Optimization and supervised learning for decision making: competitors or partners?

A. Mor[1], *C. Orsenigo*[1], *M.G. Speranza*[2]
[1] *Department of Management, Economics and Industrial Engineering*
Politecnico di Milano, Italy
{andrea.mor,carlotta.orsenigo}@polimi.it
[2] *Department of Economics and Management*
University of Brescia, Italy
grazia.speranza@unibs.it

Abstract

Machine learning (ML) is a constantly growing research area. While optimization researchers are trying to understand how to take advantage of ML methods, ML researchers are developing algorithms to solve classical optimization problems. Given the huge impact of ML in a large variety of disciplines and application domains, an underlying discussion topic concerns the possibility that ML may become capable of replacing optimization. As ML is a very broad research area, we focus on supervised learning, one of the most prominent approaches. In this paper we discuss the different roles of optimization and supervised learning in decision making processes, provide a high level overview of the ways in which each can enrich and benefit the other, and highlight research directions for an impactful interaction of the two paradigms.

1 Introduction

Decision making can be seen as the process of selecting the best course of action among multiple alternatives to achieve a desired outcome. Effective decision making requires a thorough understanding of the problem, the identification of potential solutions, their evaluation, and the implementation of the selected one.

Operational Research (OR) is the discipline that employs mathematical models, analytics, and algorithms to enable effective decision making. OR arose during World War II, although methods that can be classified as belonging to OR had been invented much earlier. While the OR toolbox includes, besides optimization, other classes of methods, such as queuing models and simulation (Wagner [1975], Winston [2004], Hillier and Lieberman [2015]), optimization is the most representative class of OR methods. Because of its prominence in the OR field, this discussion paper will focus specifically on optimization.

Machine Learning (ML) is a subfield of Artificial Intelligence (AI) devoted to the study of algorithms that allow systems to automatically improve their performance over time through experience, that is, by learning from data, without being explicitly programmed. As a research area, ML progressed dramatically in the last few decades due to notable methodological improvements (deep learning, in particular), hardware advances, and the exponential growth of available data (see, for instance, Dean [2022]). Moreover, ML strongly emerged as the method of choice in a variety of AI-related contexts, ranging from computer vision and speech recognition to robot control (see Voulodimos et al. [2018], Nassif et al. [2019], Soori et al. [2023], respectively). The field of ML offers a variety of learning setups, depending on the task addressed. In this paper, we will focus on supervised learning, which underpins many practical applications and profoundly impacted numerous industries by exhibiting clear and substantial benefits.

While OR researchers are engaged in trying to understand how to take advantage of ML to improve optimization methods (see, for example, Bengio et al. [2021]), ML researchers are drawing on tools from the optimization toolkit to enhance the performance of ML techniques (see, for example, Piccialli and Sciandrone [2022]). As ML methods continue to evolve and mature, an underlying discussion topic emerges: could ML one day replace traditional optimization? To contribute to this debate, in this paper we discuss the distinct yet interconnected roles that optimization and supervised learning play in decision making processes, focusing on their respective core components and goals. Moreover, we describe ongoing research efforts that leverage one paradigm to enhance the other, highlighting how their interplay can give rise to novel decision-support frameworks. Finally, we discuss the potential use of supervised learning as an optimization algorithm.

The paper is organized as follows. In Section 2 we summarize the main characteristics and different roles of optimization and supervised learning in decision making processes. Section 3 is focused on how optimization can benefit from supervised learning and vice-versa, and on how the two paradigms can interact. The potential use of supervised learning as an optimization algorithm is discussed in Section 4. In Section 5 we draw some conclusions and discuss future research directions.

2 Optimization and supervised learning in decision making

In this section, we summarize the different core components and goals of optimization and supervised learning. To clarify the concepts, we exemplify them with two applications, one in optimization and one in supervised learning.

2.1 Optimization

An optimization-based approach is a process that follows a problem-model-solution scheme. Initially, a real-world problem is analyzed and its most relevant elements are captured in an optimization model. Then, an algorithm, whether exact or heuristic, must be designed to solve the model, if a suitable solution method is not already available through commercial or open-source software. A solution is obtained when the model is fed with the data of a specific problem instance. This solution can be implemented directly or used to support the decision making process. A scheme of the process that starts from the real-world problem and ends with an output of the algorithm, that is, a solution, is shown in Figure 1.

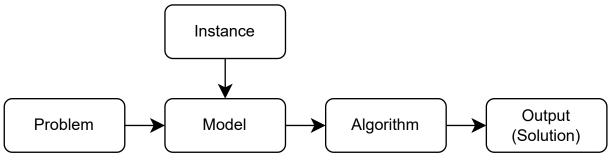


Figure 1: The scheme of an optimization approach.

