility and plausibility constraints on the features, specified using the constraint language PLAF. This allows one to rule out certain counterfactuals that would be useless to the user (e.g., counterfactuals where the user changes their gender or decreases their age). Like CERTIFAI, the genetic algorithm minimizes the distance from the input instance to the counterfactual examples, prioritizing examples on the other side of the decision boundary while keeping examples close to the decision boundary if not enough counterfactuals are available. The fitness function does not consider how many features are changed relative to the input instance (with a smaller number being preferred for ease of understanding), but the algorithm is biased toward a smaller number of changes by initializing the population with only one feature changed.

Multi-objective counterfactuals (MOC) [115] explicitly use multi-objective optimization to consider multiple desirable properties of the explanations. MOC uses a modified version of NSGA-II to perform its search. Among the changes are the use of mixed integer evolution strategies (MIES) [116] to search a mixed discrete and continuous space and a different crowding-distance sorting algorithm which prioritizes diversity in feature space. A total of four objectives are used, optimizing for these four desirable properties: the model output for the example should be close to the desired output; the example should lie close (in the feature space) to the input to be explained; the example should not differ from the input in too many features; and, the example should be plausible (i.e., likely to be drawn from the same distribution as the real data), which is measured by its distance to the closest k data points.

3) Adversarial examples: Adversarial examples are closely related to counterfactuals. An adversarial example is counterfactual, but it intends to create an incorrect prediction [117]. This is done by applying a small perturbation to an example to change its classification. Most approaches search for examples that are as close to the original input as possible and perceptually similar to the input. These examples are a method to highlight failure modes of the model as well as a potential attack vector on deep learning models.

Su et al. [118] propose a method of finding adversarial examples which modify only one pixel in an image. This contrasts previous methods that modify multiple pixels in the image and are more obvious to humans. Their method uses differential evolution, where each individual is encoded by the coordinate of the pixel to be modified and the perturbation in the RGB space. They find that, in many cases, one pixel is sufficient to deceive the model. Other works explored the generation of adversarial image perturbations through evolution strategies [119] and the clonal selection algorithm [120].

Adversarial examples are also present in models built for other domains, such as natural language processing. Alzantot et al. [121] generate adversarial examples on a sentiment analysis model and a textual entailment model. In addition, the examples they produce are designed to be semantically and syntactically similar to the original input, making the attack more difficult to spot. A genetic algorithm is used to optimize for a different target label than the original. Mutation occurs by changing words in the input to similar words as measured by a word embedding model (GloVe) and filtering out words that do not fit the context.

G. *Assessing Explanations*

Finally, rather than using EC to generate the explanations themselves, we will discuss some ways in which EC can be used to assess or improve the quality of other explanation methods.

Huang et al. [122] propose two metrics to assess the robustness of an explanation: worst-case misinterpretation discrepancy and probabilistic interpretation robustness. Interpretation discrepancy measures the difference between two interpretations, one before and one after perturbation of the input. For an interpretation to be robust to perturbations, it is desirable for this value to be low. The authors then measure the discrepancy in two worst cases: the largest interpretation discrepancy possible while still being classified as the same class and the smallest interpretation discrepancy possible while being classified differently (adversarial example). These values are optimized using a GA. The other metric, probability of misinterpretation, calculates probabilistic versions of the above: the probability of an example having the same classification but a significantly different interpretation and the probability of an example having a different classification but a similar interpretation. This is estimated using subset simulation.

It is also possible to perform an adversarial attack on the explanations themselves. Tamam et al. [123] do this with AttaXAI, a black-box approach based on evolution. AttaXAI tries to evolve an image similar in appearance to the original input that produces the same prediction from the model but with an arbitrary explanation map. In their experiments, pairs of images were selected and were shown to be able to generate a new image with the appearance and prediction of the first image but with a similar explanation map to the second.

Much of the visualization work described above has a considerable drawback when considering explainability in that only a limited amount of evaluation has been undertaken with explainability in mind. A standard approach to evaluating visualization research within EC is to apply visualizations to a benchmark dataset – perhaps a benchmark approximation set, a group of Pareto front approximations generated on multi-objective test problems with known characteristics [124], or a run of an algorithm on a problem that has a specific type of landscape whose performance we would like to visualize. These are valuable approaches, as they confirm that a proposed visualization technique is able to represent the characteristics of solutions or algorithm execution that we seek to present to users. Other approaches included examining the features offered by a visualization according to a usage taxonomy and undertaking a usability study. These latter approaches are important to the analysis of visualizations from an explainability perspective.

A usability study is a process by which a human user's ability to use a computer system is formally evaluated. In the context of visualization, this typically assesses the extent to which a user can interpret the information presented in a visualization. A few examples can be found in the EC literature wherein a usability study has been conducted. A small-scale usability study was conducted in [125], wherein participants were asked to engage in a number of tasks (selecting the best and worst solutions from a number of visualizations) and were assessed on their accuracy and time taken to complete the task. Another study [126] asked users to reflect on their use of a visualization tool using a questionnaire with a range of scored and open questions. Considering the range of cases in which usability studies have been used within the wider visualization community, we argue that the EC community can gain much from incorporating them. As a first step, studying the accessibility of existing methods on a considerably larger scale – both in terms of respondents and the tasks they are asked to complete – is recommended. This, in turn, will require careful consideration since, for example, the DTLZ test problem suite used to showcase many of the visualization tools discussed above is not easily interpretable by non-experts. Instead, benchmark tasks that human users can easily understand must be identified. How best to leverage usability studies within the wider context of XAI is still an open question [127], with proposals including the creation of question banks [128], or evaluating different query and modality types [129]. All of these approaches can be readily adapted for use within EC.

IV. XAI for EC

In this section, we consider a complementary perspective to that in Section III: explainability for EC and optimization approaches in general. The motivation is similar: an optimization algorithm will follow lengthy and often complex processes to find optimal or near-optimal solutions that are presented to a decision-maker. Explanations here also help the decision-maker answer our general questions set out in Section II. Overall, we view the process (Figure 4) of optimization as having three stages: problem setup or definition, iterative optimization or search, and analysis of solutions. There is scope for explainability at each of these stages, which we will elaborate on in the following sections.

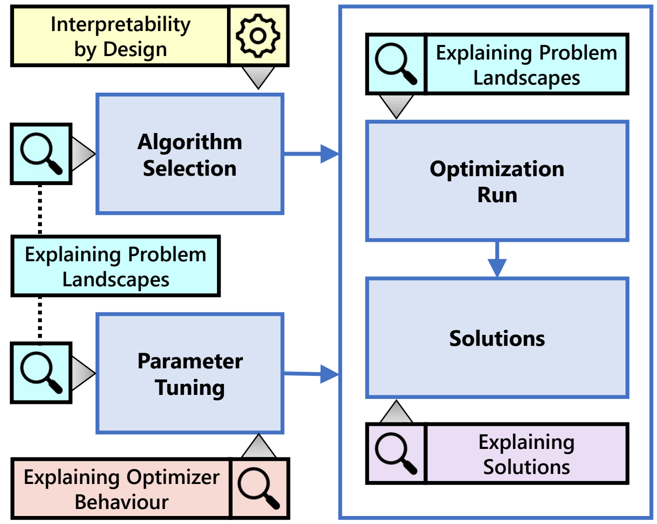

Fig. 4. Overview of the process of using an optimization algorithm, showing areas where explanations (magnifying glasses) can be applied.

A. Interpretability by Design

Much of the challenge in tackling an optimization problem is in designing and formulating the objectives and solution representation. Interpretability can be a key factor in the design choices made at this stage; conversely, the problem's design is an important part of explaining it. Thus, more direct representations and explicit encoding of the variables, objectives, and constraints of real-world problems might be favored if interpretability is important. In this setting, a MILP formulation of a problem's objectives and constraints, whether to be solved by mathematical optimization or EC, would be preferable to a "black-box" function evaluation. Matheuristics [130] are a successful development in this area. An alternative approach [131] used decision trees to provide interpretable rules to choose solutions for optimization problems with the trees constructed by MILP or heuristics. Handling the components of an objective separately rather than together can permit post-hoc analysis of how solutions have been chosen throughout the evolutionary process; this motivates the use of lexicographical approaches to tournament selection such as those proposed by Deb for constraints [132] and multiple objectives [133], or lexicase selection in GP [134]. The tackling of multi-objective problems can also be made more transparent through the use of *post-hoc* multi-objective evolutionary algorithms [135]–[137] that approximate a Pareto front, allowing the decision-maker to understand the trade-off between objectives, rather than having to guess the trade-off when choosing weightings for a priori optimization (e.g., a weighted sum of objectives). We will return to this theme in Section IV-C.

Explainability also motivates the use of direct over indirect representations: it is self-evident that the closer the solution representation and decision variables are to the real-world application, the easier the solution explanations are in the applied setting. There is often a trade-off here. For example, indirect representations such as hyperNEAT [138], [139] and Grammatical Evolution have generally been found to outperform direct representations such as the tree structures of classical GP [140]. On the other hand, more explicit formulations of the problem allow for greater control of the operators that can be customized to the problem at hand. For example, grey-box optimization [141] exploits knowledge of the problem domain using direct encodings for combinatorial problems in order to improve performance. As with ML, as noted in Section III-D2, direct representations should still be at the right level for interpretability: too low and the information is too fine-grained and dense for a human to interpret; too high and the real-world meaning is lost. We suggest that there could also be scope for evolutionary algorithms in engineering and selecting features with respect to interpretability for optimization problems themselves; this may be accomplished by following a cooperative coevolution approach as is already successful in large-scale global optimization [142].