Example 1 Consider a problem in which a vehicle has to deliver parcels, each to a specific address. In its simplest form, the problem can be modeled with the well-known Traveling Salesman Problem (TSP), a classical combinatorial optimization model. Given a depot where the vehicle must start and end the delivery tour and the locations where the parcels must be delivered, an instance of the model is given by the matrix of the travel times between all pairs of locations to be visited by the vehicle, including the depot. A solution to the model, obtained by means of an algorithm, is a sequence of the locations to visit that starts and ends at the depot. Because of the wide applicability of the model, its simple description and its computational complexity, many solution algorithms have been proposed over the years, both exact and heuristic (Hoffman et al. [2013]).

Table 1 presents the key components of the optimization-based approach reported in Figure 1, both in general terms and with reference to the example. Although these components are familiar to an OR expert, presenting them here is essential for the comparison with the framework and the components of supervised learning discussed later in this section.

Term	Definition	Example
Problem	A real-world problem.	A vehicle needs to deliver parcels to locations, that is to specific addresses.
Model	A simplified representation of the problem that captures the most relevant characteristics of the problem, typically a mathematical programming formulation with constraints on the decision variables and an objective function.	The TSP.
Instance	A specific set of data for the model.	The locations and the matrix of the travel times between all pairs of locations to be visited by the vehicle, including the depot.
Output	A solution to the optimization model.	The sequence of the locations to visit.
Algorithm	A procedure that finds the optimal or a good solution to the model.	A procedure that finds the optimal or a good solution to the TSP.

Table 1: The optimization-based approach to decision making.

Because of their relevance to the comparison with supervised learning, we now focus on two particular aspects of the optimization-based approach, namely the algorithm selection and the model update.

Algorithm selection When more than one algorithm, or a parametric algorithm, is available for the solution of a model, the selection of the best algorithm, or of the best parameter values, is typically carried out through tests on a set of benchmark instances. The performance of the algorithm is assessed under the assumption that it will translate reliably to the future problem instances.

The performance of an exact algorithm is typically evaluated by its running time and by the largest size of the instances that can be solved within a predefined time limit (e.g., one hour). Instead, a heuristic algorithm is typically evaluated by the gap between the solution obtained and the optimal, if available, or best known solution to the instance solved.

Model update When a model is identified to support decision making for a real-world problem, several simplifying assumptions need to be made. This implies that the solution might turn out not to satisfy all the relevant characteristics of the real-world setting. When this happens, the model needs to be revised. In the example, for instance, the travel times between pairs of locations may vary during the day, and a solution obtained under the assumption that the times are static, as in the TSP, may be unsatisfactory. A revision of the model, accounting for such a dynamic aspect, could lead to the definition of a time-dependent TSP. In most cases, the need or opportunity to revise a model is identified by experts in the real-world problem, who, by examining the solution obtained, provide feedback about its shortcomings. It is worth noting that the process of model revision needs to be carried out manually, that is, it requires the intervention of a human.

2.2 Supervised learning

Supervised learning leverages historical data describing a phenomenon to make predictions for new, previously unseen data. It has been proved to be of crucial relevance to decision making in a variety of domains, such as marketing, finance, and medical diagnosis (see for example Duarte et al. [2022], Dixon et al. [2020], and Anthimopoulos et al. [2016], respectively).

To apply supervised learning for predicting a phenomenon, a dataset of past examples, or instances, is required, where each example consists of a set of features related to the phenomenon paired with the corresponding observed value, known as label or target. This labeled dataset is then processed by an algorithm to generate a model capable of capturing the underlying relationship between the features and the label. This phase is called training. At the end of the training, the model can be applied to predict the unknown label (output) of new instances for which only the features are known. Notice that the type of output to predict defines the supervised learning task and guides the choice of the algorithm. Specifically, if the label is numerical the task is called regression, while if it has a discrete domain representing a qualitative attribute, i.e., categorical, it is referred to as classification. A scheme of the process that goes from the real problem to an output is shown in Figure 2.

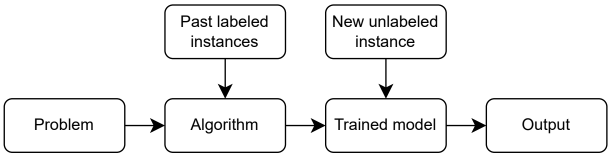


Figure 2: The scheme of the supervised learning approach.

Example 2 Consider the problem of identifying patients with a high risk of developing a disease. This can be cast in the form of binary classification, where past instances are patients for which a set of features (e.g., genetics, medical history, laboratory test results, lifestyle, environmental exposure, etc.) have been recorded, and for which a paired known label indicates whether the patient developed the given disease (ill) or not (healthy). A supervised learning algorithm, e.g., a neural network, uses these instances to train a model, which must be able to accurately classify the patients as ill or healthy. The same model is then used to label new patients based on their own feature values, with the final aim of assessing their risk of developing the disease. In Table 2 we describe the main components of the supervised learning approach, in general and for the example.

Term	Definition	Example
Problem	A real-world problem.	To identify patients with a high risk of developing a disease.
Past labeled instances	A collection of known cases of the problem where each case is characterized by the values of a set of features, or explanatory variables, paired with a label or target.	A set of patients with associated relevant features, such as laboratory test results and lifestyle elements. The label indicates whether the patient developed the given disease.
Algorithm	The procedure that generates (trains) the model that best describes the relationship between the input features and the output (label).	A neural network.
Model	The representation of the relationship learned through training.	A trained neural network, that is one for which the adjustable parameters have been learned.
New unlabeled instance	A new case for which the features are known but the label is unknown.	A patient for which just the features, that is the laboratory test results and the lifestyle elements, are known.
Solution	The predicted label for a new unlabeled case.	The predicted label (ill or healthy) for a new patient.

Table 2: The supervised learning approach to decision making.

We now focus on the aspects of algorithm selection and model update.

Algorithm selection Differently from optimization, in supervised learning the selection of the most suitable algorithm and the best values of its parameters is performed through training. The available labeled instances are divided into a training and a test set. The former is used to train the algorithm, generating a model that best captures the relationship between the features and the label. The latter, which is not used during training, serves to evaluate the model performance on future instances. This is, of course, done under the assumption that these future instances belong to the same population of the past labeled ones.

The predictions generated by the model on the test set are compared with the known labels, and the selection of the model to adopt is based on the results of this evaluation.

Model update As new labeled instances become available, the selected model can be updated to incorporate the additional information they provide. This update requires repeating the training phase. This can be done from scratch (full retraining), using the previously exploited and the newly acquired labeled instances, or incrementally (fine-tuning), by adjusting the model on the novel instances without retraining on the entire dataset. In both approaches, the model update is performed automatically and, in general, does not require human intervention. In the example, new patients with their known features and known labels may be made available over time. These new labeled instances can be used to retrain the model, allowing it to incorporate changes in the relationship between the characteristics of patients and the occurrence of the disease.

3 Optimization and supervised learning support each other

In this section, we discuss how optimization may be beneficial to supervised learning and how optimization can benefit from supervised learning.

3.1 Optimization for supervised learning

Optimization lies at the core of several supervised learning methods. In classical algorithms such as Support Vector Machines (SVMs) and linear regression or in modern deep neural networks, the training process is framed around minimizing a loss function that quantifies the discrepancy between the model predictions and the true labels. SVMs, for example, can be defined as constrained convex programming problems (see, for example, Cortes and Vapnik [1995]). Neural networks, by contrast, rely on non-convex optimization. In this case, gradient-based algorithms such as stochastic gradient descent and its variants enable models with millions of parameters to progressively and efficiently refine their internal representations (see, for instance, Ruder [2016]).

In supervised learning, a fundamental challenge is balancing the goodness of fit of the model on the past labeled instances with its generalization capability, that is, its capacity to perform well on future unseen data. In order to achieve a satisfactory balance, optimization plays a central role by allowing various machine learning algorithms to formulate the training process as a multi-criteria optimization problem that explicitly incorporates the generalization performance as a term of the objective function. This term is referred to as the regularization term and can be defined as a function of the parameters of the model, e.g., the squared magnitude of the coefficients in linear regression or neural networks (see Gambella et al. [2021] for a recent survey on optimization problems in machine learning).

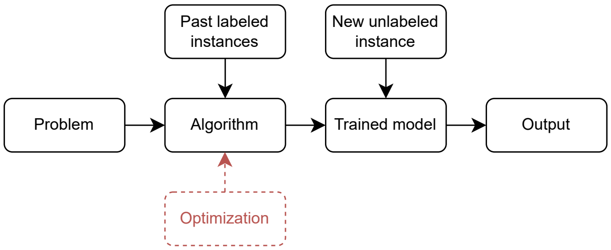


Figure 3: Role of optimization in supervised learning.

3.2 Supervised learning for optimization

As shown in Figure 1, three main blocks compose the optimization approach: the instance, the model, the algorithm. In this section, we discuss how supervised learning can be beneficial in the definition of the model for the problem at hand, in the design of the algorithm for a given model, in the generation of the instances used to test the algorithm, and in the creation of the instance on which the model is applied.