The algorithmic framework itself also has an impact on explainability. A greedy or steepest ascent hill-climber is deterministic, with a single point to follow, resulting in an easily understood search process more interpretable than a stochastic search or a population-based algorithm. In this regard, Estimation of Distribution Algorithms [143]–[146] construct explicit representations of the problem, highlighting a clear mathematical route to the solutions, though this task can itself be sufficiently complex as to be non-interpretable.

B. Explaining Problem Landscapes

Landscape analysis aims to capture the interactions between algorithms and their operators with solution representations. Such approaches might be considered as being about understanding *how* the search proceeds rather than *why* particular solutions are chosen, but both aspects can be viewed as important for explainability.

1) Landscape analysis and trajectories: Landscape analysis [147] is, arguably, one of the main points of contact between XAI and EC. Landscape analysis, in fact, encompasses a set of tools that aim to understand and explain algorithm behavior based on the problem features, as well as predict algorithm performance and perform automatic algorithm configuration and selection. In this area, some works that explicitly aim at explainable landscape-aware prediction [148], [149] have been proposed recently.

We might consider an algorithm's behavior to be defined in terms of its trajectory through the search space. Such a trajectory is the sequence of points occupied in the search space by the algorithm's population (or even a single solution, in the case of single-solution algorithms) over the course of the algorithm's run. In this way, the trajectory captures the algorithm's progress: when particular features of the solutions were discovered, when the algorithm got stuck in a local optimum or on a plateau, when premature convergence occurred, and so on. [150] introduced the concept of search trajectory networks as a tool to visualize these trajectories, demonstrating the approach for several combinations of algorithms and problems.

Search trajectories have also been explicitly proposed as a promising route towards XAI for EC [151]. In this work, Principal Component Analysis is applied to solutions visited by an EA in order to capture features prevalent in the population of an algorithm at each generation. Each component captures features in the decision space, with the loadings for each component identifying correlations between groups of variables. As variables begin to converge over the run, each component varies in prominence within the population, allowing for visualization of the algorithm's progress. The authors also demonstrated strong connections between the loadings of each component and known global optima for multiple bit-string encoded benchmark functions (e.g., groups of k-bits being correlated for trap-k functions and alternate bits negatively correlated for alternating-ones functions).

In a related work [152], the authors propose a feature extraction method that describes the trajectories of optimization algorithms using simple descriptive statistics. These statistics can then be used by ML methods for performance prediction or the automatic configuration of an algorithm on unseen problems.

Population Dynamics Plots were also recently proposed in [153] as a way to visualize the progress of an EA as the search proceeds, allowing the lineage of solutions to be traced back to their origins and providing a route to explain the behavior of different algorithms. The authors visualized solutions to multi-objective knapsack problems in terms of their objectives, projecting multi-objective values down to two dimensions for visualization. The proximity of solutions to the feasible/infeasible boundary was captured, as were the convergence behaviors of different algorithm configurations.

A further alternative approach to capturing the trajectory of a metaheuristic run is the creation and mining of surrogate fitness models fitted to the population [154], [155]. Surrogate models are most commonly used to speed up the runs of EAs by training a model that takes the place of the fitness function. The idea proposed by [154], [155] is that the surrogate model is biased towards the solutions visited by the EA as it runs, and probing the model reveals the algorithm's perspective of characteristics like the sensitivity of the objectives to each variable, as well as inter-variable relationships. These papers presented some preliminary results on well-known bitstring-encoded benchmark functions.

Studies about *hyper-heuristics* [156] and *parameter selection* [157], instead, have highlighted that specific parameter settings allow EC methods to exhibit a "generalistic" behavior, i.e., to perform generally well even on very different types of functions. The search for such settings has been shown to be effective, for instance, in selecting solutions from the Pareto fronts of multi-objective optimization problems [158]. Stemming from these considerations, it might be worth exploring whether the search for simple parameter configurations motivated by an easier explainability of the corresponding algorithm may also lead to generalistic solutions, as the ML theory (and Occam's razor) seems to suggest.

2) User-guided evolution: Allowing the user to influence or provide input into the model-building process can also improve trust. An approach described by [159] combines the principles of parallel coordinate plots with a multi-objective EA to allow users to define areas of interest where they would like to find solutions.

Another mechanism for understanding the solution landscape is through quality-diversity or illumination algorithms, such as MAP-Elites [160], [161]. These algorithms can generate diverse high-quality solutions varying along user-defined dimensions. This allows the user to understand how the quality of a solution varies with respect to different parameters, which may differ completely from the underlying parameters used by the model. An interesting future direction here could be the design of algorithms that provide human-interpretable explanations as they proceed, incorporating human feedback as part of the search. This might resemble the preference-based approach used in multi-objective optimization [162] but focused on the decision space.

Gaier et al. [161] proposed a hybrid approach by using MAP-Elites alongside a surrogate model to add efficiency to the MAP-Elites process. They proposed reducing the need for the large number of checks normally required for MAP-Elites. The proposed solution, Surrogate-assisted illumination (SAIL), aims to achieve this by integrating an approximation model (surrogate) alongside an intelligent sampling of the fitness function. As with MAP-Elites, the search space is partitioned into *shape bins*, each of which holds a map with a different layout of feature values. Firstly, a surrogate is constructed based on an initial population of possible solutions, also including their fitness scores. MAP-Elites is then used to produce solutions to maximize the fitness function and generate an acquisition map. Thereafter, new solutions are sampled from this map and additional observations are used to iteratively improve the model, looping through this process to generate increasingly better solutions. The performance predictions are then used by MAP-Elites in place of the original fitness function to generate a prediction map of near-optimal representations.

Urquhart et al. proposed in [163] an application of MAP-Elites to increase trust in metaheuristics. This paper specifically aimed to address the criticism that end-users have no role in the construction of the end solution. The authors proposed that MAP-Elites can be used to filter the solution space and provide a set of solutions for the users, from which they can select the one most applicable to them and their needs. This increases trust in the selected solution because the user is provided with an opening to the process and a measure of influence as to what constitutes a *good* solution.

More recently, [164] presented an extension of the MAP-Elite process that extracts explainable rules from MAP-Elite archives. This work addresses the issue that MAP-Elites generate thousands of solutions to a problem; extracting information from such a large number of solutions is a challenge for a decision-maker. Instead, the authors proposed the use of GP and a rule-induction approach that generates a small number of rules that capture the characteristics of the solutions generated by the optimizer.

C. Explaining Solutions

XAI for EC is applicable in many stages of the EC pipeline as outlined in Figure 4. The output of the optimization run – the solutions – could also be mined for explanatory artefacts. The generation of such artefacts requires the post-hoc evaluation of the solutions created by the optimizer, whether for Pareto fronts, populations, or single solutions. This post-hoc analysis involves, in essence, the exploration of alternative causes to generate an explanation regarding solution quality and to reveal something about the model used.

1) Interpreting solutions: The interpretability of solutions can often be difficult to define. As noted by [2], it may broadly be understood as the "extraction of relevant knowledge from a machine-learning model concerning relationships either contained in data or learned by the model". This is connected to the older concept of *backbones* [165], which represent components of a solution that are critical to its optimality. In a satisfiability decision problem, the backbone of a formula is the set of literals which are true in every mode. Identification of such characteristics in a solution could form part of an explanation of its quality.

Dimensionality reduction techniques have been shown to help explain optimizer solutions, as proposed in [166]. Here, the latent problem structure and its effect on optimizer output are investigated by decomposing the search trajectories using Multiple Correspondence Analysis (MCA). By projecting the trajectories into variance-based lower-dimension spaces, feature importance at various stages of the search can be determined. These, in turn, may be used to aid end-users in interpreting both high- and low-impact influences to a solution in single-objective problems.

In multi-objective space, the trade-off between a solution's explainability and its representation's accuracy has been explored in [167]. Here, a successful reduction in the complexity of the explanation representation is achieved through the step-wise regularization of the set of linear regression models generated from the output of the optimizer. This reduction retains the interpretability of the solution explanation while maintaining the predictive ability to outline domain-relevant mappings between the regressors and the objective function.

Innovisation [168], [169] was proposed by Deb et al. so that design principles shared between solutions to multi-objective optimization problems can be identified, explaining optimality in the decision space by highlighting the principles that ensure Pareto-optimality. While innovisation emphasizes understanding the underlying principles that lead to optimal solutions, more recently a focus on exploring the factors that lead to maintaining a level of coherence between solutions can be observed in [170]. Here, methods for maintaining the similarity of multi-objective solutions comprising the Pareto front are investigated. This is done to provide experts with a smoother view of the transition in the solution space between the solutions in a Pareto front approximation.

2) Visualization of solutions: Within the many-objective optimization community, a considerable body of work exists around visualizing Pareto front approximations. The challenge therein is representing solutions' objective vectors in cases with $M > 3$ objectives. Human cognition prevents decision-makers from comprehending four or more spatial dimensions, and work has therefore focused on three approaches: (1) identifying visualization techniques that can present solutions in terms of the full set of objective vectors; (2) identifying objectives that are redundant and can be discarded so that a standard visualization tool can be used; and (3) approaches for applying feature extraction to identify new coordinate sets that can more easily be visualized.