Support to instance creation and definition

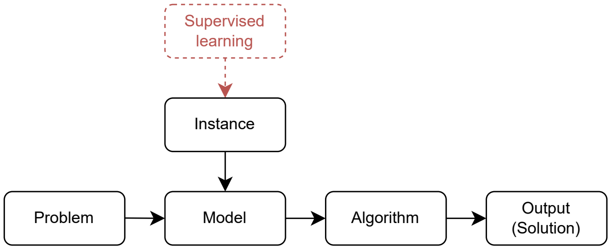


Figure 4: Role of supervised learning in the creation of optimization instances.

Supervised learning can be exploited by optimization for the creation of test instances that well represent real-world instances, and for the definition of the instance on which an optimization model is applied (see Figure 4).

As previously noted, exact algorithms are typically evaluated through their running time and the largest size of the instances they can solve within a predefined time limit, whereas heuristics are evaluated in terms of their gap from the optimal, if available, or best known solution. The tests are usually performed on a set of synthetic instances which are commonly taken from the literature or randomly generated on the basis of some underlying stochastic process. In the instance generation process, several assumptions are made about the structure of real-world instances (see, for example, Pelegrín and Cerulli [2023]) at the expense of the transferability of the results obtained and of the related considerations. It should be noted that, even when an algorithm is tested on instances taken from a real case these instances cannot always be considered as truly representative of the real instances on which the algorithm will ultimately be applied.

The issue of what instances to use for the testing of algorithms is constantly discussed in the OR community, but no better ways than the ones mentioned above have been identified so far. Although, to our knowledge, no prior studies have explored this direction, we believe that supervised learning could help generate benchmark instances that capture the intricate relationships that exist among the phenomena defining them, thus reducing the gap with the underlying real-world application. This framework could be called "ML-enhanced instance generation". In turn, this could facilitate the selection of a solution algorithm that best fits the needs of a decision maker. Consider, for example, the case of the Traveling Salesman Problem applied to a same-day logistics case where the parcels to be delivered arrive dynamically to the depot during the day. Supervised learning could help extract the relationship between the arrival time of the parcel to the depot and some variables of interest, such as traffic level, allowing the generation of more realistic instances for testing high or low traffic scenarios. This could also contribute to tailoring the solution algorithm to the specific requirements of a real-world application.

Besides the generation of instances for testing purposes, supervised learning could also be used to provide estimations for the realizations of stochastic phenomena that may be part of the input data for an optimization model, such as customer demands, costs, or times, possibly also accounting for their temporal dimension.

It must be noted that the training of a ML model is typically performed independently of its use in the downstream optimization problem, in a paradigm sometimes referred to as "predict-then-optimize" (see, for instance, Elmachtoub and Grigas [2022]). Consider, for instance, Example 1. Many elements of a TSP, e.g., travel times, are stochastic in nature but are treated as deterministic by the model. Supervised learning could provide more accurate predictions that would result in solutions that are closer to the requirements of the real setting.

Support to model definition

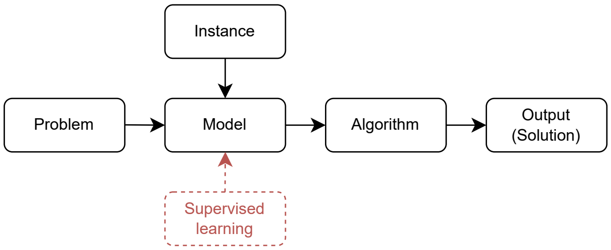


Figure 5: Role of supervised learning in model definition.

The phase of the optimization approach considered to be crucial for its success is the model definition. It is also the phase that requires human knowledge and experience, as it cannot rely on the support of any automated technique. A mathematical programming model is defined through the decision variables, the data, defined by the instance, that are assumed to be known, the objective function and the constraints.

In this context, supervised learning may be leveraged to define a surrogate objective function when the true, original one is unknown or difficult to handle, or to infer whether a non-linear objective function should be linearized (see references in Karimi-Mamaghan et al. [2022]). Additionally, supervised learning may guide the definition of the model by identifying the most effective formulation to address a given problem. For instance, it could be used to learn whether a problem should be decomposed into simpler sub-problems (see, e.g., Kruber et al. [2017]).

Support to algorithm design

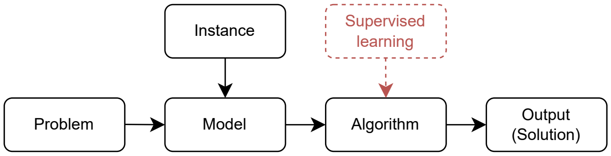


Figure 6: Role of supervised learning in optimization algorithms.