The first category comprises techniques such as parallel coordinate plots [171], [172] and heatmaps [173]. Both are popular techniques in the visualization community for visualizing large datasets because of their ability to scale. Both can handle many data items (such as the objective vectors in a Pareto front approximation) and features (corresponding to the objectives in this case). Unfortunately, both techniques suffer from a lack of clarity in their basic form. In the case of the parallel coordinate plot, solutions overlay each other, which leads to a large proportion of the solution set being obscured. Heatmaps are arbitrarily ordered — in terms of both rows and columns -– causing the relationship between pairs (or larger groups) of solutions or objectives to be extremely difficult to observe. In both cases, simply reordering the data can help with the accessibility of the methods. Parallel coordinate plots have seen the objectives reordered for the trade-off between objectives to be more readily identified [174] while the clarity of heatmaps has been improved by reordering both the rows (solutions) and the columns (objectives) with agglomerative clustering [173] and spectral clustering [175] to better reveal patterns and trends. Parallel coordinate plots have been further enhanced using user interaction, such that the users can filter out solutions that are outside of the objective value bounds they specify. This reduces their cognitive load by requiring them to focus on fewer objective vectors [172].

The latter two categories both deal with dimensionality reduction. Feature extraction techniques that have been used include PCA [176]; self-organizing maps (SOM) [177]; and multidimensional scaling (MDS) [178], among others. Whatever the technique, the approach relies on projecting the objective vectors from $\mathbb{R}^{M>3}$ into a new space $\mathbb{R}^{M\in\{2,3\}}$. This makes it possible to use a standard visualization technique such as a scatter plot, which enables the Gestalt principles of presentation to be followed – similar points in the visualization are placed close together, for example. From an explainability perspective, however, projecting objective vectors into a new space that bears no resemblance to the original objectives around which the problem was formulated can confuse a decision-maker. This situation can be improved with simple approaches, for example, by allowing the user to vary the color scheme according to different objectives; however, it can be difficult for the users to orientate themselves in terms of multiple objectives such that, in some cases, the trade-off can be difficult to observe. A proposal [175] designed to address this issue was to annotate the projected solutions with information such as the best and worst solution on each objective, as well as projecting samples drawn from the coordinate axes into the visualization. Further work sought to identify the edges of the original high-dimensional Pareto front approximation so that the distance from extremes in the front can be identified in a low-dimensional space [179].

D. Explaining optimizer behavior

The analysis of different optimizers is closely related to landscape analysis, regardless of the optimization problem. In this field, researchers try to decouple the effects of the optimizers' internal workings and the effect of the search landscape imposed on the algorithm. One way of doing this is by using special functions such as the f_0 function proposed in [180], a uniform random fitness function, or a constant function to assess certain behavioral patterns in algorithms by running them repeatedly and observing patterns in the distribution of the points finally found. Another way is to use a large and diverse set of benchmark functions or gradually change the properties of benchmark functions using affine combinations [181], [182].

Behavior-based benchmarks are a sophisticated means to analyze the operational dynamics of metaheuristics, especially under varying conditions. One example of such a benchmarking tool is the BIAS toolbox [183], [184]. This tool stands out by offering a behavior-centric analysis framework, enabling researchers to scrutinize whether and how different algorithms or their specific components may introduce structural bias (SB) into the optimization process. SB refers to a bias intrinsic to iterative optimization algorithms, which may drive the optimization process towards parts of the search space independently of the objective function, thus influencing the algorithm's efficiency and its outcome. Through the application of the BIAS toolbox, one can detect the presence, intensity, and nature of SB within these algorithms. A deep learning approach to detect SB was introduced in [185], where XAI techniques are used to highlight different SB patterns. Such insights are important, as they shed light on potential improvement areas, helping to refine these algorithms for enhanced performance.

For the purpose of gaining additional insights by benchmarking and analyzing algorithmic performance, the paper [186] introduces a concept termed "explainable benchmarking". Specifically, the authors propose a framework and a software package designed to dissect and analyze the performance of various optimization algorithms alongside the influences wielded by their different algorithmic components and hyper-parameters. This methodology is applied to two modular optimization frameworks, facilitating a granular analysis of the effects of various algorithmic elements and configurations on performance across many scenarios. This work uses TreeSHAP and other global and local XAI techniques to calculate and visualize the performance contributions of each of these algorithmic components and hyper-parameters, providing several insights into what drives performance on different types of objective functions. In a similar work [187], the f-ANOVA method is used to derive insights into which components of modular algorithms contribute to optimization performance. Data gathered from these experiments can then be used in combination with landscape analysis methods to derive additional insights and eventually learn the mapping between algorithm configuration, problem landscape characteristics, and performance.

Another way of explaining algorithm behavior and, more specifically, benchmarking results is by comparing many different benchmark experiments reported in the literature over the years and combining these result datasets in a unifying ontology. For this reason, the optimization algorithm benchmarking ONtology (OPTION) was proposed in [188], with an earlier similar attempt done in [189]. OPTION provides the vocabulary needed for semantic annotation of entities such as algorithms, problems, and evaluation measures. It also provides means for improved interoperability and reasoning, making these benchmark experiments much more explainable. In [190], a performance prediction model built on top of OPTION was proposed. More specifically, the authors extended the OPTION ontology with the vocabulary needed to represent modular black-box optimization algorithms. They then derived knowledge graphs with fixed-budget performance data for two modular algorithm frameworks, modCMA and modDE. On top of that, a performance prediction model was proposed using the derived knowledge graphs, leading to explainable predictions of different modular algorithm configurations.

V. Research Outlook

The list of works mentioned above is not meant to be exhaustive. As the XAI field is rapidly growing, it is likely that more studies based on EC aimed at achieving XAI will appear in the near future. For instance, we believe that ever more studies will focus on hybrid systems, e.g., combining EC-induced interpretable models and black-box models for feature extraction and low-level data manipulation. Such a combination has the potential to leverage the benefit of both areas of ML, and fully exploit the exploration capabilities that represent a unique feature of EC.

A. Challenges

One major challenge for evolutionary approaches to XAI (but faced to some extent by any XAI method) is scalability. As data continues to grow and ML models become increasingly more complex, the number of parameters and features to be optimized grows as well. On the one hand, methods that work well on small models and datasets may become too expensive on larger ones. On the other hand, large models are the most opaque and most in need of explanation, so improving the scalability of XAI methods is necessary to ensure they can be applied to even the largest models. In particular, producing fully interpretable global explanations that accurately capture model behavior while being simple enough to understand may become too challenging as models become larger – necessitating more local explanations or a more focused approach concentrating on explaining particular properties or components of the model. Here, too, we see the potential for more use of automated approaches to explainability – for example, by using evolutionary search to find local explanations of interest and optimize for particular properties. This idea has been explored with counterfactual examples, but it could be extended to other types of explanations. In any case, evolutionary ML has proposed a broad range of scaling-up mechanisms over the years [191] that, to some extent, can also be applied to EC-based XAI methods.

Another challenge for all kinds of XAI methods is the incorporation of domain knowledge. This can include knowledge from subject matter experts as well as prior knowledge about the dataset or problem. Current approaches to XAI are broad and aim to provide explanations that are independent of the problem setting or, at most, are model-specific rather than problem-specific. However, it can be useful to see how well a solution found by an ML model aligns with current knowledge in the field to evaluate the quality of the solution or, conversely, to identify areas where the model deviates from current understanding. For example, a practitioner may want to see how well the gene associations found by a genomics model align with the literature and which associations are novel. This domain knowledge can be provided in the form of expert rules, constraints, or structured data, such as a graph structure or a tree from which metrics can be defined for model evaluation. Domain knowledge can also be incorporated into the model-building process to improve interpretability, for instance, by constraining the models to focus on associations known to be plausible (e.g., by incorporating causality) or excluding irrelevant features. We believe that EC methods are particularly suited for effectively leveraging domain knowledge for building better models because of (1) their global search capacity enabling robust and complex optimization processes, (2) the possibility of hybridizing them with local search mechanisms tailored to exploit domain knowledge, and (3) their flexibility in exploration mechanisms, which provide yet more opportunities to use the domain knowledge.

B. Opportunities

We see some additional opportunities for future work employing EC for XAI. One promising direction in current research is the use of multiple objectives to optimize explanations. Explainability is inherently a multi-objective problem, requiring the explanation to be both faithful to the ML model and simple enough to be interpretable. EC is well-suited to explicitly optimizing for this. Thus, we believe introducing these ideas into current and future explanation methods can be a straightforward but effective way of improving the quality of explanations.

Along similar lines, diversity metrics and novelty search are another unique strength available to evolutionary algorithms that can help improve the explanations provided. The use of quality-diversity (illumination) algorithms can produce a range of explanations that are both accurate and provide different perspectives on the model's behavior. For example, a quality-diversity approach to counterfactual explanations could ensure that a range of behaviors are showcased in the examples. Existing work [192]–[194] has already shown some explanatory value of search space illumination for optimization problems, but there are still many opportunities to identify ways to interpret and analyze sets of solutions to better support decision-making. New approaches to visualization, interactivity, and sensitivity analysis on solutions will all add to the XAI picture for EC.

Another opportunity for EC – both in the EC for XAI and XAI for EC settings – is the incorporation of user feedback, considering the evolution of explanations as an open-ended evolution process. Explainability is intended for the human user, and, as such, explanation quality is ultimately subjective and can only be approximated by metrics. Users may also have their own unique preferences for what constitutes a useful explanation. Incorporating user feedback into the evolution process can allow better-tailored explanations that continue to improve. At the same time, better metrics measuring an explanation's quality are also necessary to avoid overwhelming the user. The design of new operators and algorithmic approaches that explicitly generate explanations as part of the search would also be an interesting direction for future EC.

C. Real-world Impacts

As AI becomes increasingly integrated into real-world applications, developing better methods for providing explanations is essential for ensuring safety and trust across various domains. With this in mind, it is also crucial to consider the practical effects and benefits that XAI research can have. We would like to highlight here a few application areas where work on evolutionary approaches to XAI can have a substantial impact.

Healthcare is a domain where the consequences of errors can be especially high. Untrusted models may be ignored by clinicians, wasting resources and providing no benefit. Even worse, seemingly trustworthy but flawed models may cause harm to patients. Even models with few errors may exhibit systematic biases, such as diagnostic models under-diagnosing certain patient groups while appearing accurate [195]. Explainability can help identify these systematic errors and biases [196]. AI models are employed in the financial sector for fraud detection and risk assessment. Similar systematic biases in these models can also be harmful, for example, by disproportionately denying loans to certain groups. In addition, regulatory bodies often require explanations for these models to ensure compliance and maintain transparency.

Explainability also holds significant potential to advance engineering and scientific discovery. AI models are used in various engineering applications, such as AI-driven materials design and drug discovery, and to produce scientific insights in fields like genomics and astrophysics. Explanations can offer insight into the underlying mechanisms and relationships, improving hypothesis generation and validating domain knowledge.

Natural language processing has experienced many recent breakthroughs, with the development and deployment of models of unprecedented size. In particular, there is an emerging paradigm of building "foundation models", generalist deep learning models that are trained on a wide range of data for general capabilities and can be further fine-tuned for downstream tasks [197]. These models can perform tasks they are not specifically trained for, but it is still unclear how they make decisions or generate outputs. Any flaws in these foundation models may be carried over to application-specific models built on top of them. As these models become more pervasive and their applications expand, understanding them and identifying their failure modes becomes increasingly important.

VI. Conclusion

We have shown that there is a strong mutual connection between XAI and EC. However, we believe that there are still several research opportunities that have not been thoroughly explored yet, which should mainly aim at: 1) devising tools, be them analytical, visual, data-driven, model-based, etc., to explain EC methods, i.e., their internal functioning, their results, and what properties/settings/instances make an algorithm suitable for achieving the result; 2) defining how solutions provided by EC methods should be checked and verified, and evaluating how much problem knowledge is actually needed to understand these solutions; and 3) fully exploiting the main features of EC methods (e.g., their exploration of "illumination" capabilities) to either provide post-hoc explanations (e.g., in the form of local explanations, or approximations of black-box models) or generate white-box models that are explainable by design. Another important challenge relates to the connection between XAI and neuroevolution (and, in general, neural architecture search): for instance, is there any link between optimized architectures and explainability? (e.g., smaller networks may be easier to explain). We consider these opportunities the basis for a potential bridge between EC and general AI (where machine/deep learning is currently mainstream) and believe that the EC community may play a fundamental role in the promising research area of XAI.

Evidently, XAI is an emerging field with important implications for AI as a whole. With the increasing use of systems built on ML and optimization in real-world applications, it is more important than ever that we understand such systems and what they learn. EC is well-poised to contribute to the field, bringing a rich toolbox of tools for performing black-box optimization. In this paper, we introduced various paradigms for explaining an ML model and the current methods of doing so. We then discussed how EC can fit into these paradigms and the advantages of employing it. In particular, EC as an optimizer is well suited for tricky interpretability metrics that are difficult to handle due to reasons such as non-differentiability, as well as for population-based metrics such as diversity, and for optimizing a multitude of these metrics at the same time. We highlighted a few methods in each category that leverage some of these strengths. However, there is still significant room for more exploration and more advanced evolutionary algorithms.

To conclude, much knowledge remains locked away within trained models that we still do not have the means to decipher. The use of EC for XAI is still uncommon, but there are many opportunities ripe for the picking, and we believe that it has the potential to play a key part in the future of XAI.

References

[1] Z.-G. Chen, Z.-H. Zhan, S. Kwong, and J. Zhang, "Evolutionary computation for intelligent transportation in smart cities: A survey," *IEEE Computational Intelligence Magazine*, vol. 17, no. 2, pp. 83–102, 2022.

[2] W. J. Murdoch, C. Singh, K. Kumbier, R. Abbasi-Asl, and B. Yu, "Interpretable machine learning: definitions, methods, and applications," *Proceedings of the National Academy of Sciences*, vol. 116, no. 44, pp. 22 071–22 080, 2019.

[3] J. Bacardit, A. E. I. Brownlee, S. Cagnoni, G. Iacca, J. McCall, and D. Walker, "The intersection of evolutionary computation and explainable AI," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, ser. GECCO '22. New York, NY, USA: Association for Computing Machinery, Jul. 2022, pp. 1757–1762.

[4] R. Zhou and T. Hu, "Evolutionary Approaches to Explainable Machine Learning," in *Handbook of Evolutionary Machine Learning*, W. Banzhaf, P. Machado, and M. Zhang, Eds. Singapore: Springer Nature, 2024, pp. 487–506.

[5] J. Lehman, J. Clune, and D. Misevic, "The surprising creativity of digital evolution: A collection of anecdotes from the evolutionary computation and artificial life research communities," pp. 274–306, 2020, arXiv:1803.03453.

[6] M. Virgolin, A. De Lorenzo, F. Randone, E. Medvet, and M. Wahde, "Model learning with personalized interpretability estimation (ML-PIE)," 2021, arXiv:2104.06060.

[7] T. Miller, "Explanation in artificial intelligence: Insights from the social sciences," *Artificial intelligence*, vol. 267, pp. 1–38, 2019.

[8] C. Ginet, "In defense of a non-causal account of reasons explanations," *Journal of Ethics*, vol. 12, pp. 229–237, 2008.

[9] H. Veatch, "Carl G. Hempel Aspects of Scientific Explanation and Other Essays in the Philosophy of Science. New York: The Free Press, 1965. 505 pp." *Philosophy of Science*, vol. 37, no. 2, pp. 312–314, Jun. 1970.

[10] D. Slack, S. Krishna, H. Lakkaraju, and S. Singh, "Explaining machine learning models with interactive natural language conversations using TalkToModel," *Nature Machine Intelligence*, vol. 5, no. 8, pp. 873–883, Aug. 2023.

[11] Z. C. Lipton, "The mythos of model interpretability," *Commun. ACM*, vol. 61, no. 10, pp. 36–43, 2018.

[12] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature Machine Intelligence*, vol. 1, pp. 206–215, 2019.

[13] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins *et al.*, "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI," *Information Fusion*, vol. 58, pp. 82–115, 2020.

[14] T. Hu, "Genetic Programming for Interpretable and Explainable Machine Learning," in *Genetic Programming Theory and Practice XIX*, L. Trujillo, S. M. Winkler, S. Silva, and W. Banzhaf, Eds. Singapore: Springer Nature, 2023, pp. 81–90.

[15] P. Barceló, M. Monet, J. Pérez, and B. Subercaseaux, "Model Interpretability through the lens of Computational Complexity," in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., 2020, pp. 15 487–15 498.

[16] G. E. Hinton and D. Van Camp, "Keeping the neural networks simple by minimizing the description length of the weights," in *Proceedings of the sixth annual conference on Computational learning theory*, 1993, pp. 5–13.

[17] B. Rylander, T. Soule, and J. Foster, "Computational complexity, genetic programming, and implications," in *Genetic Programming: 4th European Conference, EuroGP 2001 Lake Como, Italy, April 18–20, 2001 Proceedings 4*. Springer, 2001, pp. 348–360.

[18] R. Dwivedi, D. Dave, H. Naik, S. Singhal, R. Omer, P. Patel, B. Qian, Z. Wen, T. Shah, G. Morgan, and R. Ranjan, "Explainable AI (XAI): Core Ideas, Techniques, and Solutions," *ACM Computing Surveys*, vol. 55, no. 9, pp. 194:1–194:33, Jan. 2023.

[19] W. Saeed and C. Omlin, "Explainable AI (XAI): A systematic meta-survey of current challenges and future opportunities," *Knowledge-Based Systems*, vol. 263, p. 110273, Mar. 2023.

[20] J. Bacardit, "Pittsburgh genetics-based machine learning in the data mining era: Representations, generalization, and run-time," Ph.D. dissertation, Ramon Llull University, Barcelona, Spain, 2004.

[21] J. Bacardit, E. K. Burke, and N. Krasnogor, "Improving the scalability of rule-based evolutionary learning," *Memetic Computing*, vol. 1, pp. 55–67, 2009.

[22] S. W. Wilson, "Classifier fitness based on accuracy," *Evolutionary computation*, vol. 3, no. 2, pp. 149–175, 1995.

[23] J. H. Holland, L. B. Booker, M. Colombetti, M. Dorigo, D. E. Goldberg, S. Forrest, R. L. Riolo, R. E. Smith, P. L. Lanzi, W. Stolzmann *et al.*, "What is a learning classifier system?" in *International Workshop on Learning Classifier Systems*. Cham: Springer, 1999, pp. 3–32.

[24] L. Bull, "Learning classifier systems: A brief introduction," in *Applications of learning classifier systems*. Cham: Springer, 2004, pp. 1–12.