Supervised learning can also play a valuable role in the design of algorithms for solving optimization models (see Figure 6). This is an area where OR researchers have worked with the goal of improving classical algorithms using ML methods and is called “ML-augmented optimization” (see Kotary et al. [2021]).

Supervised learning can be used to recommend a specific algorithm, or the values of its parameters, on the basis of the characteristics of a problem instance. Moreover, it can guide the search within a solution algorithm. It might speed up the search for an optimal solution of an exact algorithm, for example by selecting the branching variable or a cutting plane to add to a formulation. It might also enhance the search for a heuristic solution, improving the quality of the final solution without increasing the computational time. For example, it could adapt the values of the parameters of a heuristic to the current state of the search and/or to the remaining computational time available. It could also choose the neighborhood to explore. Reviewing the growing literature in this active research area falls outside the scope of this paper. We refer the interested reader to the valuable discussion in Bengio et al. [2021]. The use of ML to support the solution of mixed-integer programming models is comprehensively surveyed in Zhang et al. [2023], while the application of ML techniques to enhance metaheuristic algorithms is reviewed in Karimi-Mamaghan et al. [2022].

Concepts from optimization and supervised learning can also be combined to create innovative solution schemes. For example, Parmentier [2022] proposes a new way to solve an optimization model that involves using supervised learning to map instances of a computationally hard optimization problem into instances of a well-studied and easier problem for which efficient algorithms exist. The solution of the latter problem is then converted to one for the hard problem.

3.3 Integration of supervised learning and optimization

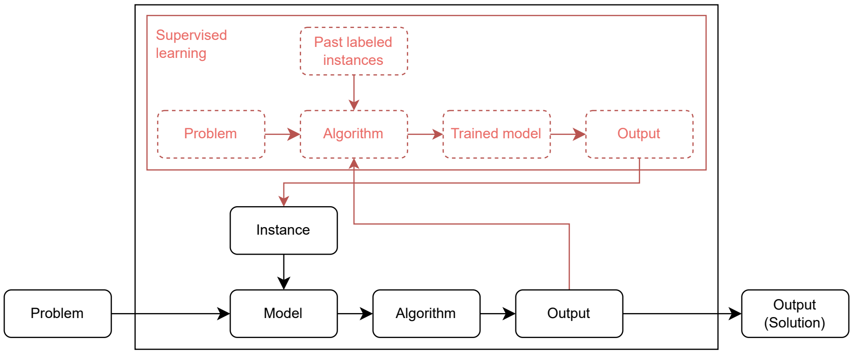


Figure 7: Integrating supervised learning and optimization.

In recent years, a few contributions have attempted to integrate optimization and supervised learning for the solution of an optimization problem through a more complex approach compared to those described in the previous sections. Returning to the discussion presented in Section 3.2, and in particular to the use of supervised learning to predict data, it is clear that different predictions may result in different solutions (and solution values) to an optimization model.

Although supervised learning can capture complex relationships between features and labels, its performance is usually evaluated using error-based metrics that do not consider the specific context in which the predictions will be used. Let us clarify this concept for Example 1, where underestimating the cost of an arc of the graph by a certain amount, or overestimating the cost of the same arc by the same amount, results in the same Mean Squared Error (a classical error-based metric in supervised learning) but may lead to different solutions. A detailed example is provided in the Appendix for a stochastic shortest path problem.

The "decision-focused learning" (DFL) paradigm (see Mandi et al. [2024]) aims to integrate prediction and optimization. The goal is to directly train a ML algorithm to make predictions that lead to good decisions. In other words, DFL integrates prediction and optimization in an end-to-end process trained to optimize a criterion (i.e., a loss function) that is based on the resulting decisions. This approach aims to obtain predictions that are optimized for their impact on the final decision making process. While results are promising, major challenges arise, most prominently concerning computational time and gradient estimation.

Embedding an optimization problem within the learning process implies the complexity of iteratively solving it during training (see Figure 7). A potential strategy to mitigate this issue is the use of heuristic algorithms or other so-called "weaker oracles". Valuable surveys and discussions on the topic are presented in Bengio et al. [2021], Kotary et al. [2021], Mandi et al. [2024], Sadana et al. [2025].

It is also interesting to report that this approach has been used within a solution algorithm (see, e.g., Parmentier [2022], Baty et al. [2024]). For example, Baty et al. [2024] propose to solve a dynamic Vehicle Routing Problem (VRP) with time windows by repeatedly solving a price collecting VRP where the prediction of the price value and the solution of the routing problem are tackled in an integrated way.

4 End-to-end learning

It is apparent from Section 2 that optimization and supervised learning have different missions and operate in substantially different ways. Nonetheless, supervised learning has been applied to solve a variety of optimization-related tasks and the scope of such tasks continues to expand.

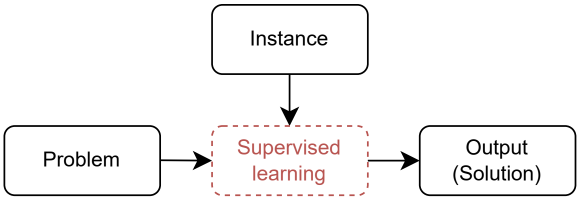


Figure 8: Supervised learning as a substitute of optimization.

The expression “end-to-end learning” (E2EL, see Bengio et al. [2021]) is used to indicate that a machine learning algorithm is trained to perform the optimization task directly, from the raw input to the desired output, that is, without explicitly solving an optimization problem (see Figure 8, also with respect to Figure 1). In this role, supervised learning would replace optimization. The core idea is that, although supervised learning would require a significant amount of time to train, once trained, it could quickly provide an output for any instance of a problem. In contrast, with a classical optimization algorithm, the solution of any instance of a problem will take the time needed to run a, possibly very slow, algorithm. In this case, supervised learning would become a competitor for optimization.

Different approaches have been explored to use supervised learning in an E2EL setting (see Kotary et al. [2021]). In this case, supervised learning algorithms take as input a representation of a problem instance and generate, in addition to the value of the objective function, a solution as output. For Example 1, the instances for the optimization problem should be transformed into instances for a supervised learning algorithm by devising a set of features to describe the graph, e.g., the costs of the arcs, and a set of targets to represent the solution, e.g., one for each arc of the graph, taking binary values to indicate whether the arc is part of the solution.

In general, ensuring the feasibility of a solution in the presence of constraints remains a major challenge (see Detassis et al. [2021]). It is worth mentioning that an interesting research area concerns the use of Graph Neural Networks to solve combinatorial optimization problems cast on graphs, in an attempt to overcome the aforementioned issue (see Cappart et al. [2023]). Another major obstacle arises from the need to have a target solution for every problem instance used in the training phase, meaning that each instance should be solved beforehand. This severely increases the computational burden, given that many optimization problems are computationally hard to solve and that, in general, a large training set is required to extract meaningful relationships between features and labels.

Although at the current stage of research development it is still highly unlikely that supervised learning will replace optimization, there exist application areas where the use of E2EL deserves attention. For example, in some cases finding an optimal solution is not essential, and a vast number of problem instances must be solved in an approximate fashion and rapidly, with the aim of predicting the value of the solution rather than the solution itself (see, e.g., Fischetti and Fraccaro [2019]).

A review of the literature where attempts to use supervised learning in an E2EL setting are presented is out of the scope of this paper. We refer to Bengio et al. [2021] and Kotary et al. [2021] as valuable references.

5 Discussion and future research directions

As noted in the previous sections, the interface between optimization and supervised learning is actively being investigated. Although there have been and will be attempts to use supervised learning as an end-to-end solution algorithm for an optimization problem, it is currently unlikely that these attempts will make supervised learning a successful competitor to the classical optimization models and algorithms. This stems, among other reasons, from the fact that supervised learning models rely on patterns observed in training data and, therefore, tend to generalize poorly to instances of the optimization problem that differ substantially from those encountered during training. Moreover, classical supervised learning architectures typically produce outputs with a fixed dimension, whereas the size of a solution often depends on the specific structure of each instance; this mismatch makes it inherently challenging to scale predictions across heterogeneous sizes of the problem instances. Additionally, embedding the feasibility requirements of an optimization problem directly within a supervised learning prediction is non-trivial, as standard learning paradigms do not natively enforce such combinatorial or logical constraints. Although replacing optimization with supervised learning is attractive, it may also entail some risks. Since supervised learning provides no formal guarantees of constraint adherence, its use in safety-critical decision processes can result in unsafe or invalid actions. For example, in a vehicle routing problem the violation of operational constraints, such as vehicle capacity or road restrictions, may generate dangerous routes. Moreover, supervised learning models may inherit human or systemic biases embedded in historical data, leading to skewed, and potentially unfair, outcomes. For the routing problem, biases may be, for example, represented by preferential servicing of certain locations or customers. Supervised learning can embed these preferences into the trained model, thereby reproducing inequitable, or fairly suboptimal, routing patterns. It is worth mentioning that another ML approach, namely reinforcement learning, is gaining much attention in the OR community for its potential in the solution of optimization problems with a strong dynamic and uncertain component.