[25] L. Bull and T. Kovacs, "Foundations of learning classifier systems: An introduction," in *Foundations of Learning Classifier Systems*. Cham: Springer, 2005, pp. 1–17.

[26] R. J. Urbanowicz and J. H. Moore, "Exstracs 2.0: description and evaluation of a scalable learning classifier system," *Evolutionary intelligence*, vol. 8, no. 2, pp. 89–116, 2015.

[27] E. Bernadó-Mansilla and J. M. Garrell-Guiu, "Accuracy-based learning classifier systems: Models, analysis and applications to classification tasks." *Evolutionary Computation*, vol. 11, no. 3, pp. 209–238, 2003.

[28] X. Llora and J. M. Garrell, "Evolution of decision trees," in *Catalan Conference on Artificial Intelligence*. ACIA Press, 2001, pp. 115–122.

[29] M. V. Butz, "Kernel-based, ellipsoidal conditions in the real-valued XCS classifier system," in *Proc. Genetic Evol. Comput. Conf., GECCO 2005*. New York, NY, USA: ACM, 2005, pp. 1835–1842.

[30] W. La Cava, P. Orzechowski, B. Burlacu, F. O. de França, M. Virgolin, Y. Jin, M. Kommenda, and J. H. Moore, "Contemporary symbolic regression methods and their relative performance," *arXiv preprint arXiv:2107.14351*, 2021.

[31] W. B. Langdon, "Fitness causes bloat in variable size representations," University of Birmingham, School of Computer Science, Tech. Rep. CSRP-97-14, 1997, position paper at the Workshop on Evolutionary Computation with Variable Size Representation at ICGA-97.

[32] J. Bacardit and J. M. Garrell, "Bloat control and generalization pressure using the minimum description length principle for a pittsburgh approach learning classifier system," in *Learning Classifier Systems, Revised Selected Papers of the International Workshop on Learning Classifier Systems 2003-2005*. Springer-Verlag, LNCS 4399, 2007, pp. 59–79.

[33] X. Llora, D. E. Goldberg, I. Traus, and E. Bernadó, "Accuracy, parsimony, and generality in evolutionary learning systems via multi-objective selection," in *Learning Classifier Systems: 5th International Workshop, IWLCS 2002, Granada, Spain, September 7-8, 2002. Revised Papers 5*. Springer, 2003, pp. 118–142.

[34] J. Bacardit and N. Krasnogor, "Performance and efficiency of memetic pittsburgh learning classifier systems," *Evolutionary Computation Journal*, vol. 17, no. 3, p. in press, 2009.

[35] D. A. Calian and J. Bacardit, "Integrating memetic search into the biohel evolutionary learning system for large-scale datasets," *Memetic Computing*, vol. 5, pp. 95–130, 2013.

[36] M. A. Franco, N. Krasnogor, and J. Bacardit, "Post-processing operators for decision lists," in *Proceedings of the 14th annual conference on Genetic and evolutionary computation*, 2012, pp. 847–854.

[37] M. Butz, P. Lanzi, and S. Wilson, "Function approximation with XCS: Hyperellipsoidal conditions, recursive least squares, and compaction," *Evolutionary Computation, IEEE Transactions on*, vol. 12, no. 3, pp. 355–376, 2008.

[38] S. W. Wilson, "Compact rulesets from XCSI," in *Revised Papers from the 4th International Workshop on Advances in Learning Classifier Systems*. Springer-Verlag, 2002, pp. 197–210.

[39] H. Shang, J.-L. Wu, W. Hong, and C. Qian, "Neural Network Pruning by Cooperative Coevolution," May 2022.

[40] R. Zhou and T. Hu, "Evolving Better Initializations For Neural Networks With Pruning," in *Proceedings of the Companion Conference on Genetic and Evolutionary Computation*, ser. GECCO '23 Companion. New York, NY, USA: Association for Computing Machinery, Jul. 2023, pp. 703–706.

[41] D. Hein, S. Udluft, and T. A. Runkler, "Interpretable policies for reinforcement learning by genetic programming," *Engineering Applications of Artificial Intelligence*, vol. 76, pp. 158–169, 2018.

[42] L. L. Custode and G. Iacca, "Evolutionary learning of interpretable decision trees," 2020, arXiv:2012.07723.

[43] ——, "A co-evolutionary approach to interpretable reinforcement learning in environments with continuous action spaces," in *Symposium Series on Computational Intelligence (SSCI)*. New York, NY, USA: IEEE, 2021, pp. 1–8.

[44] ——, "Interpretable AI for policy-making in pandemics," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, ser. GECCO '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 1763–1769.

[45] ——, "Interpretable pipelines with evolutionary optimized modules for reinforcement learning tasks with visual inputs," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2022, pp. 224–227.

[46] ——, "Social interpretable reinforcement learning," *arXiv preprint arXiv:2401.15480*, 2024.

[47] A. Ferigo, L. L. Custode, and G. Iacca, "Quality diversity evolutionary learning of decision trees," in *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*, 2023, pp. 425–432.

[48] ——, "Quality–diversity optimization of decision trees for interpretable reinforcement learning," *Neural Computing and Applications*, pp. 1–12, 2023.

[49] M. Crespi, A. Ferigo, L. L. Custode, and G. Iacca, "A population-based approach for multi-agent interpretable reinforcement learning," *Applied Soft Computing*, vol. 147, p. 110758, 2023.

[50] M. Galende, G. Sainz, and M. J. Fuente, "Accuracy-interpretability balancing in fuzzy models based on multiobjective genetic algorithm," in *2009 European Control Conference (ECC)*. New York, NY, USA: IEEE, 2009, pp. 3915–3920.

[51] M. Virgolin, E. Medvet, T. Alderliesten, and P. A. Bosman, "Less is more: A call to focus on simpler models in genetic programming for interpretable machine learning," 2022.

[52] W. La Cava and J. H. Moore, "Genetic programming approaches to learning fair classifiers," in *Genetic and Evolutionary Computation Conference*. New York, NY, USA: ACM, 2020, pp. 967–975.

[53] R. J. Urbanowicz, A. Granizo-Mackenzie, and J. H. Moore, "An analysis pipeline with statistical and visualization-guided knowledge discovery for michigan-style learning classifier systems," *IEEE computational intelligence magazine*, vol. 7, no. 4, pp. 35–45, 2012.

[54] Y. Liu, W. N. Browne, and B. Xue, "Visualizations for rule-based machine learning," *Natural Computing*, vol. 1, pp. 1–22, 2021.

[55] K. Pearson, "LIII. On lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, Nov. 1901.

[56] H. Hotelling, "Analysis of a complex of statistical variables into principal components," *Journal of Educational Psychology*, vol. 24, pp. 417–441, 1933.

[57] L. van der Maaten and G. Hinton, "Visualizing Data using t-SNE," *Journal of Machine Learning Research*, vol. 9, no. 86, pp. 2579–2605, 2008.

[58] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *kdd*, vol. 96, 1996, pp. 226–231.

[59] A. Lensen, B. Xue, and M. Zhang, "Genetic programming for evolving a front of interpretable models for data visualization," *IEEE Transactions on Cybernetics*, vol. 51, no. 11, pp. 5468–5482, 2021.

[60] F. Schofield and A. Lensen, "Using Genetic Programming to Find Functional Mappings for UMAP Embeddings," in *2021 IEEE Congress on Evolutionary Computation (CEC)*, Jun. 2021, pp. 704–711.

[61] I. Icke and A. Rosenberg, "Multi-objective Genetic Programming for Visual Analytics," in *Genetic Programming*, ser. Lecture Notes in Computer Science, S. Silva, J. A. Foster, M. Nicolau, P. Machado, and M. Giacobini, Eds. Berlin, Heidelberg: Springer, 2011, pp. 322–334.

[62] A. Cano, S. Ventura, and K. J. Cios, "Multi-objective genetic programming for feature extraction and data visualization," *Soft Computing*, vol. 21, no. 8, pp. 2069–2089, Apr. 2017.

[63] A. Lensen, B. Xue, and M. Zhang, "Genetic programming for manifold learning: Preserving local topology," *IEEE Transactions on Evolutionary Computation*, vol. early access, 2021.

[64] S. Sayed, M. Nassef, A. Badr, and I. Farag, "A Nested Genetic Algorithm for feature selection in high-dimensional cancer Microarray datasets," *Expert Systems with Applications*, vol. 121, pp. 233–243, May 2019.

[65] Z. Sha, T. Hu, and Y. Chen, "Feature Selection for Polygenic Risk Scores using Genetic Algorithm and Network Science," in *2021 IEEE Congress on Evolutionary Computation (CEC)*, Jun. 2021, pp. 802–808.

[66] Y. Xue, Y. Tang, X. Xu, J. Liang, and F. Neri, "Multi-Objective Feature Selection With Missing Data in Classification," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 6, no. 2, pp. 355–364, Apr. 2022.

[67] T. Hu, "Can genetic programming perform explainable machine learning for bioinformatics?" in *Genetic Programming Theory and Practice XVII*. Springer, 2020.

[68] T. Hu, K. Oksanen, W. Zhang, E. Randell, A. Furey, G. Sun, and G. Zhai, "An evolutionary learning and network approach to identifying key metabolites for osteoarthritis," *PLoS Computational Biology*, vol. 14, no. 3, p. e1005986, 2018.