As discussed in this paper, which is focused on the interplay of supervised learning and optimization, one of the areas of interaction that has received considerable attention is the design of optimization algorithms. Although notable contributions have already been proposed (see Karimi-Mamaghan et al. [2022]), many topics in this field remain open for further investigation. For example, learning characteristics of optimal solutions based on the input instance could narrow the search space of an algorithm, reducing the computational burden. Another way to reduce the search space of an algorithm is to learn through supervised learning which neighborhood is more helpful for instances with given characteristics. Similarly, this idea could be applied to learn when it is worth solving integrated problems, e.g., the inventory routing problem, as opposed to the underlying subproblems, i.e., inventory management and routing, separately. Problem integration has been shown to carry significant improvement in the quality of the overall solution but typically entails significant computational costs, concentrated in one single moment. Conversely, studying the subproblems separately can lead to lower quality decisions but at a lower computational cost, possibly distributed over time. Supervised learning could help balance this trade-off by learning from data when integration is beneficial and when decomposition is preferable.

Another area where supervised learning could benefit optimization concerns the generation of benchmark instances. Supervised learning techniques, including probabilistic models, could be used to learn from existing instances and generate new ones that reproduce their structural properties. This approach would enable the creation of large, diverse, and realistic sets of problem instances for testing and validation. By learning the distribution of features and solution characteristics from real-world or historically collected cases, such models could support the development of more representative and challenging benchmarks. The availability of more realistic instances could also help a better understanding of the differences between the model and the real-world setting, guiding the update of the optimization model and the improvement of the solution algorithm.

A key opportunity for future research lies in a crucial aspect of optimization algorithms, both exact and heuristic, whose implementation is typically dependent on the representation of the problem, i.e., its model. Even small generalizations of a studied problem, for which a custom algorithm has been developed, typically require intensive implementation effort to adapt the existing algorithm to the new problem. This is less so in supervised learning: while this transfer is not cost-free, since data still needs to be processed and the model still needs to be learned, the algorithmic structure remains largely intact, enabling relatively straightforward adaptation across domains. Future research could focus on leveraging supervised learning to approximate problem-agnostic representations that can serve as the input of optimization algorithms. Still, considerable issues lie in the definition of such representations. In particular, learning a representation that generalizes across problems requires identifying common structural patterns in their formulation, in such a way that different optimization problems can be mapped into a shared representational space. This demands a joint encoding of both decision variables and problem data within the same representation, as both jointly determine the feasible space and objective structure of the problem. At the same time, this shared representation must remain expressive enough to preserve the distinctive features of each problem instance, as even minor variations in their definition may substantially change the optimal solution.

Overall, the interplay between optimization and supervised learning suggests a shift from handcrafted modeling and algorithmic design to data-driven and representation-aware paradigms. Supervised learning offers tools to reshape how problems are modeled, explored, and evaluated. The interaction of these methodologies opens a path toward adaptive, more general optimization frameworks.

Disclosure of interest

There is no potential competing interests to declare.

Acknowlegments

The research of the third author has been funded by a grant PRIN 2022-PNRR.

References

M. Anthimopoulos, S. Christodoulidis, L. Ebner, A. Christe, and S. Mougiakakou. Lung pattern classification for interstitial lung diseases using a deep convolutional neural network. *IEEE transactions on medical imaging*, 35:1207–1216, 2016.

L. Baty, K. Jungel, P. S. Klein, A. Parmentier, and M. Schiffer. Combinatorial optimization-enriched machine learning to solve the dynamic vehicle routing problem with time windows. *Transportation Science*, 58:708–725, 2024.

Y. Bengio, A. Lodi, and A. Prouvost. Machine learning for combinatorial optimization: a methodological tour d'horizon. *European Journal of Operational Research*, 290:405–421, 2021.

Q. Cappart, D. Chételat, E. B. Khalil, A. Lodi, C. Morris, and P. Veličković. Combinatorial optimization and reasoning with graph neural networks. *Journal of Machine Learning Research*, 24:1–61, 2023.

C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20: 273–297, 1995.

J. Dean. A golden decade of deep learning: Computing systems & applications. *Daedalus*, 151:58–74, 2022.

F. Detassis, M. Lombardi, and M. Milano. Teaching the old dog new tricks: Supervised learning with constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 3742–3749, 2021.

M. F. Dixon, I. Halperin, P. Bilokon, et al. *Machine learning in finance*, volume 1170. Springer, 2020.

V. Duarte, S. Zuniga-Jara, and S. Contreras. Machine learning and marketing: A systematic literature review. *IEEE Access*, 10:93273–93288, 2022.