[69] C. Sha, M. Cuperlovic-Culf, and T. Hu, "SMILE: systems metabolomics using interpretable learning and evolution," *BMC Bioinformatics*, vol. 22, p. 284, 2021.

[70] B. Xue, M. Zhang, W. N. Browne, and X. Yao, "A Survey on Evolutionary Computation Approaches to Feature Selection," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 4, pp. 606–626, Aug. 2016.

[71] Y. Xue, B. Xue, and M. Zhang, "Self-adaptive particle swarm optimization for large-scale feature selection in classification," *ACM Trans. Knowl. Discov. Data*, vol. 13, no. 5, pp. 50:1–50:27, 2019.

[72] B. H. Nguyen, B. Xue, and M. Zhang, "A survey on swarm intelligence approaches to feature selection in data mining," *Swarm and Evolutionary Computation*, vol. 54, p. 100663, May 2020.

[73] E. Hancer, B. Xue, and M. Zhang, "A survey on feature selection approaches for clustering," *Artificial Intelligence Review*, vol. 53, no. 6, pp. 4519–4545, 2020.

[74] W. La Cava and J. H. Moore, "Learning feature spaces for regression with genetic programming," *Genetic Programming and Evolvable Machines*, vol. 21, no. 3, pp. 433–467, Sep. 2020.

[75] Z. Li, J. He, X. Zhang, H. Fu, and J. Qin, "Toward high accuracy and visualization: An interpretable feature extraction method based on genetic programming and non-overlap degree," in *Proceedings of the IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 2020, pp. 299–304.

[76] M. Muharram and G. Smith, "Evolutionary constructive induction," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 11, pp. 1518–1528, Nov. 2005.

[77] M. Virgolin, T. Alderliesten, and P. A. N. Bosman, "On explaining machine learning models by evolving crucial and compact features," *Swarm and Evolutionary Computation*, vol. 53, p. 100640, Mar. 2020.

[78] T. Uriot, M. Virgolin, T. Alderliesten, and P. A. N. Bosman, "On genetic programming representations and fitness functions for interpretable dimensionality reduction," in *Proceedings of the Genetic and Evolutionary Computation Conference*, ser. GECCO '22. New York, NY, USA: Association for Computing Machinery, Jul. 2022, pp. 458–466.

[79] L. Breiman, "Random Forest," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[80] J. H. Friedman, "Multivariate adaptive regression splines," *The annals of statistics*, vol. 19, no. 1, pp. 1–67, 1991.

[81] A. Fisher, C. Rudin, and F. Dominici, "All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously," *Journal of Machine Learning Research*, vol. 20, no. 177, pp. 1–81, 2019.

[82] J. Robertson, C. Stinson, and T. Hu, "A Bio-Inspired Framework for Machine Bias Interpretation," in *Proceedings of the 2022 AAAI/ACM Conference on AI, Ethics, and Society*, ser. AIES '22. New York, NY, USA: Association for Computing Machinery, Jul. 2022, pp. 588–598.

[83] C. Buciluă, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '06. New York, NY, USA: Association for Computing Machinery, Aug. 2006, pp. 535–541.

[84] G. E. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *CoRR*, vol. abs/1503.02531, 2015.

[85] H. Lakkaraju, E. Kamar, R. Caruana, and J. Leskovec, "Interpretable & explorable approximations of black box models," *CoRR*, vol. abs/1707.01154, 2017.

[86] B. P. Evans, B. Xue, and M. Zhang, "What's inside the black box? a genetic programming method for interpreting complex machine learning models," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 2019, pp. 1012–1020.

[87] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[88] J. Bacardit, J. D. Hirst, M. Stout, J. Blazewicz, and N. Krasnogor, "Coordination number prediction using learning classifier systems: Performance and interpretability," in *Genetic and Evolutionary Computation Conference*. New York, NY, USA: ACM Press, 2006, pp. 247–254.

[89] N. Lazzarini, P. Widera, S. Williamson, R. Heer, N. Krasnogor, and J. Bacardit, "Functional networks inference from rule-based machine learning models," *BioData mining*, vol. 9, no. 1, pp. 1–23, 2016.

[90] G. W. Bassel, E. Glaab, J. Marquez, M. J. Holdsworth, and J. Bacardit, "Functional network construction in arabidopsis using rule-based machine learning on large-scale data sets," *The Plant Cell Online*, vol. 23, no. 9, pp. 3101–3116, Sep. 2011.

[91] S. Baron, N. Lazzarini, and J. Bacardit, "Characterising the influence of rule-based knowledge representations in biological knowledge extraction from transcriptomics data," in *European Conference on the

Applications of Evolutionary Computation. Cham: Springer, 2017, pp. 125–141.

[92] H. D. Mettler, M. Schmidt, W. Senn, M. A. Petrovici, and J. Jordan, "Evolving neuronal plasticity rules using cartesian genetic programming," 2021, arXiv:2102.04312.

[93] J. Jordan, M. Schmidt, W. Senn, and M. A. Petrovici, "Evolving to learn: discovering interpretable plasticity rules for spiking networks," 2020, arXiv:2005.14149.

[94] A. Yaman, G. Iacca, D. C. Mocanu, M. Coler, G. Fletcher, and M. Pechenizkiy, "Evolving plasticity for autonomous learning under changing environmental conditions," *Evolutionary computation*, vol. 29, no. 3, pp. 391–414, 2021.

[95] Y. Tang, D. Nguyen, and D. Ha, "Neuroevolution of self-interpretable agents," in *Genetic and Evolutionary Computation Conference*. New York, NY, USA: ACM, 2020, pp. 414–424.

[96] B. Wang, W. Pei, B. Xue, and M. Zhang, "A Multi-objective Genetic Algorithm to Evolving Local Interpretable Model-agnostic Explanations for Deep Neural Networks in Image Classification," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2022.

[97] W. Samek, G. Montavon, S. Lapuschkin, C. J. Anders, and K.-R. Müller, "Explaining deep neural networks and beyond: A review of methods and applications," *Proc. IEEE*, vol. 109, no. 3, pp. 247–278, 2021.

[98] T. Adel, Z. Ghahramani, and A. Weller, "Discovering Interpretable Representations for Both Deep Generative and Discriminative Models," in *Proceedings of the 35th International Conference on Machine Learning*. PMLR, Jul. 2018, pp. 50–59.

[99] N. Cammarata, S. Carter, G. Goh, C. Olah, M. Petrov, L. Schubert, C. Voss, B. Egan, and S. K. Lim, "Thread: Circuits," *Distill*, 2020.

[100] N. Elhage, N. Nanda, C. Olsson, T. Henighan, N. Joseph, B. Mann, A. Askell, Y. Bai, A. Chen, T. Conerly, N. DasSarma, D. Drain, D. Ganguli, Z. Hatfield-Dodds, D. Hernandez, A. Jones, J. Kernion, L. Lovitt, K. Ndousse, D. Amodei, T. Brown, J. Clark, J. Kaplan, S. McCandlish, and C. Olah, "A mathematical framework for transformer circuits," *Transformer Circuits Thread*, 2021.

[101] N. Nanda, L. Chan, T. Lieberum, J. Smith, and J. Steinhardt, "Progress measures for grokking via mechanistic interpretability," Oct. 2023.

[102] M. Saletta and C. Ferretti, "A Grammar-based Evolutionary Approach for Assessing Deep Neural Source Code Classifiers," in *2022 IEEE Congress on Evolutionary Computation (CEC)*, Jul. 2022, pp. 1–8.

[103] H. Bradley, H. Fan, T. Galanos, R. Zhou, D. Scott, and J. Lehman, "The openelm library: Leveraging progress in language models for novel evolutionary algorithms," in *Genetic Programming Theory and Practice XX*. Springer, 2024, pp. 177–201.

[104] S. J. Wetzel, "Closed-Form Interpretation of Neural Network Classifiers with Symbolic Regression Gradients," Jan. 2024.

[105] M. T. Ribeiro, S. Singh, and C. Guestrin, ""Why should I trust you?": Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1135–1144.

[106] L. A. Ferreira, F. G. Guimarães, and R. Silva, "Applying genetic programming to improve interpretability in machine learning models," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, 2020, pp. 1–8.

[107] R. Guidotti, A. Monreale, S. Ruggieri, D. Pedreschi, F. Turini, and F. Giannotti, "Local Rule-Based Explanations of Black Box Decision Systems," *CoRR*, vol. abs/1805.10820, 2018.

[108] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS)*, 2017, pp. 4768–4777.

[109] S. Wachter, B. Mittelstadt, and C. Russell, "Counterfactual explanations without opening the black box: Automated decisions and the GDPR," *Harv. JL & Tech.*, vol. 31, p. 841, 2017.

[110] A.-H. Karimi, G. Barthe, B. Schölkopf, and I. Valera, "A survey of algorithmic recourse: Definitions, formulations, solutions, and prospects," *CoRR*, vol. abs/2010.04050, 2020.

[111] R. K. Mothilal, A. Sharma, and C. Tan, "Explaining machine learning classifiers through diverse counterfactual examples," in *ACM Conference on Fairness, Accountability, and Transparency*, Jan. 2020.

[112] A. Kulesza and B. Taskar, "Determinantal point processes for machine learning," *Found. Trends Mach. Learn.*, vol. 5, no. 2-3, pp. 123–286, 2012.