A. N. Elmachtoub and P. Grigas. Smart "predict, then optimize". *Management Science*, 68:9–26, 2022.

M. Fischetti and M. Fraccaro. Machine learning meets mathematical optimization to predict the optimal production of offshore wind parks. *Computers & Operations Research*, 106:289–297, 2019.

C. Gambella, B. Ghaddar, and J. Naoum-Sawaya. Optimization problems for machine learning: A survey. *European Journal of Operational Research*, 290: 807–828, 2021.

F. S. Hillier and G. J. Lieberman. *Introduction to operations research*. McGraw-Hill, 2015.

K. L. Hoffman, M. Padberg, G. Rinaldi, et al. Traveling salesman problem. *Encyclopedia of operations research and management science*, 1:1573–1578, 2013.

M. Karimi-Mamaghan, M. Mohammadi, P. Meyer, A. M. Karimi-Mamaghan, and E.-G. Talbi. Machine learning at the service of meta-heuristics for solving combinatorial optimization problems: A state-of-the-art. *European Journal of Operational Research*, 296:393–422, 2022.

J. Kotary, F. Fioretto, P. Van Hentenryck, and B. Wilder. End-to-end constrained optimization learning: A survey. *arXiv preprint arXiv:2103.16378*, 2021.

M. Kruber, M. E. Lübbecke, and A. Parmentier. Learning when to use a decomposition. In *International conference on AI and OR techniques in constraint programming for combinatorial optimization problems*, pages 202–210. Springer, 2017.

J. Mandi, J. Kotary, S. Berden, M. Mulamba, V. Bucarey, T. Guns, and F. Fioretto. Decision-focused learning: Foundations, state of the art, benchmark and future opportunities. *Journal of Artificial Intelligence Research*, 80:1623–1701, 2024.

A. B. Nassif, I. Shahin, I. Attili, M. Azzeh, and K. Shaalan. Speech recognition using deep neural networks: A systematic review. *IEEE access*, 7:19143–19165, 2019.

A. Parmentier. Learning to approximate industrial problems by operations research classic problems. *Operations Research*, 70:606–623, 2022.

M. Pelegrín and M. Cerulli. Aircraft conflict resolution: A benchmark generator. *INFORMS Journal on Computing*, 35:274–285, 2023.

V. Piccialli and M. Sciandrone. Nonlinear optimization and support vector machines. *Annals of Operations Research*, 314:15–47, 2022.

S. Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

U. Sadana, A. Chenreddy, E. Delage, A. Forel, E. Frejinger, and T. Vidal. A survey of contextual optimization methods for decision-making under uncertainty. *European Journal of Operational Research*, 320:271–289, 2025.

M. Soori, B. Arezoo, and R. Dastres. Artificial intelligence, machine learning and deep learning in advanced robotics, a review. *Cognitive Robotics*, 3:54–70, 2023.

A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis. Deep learning for computer vision: A brief review. *Computational intelligence and neuroscience*, 2018:7068349, 2018.

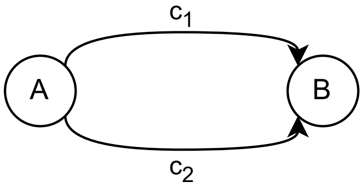
H. Wagner. *Principles of Operations Research: With Applications to Managerial Decisions.* Prentice-Hall international series in management. Prentice-Hall, 1975. ISBN 9780137095926.

W. L. Winston. *Operations research: applications and algorithm.* Thomson Learning, Inc., 2004.

J. Zhang, C. Liu, X. Li, H.-L. Zhen, M. Yuan, Y. Li, and J. Yan. A survey for solving mixed integer programming via machine learning. *Neurocomputing*, 519:205–217, 2023.

Appendix

A simple example of how training for a classic application-agnostic objective can lead to sub-optimal solutions is represented by the stochastic shortest path problem in Figure 9, adapted from Elmachtoub and Grigas [2022]. Considering two possible paths from A to B, and two sets of predictions for the cost of the arcs, predictions “1” are better from a classical supervised learning perspective, as the Mean Squared Error (MSE) is lower than that of predictions “2”. However, since in this case the predicted cost of c_2 is lower than that of c_1, arc 2 is the one that is selected as the shortest path between A and B. The opposite is true for predictions “2”, having a higher MSE but leading to a better decision.



	Cost	Pred. 1	Pred. 2
c_1	4	5	2
c_2	5	4	7
MSE	-	1	4
Opt. gap	-	1	0

Figure 9: Example of predictions for the arc costs in a stochastic shortest path.