[113] S. Sharma, J. Henderson, and J. Ghosh, "CERTIFAI: A Common Framework to Provide Explanations and Analyse the Fairness and Robustness of Black-box Models," in *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. New York NY USA: ACM, Feb. 2020, pp. 166–172.

[114] M. Schleich, Z. Geng, Y. Zhang, and D. Suciu, "GeCo: Quality counterfactual explanations in real time," *Proceedings of the VLDB Endowment*, vol. 14, no. 9, pp. 1681–1693, May 2021.

[115] S. Dandl, C. Molnar, M. Binder, and B. Bischl, "Multi-Objective Counterfactual Explanations," in *Parallel Problem Solving from Nature – PPSN XVI*, ser. Lecture Notes in Computer Science, T. Bäck, M. Preuss, A. Deutz, H. Wang, C. Doerr, M. Emmerich, and H. Trautmann, Eds. Cham: Springer International Publishing, 2020, pp. 448–469.

[116] R. Li, M. T. M. Emmerich, J. Eggermont, T. Bäck, M. Schütz, J. Dijkstra, and J. H. C. Reiber, "Mixed integer evolution strategies for parameter optimization," *Evolutionary Computation*, vol. 21, no. 1, pp. 29–64, 2013.

[117] C. Molnar, *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*. leanpub.com, 2022.

[118] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Trans. Evol. Comput.*, vol. 23, no. 5, pp. 828–841, 2019.

[119] H. Qiu, L. L. Custode, and G. Iacca, "Black-box adversarial attacks using evolution strategies," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2021, pp. 1827–1833.

[120] L. L. Custode and G. Iacca, "One run to attack them all: finding simultaneously multiple targeted adversarial perturbations," in *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2021, pp. 01–08.

[121] M. Alzantot, Y. Sharma, A. Elgohary, B.-J. Ho, M. B. Srivastava, and K.-W. Chang, "Generating natural language adversarial examples," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, Eds. Association for Computational Linguistics, 2018, pp. 2890–2896.

[122] W. Huang, X. Zhao, G. Jin, and X. Huang, "SAFARI: Versatile and efficient evaluations for robustness of interpretability," *CoRR*, vol. abs/2208.09418, 2022.

[123] S. V. Tamam, R. Lapid, and M. Sipper, "Foiling explanations in deep neural networks," *CoRR*, vol. abs/2211.14860, 2022.

[124] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multiobjective optimization," in *Evolutionary multiobjective optimization: theoretical advances and applications*. Springer, 2005, pp. 105–145.

[125] D. J. Walker, "Visualisation with treemaps and sunbursts in many-objective optimisation," *Genetic Programming and Evolvable Machines*, vol. 19, no. 3, pp. 421–452, 2018.

[126] E. Medvet, M. Virgolin, M. Castelli, P. A. Bosman, I. Gonçalves, and T. Tušar, "Unveiling evolutionary algorithm representation with du maps," *Genetic Programming and Evolvable Machines*, vol. 19, no. 3, pp. 351–389, 2018.

[127] M. Chromik and M. Schuessler, "A Taxonomy for Human Subject Evaluation of Black-Box Explanations in XAI." *Exss-atec@ iui*, vol. 94, 2020.

[128] Q. V. Liao, D. Gruen, and S. Miller, "Questioning the ai: informing design practices for explainable ai user experiences," in *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, 2020, pp. 1–15.

[129] D. Gunning, E. Vorm, J. Y. Wang, and M. Turek, "DARPA's explainable AI (XAI) program: A retrospective," p. e61, 2021.

[130] M. Fischetti, M. Fischetti *et al.*, "Matheuristics," in *Handbook of heuristics*. Springer International Publishing, 2018, vol. 1, pp. 121–153.

[131] M. Goerigk and M. Hartisch, "A framework for inherently interpretable optimization models," *European Journal of Operational Research*, vol. 310, no. 3, pp. 1312–1324, 2023.

[132] K. Deb, "An efficient constraint handling method for genetic algorithms," *Computer methods in applied mechanics and engineering*, vol. 186, no. 2-4, pp. 311–338, 2000.

[133] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[134] T. Helmuth, L. Spector, and J. Matheson, "Solving uncompromising problems with lexicase selection," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 5, pp. 630–643, 2014.

[135] Q. Xu, Z. Xu, and T. Ma, "A survey of multiobjective evolutionary algorithms based on decomposition: variants, challenges and future directions," *IEEE Access*, vol. 8, pp. 41 588–41 614, 2020.

[136] J. G. Falcón-Cardona and C. A. C. Coello, "Indicator-based multi-objective evolutionary algorithms: A comprehensive survey," *ACM Computing Surveys (CSUR)*, vol. 53, no. 2, pp. 1–35, 2020.
[137] Y. Hua, Q. Liu, K. Hao, and Y. Jin, "A survey of evolutionary algorithms for multi-objective optimization problems with irregular pareto fronts," *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. 2, pp. 303–318, 2021.
[138] K. O. Stanley, D. B. D'Ambrosio, and J. Gauci, "A hypercube-based encoding for evolving large-scale neural networks," *Artificial life*, vol. 15, no. 2, pp. 185–212, 2009.
[139] D. B. D'Ambrosio, J. Gauci, and K. O. Stanley, "Hyperneat: The first five years," *Growing Adaptive Machines: Combining Development and Learning in Artificial Neural Networks*, pp. 159–185, 2014.
[140] T. Nyathi and N. Pillay, "Comparison of a genetic algorithm to grammatical evolution for automated design of genetic programming classification algorithms," *Expert Systems with Applications*, vol. 104, pp. 213–234, 2018.
[141] D. Whitley, F. Chicano, and B. Goldman, "Gray Box Optimization for Mk Landscapes," *Evolutionary Comput.*, vol. 24, no. 3, pp. 491–519, 2016.
[142] Z. Yang, K. Tang, and X. Yao, "Large scale evolutionary optimization using cooperative coevolution," *Information sciences*, vol. 178, no. 15, pp. 2985–2999, 2008.
[143] P. Larrañaga and J. A. Lozano, *Estimation of distribution algorithms: A new tool for evolutionary computation*. Springer Science & Business Media, 2001, vol. 2.
[144] J. Ceberio, A. Mendiburu, and J. A. Lozano. (2011, May) Estimation of distribution algorithms for permutation-based problems. Intelligent Systems Group, Department of Computer Science and Artificial Intelligence, The University of the Basque Country.
[145] M. Hauschild and M. Pelikan, "An introduction and survey of estimation of distribution algorithms," *Swarm Evol. Comput.*, vol. 1, no. 3, pp. 111 – 128, 2011.
[146] J. A. Lozano, P. Larrañaga, I. Inza, and E. Bengoetxea, *Towards a New Evolutionary Computation: Advances on Estimation of Distribution Algorithms (Studies in Fuzziness and Soft Computing)*. Springer-Verlag, 2006.
[147] K. M. Malan, "A Survey of Advances in Landscape Analysis for Optimisation," *Algorithms*, vol. 14, no. 2, p. 40, Jan. 2021.
[148] R. Trajanov, S. Dimeski, M. Popovski, P. Korošec, and T. Eftimov, "Explainable landscape-aware optimization performance prediction," in *Symposium Series on Computational Intelligence*. New York, NY, USA: IEEE, 2021, pp. 01–08.
[149] ——, "Explainable landscape analysis in automated algorithm performance prediction," 2022, arXiv:2203.11828.
[150] G. Ochoa, K. M. Malan, and C. Blum, "Search trajectory networks: A tool for analysing and visualising the behaviour of metaheuristics," *Applied Soft Computing*, vol. 109, p. 107492, 2021.
[151] M. Fyvie, J. A. W. McCall, and L. A. Christie, "Towards Explainable Metaheuristics: PCA for Trajectory Mining in Evolutionary Algorithms," in *Artificial Intelligence XXXVIII*, M. Bramer and R. Ellis, Eds. Cham: Springer International Publishing, 2021, pp. 89–102.
[152] G. Cenikj, G. Petelin, C. Doerr, P. Korošec, and T. Eftimov, "DynamoRep: Trajectory-Based Population Dynamics for Classification of Black-box Optimization Problems," in *Proceedings of the Genetic and Evolutionary Computation Conference*, ser. GECCO '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 813–821.
[153] M. J. Walter, D. J. Walker, and M. J. Craven, "An explainable visualisation of the evolutionary search process," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, ser. GECCO '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 1794–1802.
[154] A. Wallace, A. E. I. Brownlee, and D. Cairns, "Towards explaining metaheuristic solution quality by data mining surrogate fitness models for importance of variables," in *Artificial Intelligence XXXVIII*, M. Bramer and R. Ellis, Eds. Cham: Springer International Publishing, 2021, pp. 58–72.
[155] M. Singh, A. E. I. Brownlee, and D. Cairns, "Towards explainable metaheuristic: Mining surrogate fitness models for importance of variables," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, ser. GECCO '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 1785–1793.
[156] J. H. Drake, A. Kheiri, E. Özcan, and E. K. Burke, "Recent advances in selection hyper-heuristics," *European Journal of Operational Research*, vol. 285, no. 2, pp. 405–428, 2020.
[157] S. K. Smit and A. Eiben, "Parameter tuning of evolutionary algorithms: Generalist vs. specialist," in *European Conference on the Applications of Evolutionary Computation*. Berlin, Heidelberg: Springer, 2010, pp. 542–551.
[158] R. Ugolotti, L. Sani, and S. Cagnoni, "What can we learn from multi-objective meta-optimization of evolutionary algorithms in continuous domains?" *Mathematics*, vol. 7, no. 3, p. 232, 2019.
[159] N. Urquhart, "Combining parallel coords with multi-objective evolut algorithms in a real-world optimisation problem," in *Genetic and Evolutionary Computation Conference*. New York, NY, USA: ACM, 07 2017, pp. 1335–1340.
[160] J.-B. Mouret and J. Clune, "Illuminating search spaces by mapping elites," 2015.
[161] A. Gaier, A. Asteroth, and J.-B. Mouret, "Data-efficient exploration, optimization, and modeling of diverse designs through surrogate-assisted illumination," in *Genetic and Evolutionary Computation Conference*. New York, NY, USA: ACM, 2017, pp. 99–106.
[162] K. Miettinen and M. M. Mäkelä, "Interactive multiobjective optimization system www-nimbus on the internet," *Computers & Operations Research*, vol. 27, no. 7-8, pp. 709–723, 2000.
[163] N. Urquhart, M. Guckert, and S. Powers, "Increasing trust in meta-heuristics using MAP-elites," in *Genetic and Evolutionary Computation Conference - Companion*, ser. GECCO '19. New York, NY, USA: ACM, 2019, pp. 1345–1348.
[164] N. Urquhart, S. Höle, and E. Hart, "Automated, explainable rule extraction from MAP-Elites archives," in *Proc. EvoApplications 2021*, 2021, pp. 258–272.
[165] W. T and S. J, "Backbones in optimization and approximation," in *IJCAI*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001, pp. 254—259.
[166] M. Fyvie, J. A. W. McCall, L. A. Christie, A.-C. Zăvoianu, A. E. I. Brownlee, and R. Ainslie, "Explaining a Staff Rostering Problem by Mining Trajectory Variance Structures," in *Artificial Intelligence XL: 43rd SGAI International Conference on Artificial Intelligence, AI 2023, Cambridge, UK, December 12–14, 2023, Proceedings*. Berlin, Heidelberg: Springer-Verlag, 2023, p. 275–290.
[167] T. M. Banda, A.-C. Zăvoianu, A. Petrovski, D. Wöckinger, and G. Bramerdorfer, "A multi-objective evolutionary approach to discover explainability tradeoffs when using linear regression to effectively model the dynamic thermal behaviour of electrical machines," *ACM Trans. Evol. Learn. Optim.*, vol. 4, no. 1, feb 2024.
[168] K. Deb and A. Srinivasan, "Innovization: Discovery of innovative design principles through multiobjective evolutionary optimization," in *Multiobjective Problem Solving from Nature: From Concepts to Applications*, J. Knowles *et al.*, Eds. Berlin, Heidelberg: Springer, 2008, pp. 243–262.
[169] K. Deb, S. Bandaru, D. Greiner, A. Gaspar-Cunha, and C. C. Tutum, "An integrated approach to automated innovization for discovering useful design principles: Case studies from engineering," *Applied Soft Computing*, vol. 15, pp. 42–56, 2014.
[170] R. J. Scholman, A. Bouter, L. R. Dickhoff, T. Alderliesten, and P. A. Bosman, "Obtaining smoothly navigable approximation sets in bi-objective multi-modal optimization," 2022, arXiv:2203.09214.
[171] A. Inselberg and B. Dimsdale, "Parallel coordinates," *Human-Machine Interactive Systems*, pp. 199–233, 2009.
[172] N. B. Urquhart, "Evaluating the performance of an evolutionary tool for exploring solution fronts," in *Applications of Evolutionary Computation: 21st International Conference, EvoApplications 2018, Parma, Italy, April 4-6, 2018, Proceedings 21*. Springer, 2018, pp. 523–537.
[173] A. Pryke, S. Mostaghim, and A. Nazemi, "Heatmap visualization of population based multi objective algorithms," in *Evolutionary Multi-Criterion Optimization: 4th International Conference, EMO 2007, Matsushima, Japan, March 5-8, 2007. Proceedings 4*. Springer, 2007, pp. 361–375.
[174] R. J. Lygoe, M. Cary, and P. J. Fleming, "A real-world application of a many-objective optimisation complexity reduction process," in *Evolutionary Multi-Criterion Optimization*, R. C. Purshouse, P. J. Fleming, C. M. Fonseca, S. Greco, and J. Shaw, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 641–655.
[175] D. J. Walker, R. Everson, and J. E. Fieldsend, "Visualizing mutually nondominating solution sets in many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 2, pp. 165–184, 2013.
[176] I. Jolliffe, *Principal Component Analysis*. Springer, 2002.
[177] T. Kohonen, *Self-Organising Maps*. Springer, 1995.
[178] J. W. Sammon, "A nonlinear mapping for data structure analysis," *IEEE Transactions on computers*, vol. 100, no. 5, pp. 401–409, 1969.

[179] R. M. Everson, D. J. Walker, and J. E. Fieldsend, “Life on the Edge: Characterising the Edges of Mutually Non-dominating Sets,” *Evolutionary Computation*, vol. 22, no. 3, pp. 479–501, 09 2014.

[180] A. V. Kononova, D. W. Corne, P. De Wilde, V. Shneer, and F. Caraffini, “Structural bias in population-based algorithms,” *Information Sciences*, vol. 298, pp. 468–490, 2015.

[181] D. Vermetten, F. Ye, and C. Doerr, “Using affine combinations of bbob problems for performance assessment,” in *Proceedings of the Genetic and Evolutionary Computation Conference*, ser. GECCO ’23. New York, NY, USA: Association for Computing Machinery, 2023, p. 873–881.

[182] D. Vermetten, F. Ye, T. Bäck, and C. Doerr, “Ma-bbob: Many-affine combinations of bbob functions for evaluating automl approaches in noiseless numerical black-box optimization contexts,” in *International Conference on Automated Machine Learning*. PMLR, 2023, pp. 7–1.

[183] D. Vermetten, B. van Stein, F. Caraffini, L. L. Minku, and A. V. Kononova, “Bias: A toolbox for benchmarking structural bias in the continuous domain,” *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 6, pp. 1380–1393, 2022.

[184] D. Vermetten, F. Caraffini, B. van Stein, and A. V. Kononova, “Using structural bias to analyse the behaviour of modular cma-es,” in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2022, pp. 1674–1682.

[185] B. Van Stein, D. Vermetten, F. Caraffini, and A. V. Kononova, “Deep bias: Detecting structural bias using explainable ai,” in *Proceedings of the Companion Conference on Genetic and Evolutionary Computation*, 2023, pp. 455–458.

[186] N. van Stein, D. Vermetten, A. V. Kononova, and T. Bäck, “Explainable benchmarking for iterative optimization heuristics,” *arXiv preprint arXiv:2401.17842*, 2024.

[187] A. Nikolikj, A. Kostovska, D. Vermetten, C. Doerr, and T. Eftimov, “Quantifying individual and joint module impact in modular optimization frameworks,” 2024, arXiv:2405.11964.

[188] A. Kostovska, D. Vermetten, C. Doerr, S. Džeroski, P. Panov, and T. Eftimov, “Option: optimization algorithm benchmarking ontology,” in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2021, pp. 239–240.

[189] A. Yaman, A. Hallawa, M. Coler, and G. Iacca, “Presenting the eco: evolutionary computation ontology,” in *European Conference on the Applications of Evolutionary Computation*. Springer, Cham, 2017, pp. 603–619.

[190] A. Kostovska, D. Vermetten, S. Džeroski, P. Panov, T. Eftimov, and C. Doerr, “Using knowledge graphs for performance prediction of modular optimization algorithms,” in *Applications of Evolutionary Computation*, J. Correia, S. Smith, and R. Qaddoura, Eds. Cham: Springer Nature Switzerland, 2023, pp. 253–268.

[191] J. Bacardit and X. Llorà, “Large-scale data mining using genetics-based machine learning,” *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 3, no. 1, pp. 37–61, 2013.

[192] N. Urquhart, S. Höhl, and E. Hart, “Automated, explainable rule extraction from MAP-elites archives,” in *Applications of Evolutionary Computation: 24th International Conference, EvoApplications 2021, Held as Part of EvoStar 2021, Virtual Event, April 7–9, 2021, Proceedings 24*. Springer, 2021, pp. 258–272.

[193] S. P. Walton, B. J. Evans, A. A. M. Rahat, J. Stovold, and J. Vincalek, “Does mapping elites illuminate search spaces? A large-scale user study of MAP–Elites applied to human–AI collaborative design,” 2024.

[194] H. Zhang, Q. Chen, B. Xue, W. Banzhaf, and M. Zhang, “MAP-Elites for Genetic Programming-Based Ensemble Learning: An Interactive Approach [AI-eXplained],” *IEEE Computational Intelligence Magazine*, vol. 18, no. 4, pp. 62–63, 2023.

[195] L. Seyyed-Kalantari, H. Zhang, M. B. A. McDermott, I. Y. Chen, and M. Ghassemi, “Underdiagnosis bias of artificial intelligence algorithms applied to chest radiographs in under-served patient populations,” *Nature Medicine*, vol. 27, no. 12, pp. 2176–2182, 2021.

[196] A. Arias-Duart, F. Parés, D. Garcia-Gasulla, and V. Gimenez-Abalos, “Focus! Rating XAI methods and finding biases,” in *IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2022, Padua, Italy, July 18-23, 2022*. IEEE, 2022, pp. 1–8.

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