Impacts of the Use of Machine Learning on Work Design

Kevin Crowston
crowston@syr.edu
Syracuse University School of Information Studies
Syracuse, NY

Francesco Bolici
francesco.bolici@eco.unicas.it
Università degli Studi di Cassino e del Lazio Meridionale
Cassino, FR, Italy

ABSTRACT

The increased pervasiveness of technological advancements in automation makes it urgent to address the question of how work is changing in response. Focusing on applications of machine learning (ML) to automate information tasks, we draw on a simple framework for identifying the impacts of an automated system on a task that suggests 3 patterns for the use of ML—decision support, blended decision making and complete automation. In this paper, we extend this framework by considering how automation of one task might have implications for interdependent tasks and how automation applies to coordination mechanisms.

CCS CONCEPTS

• **Social and professional topics → Automation**; *Employment issues*; • **Computing methodologies → Machine learning**.

KEYWORDS

work design, automation, machine learning, artificial intelligence, coordination

ACM Reference Format:
Kevin Crowston and Francesco Bolici. 2020. Impacts of the Use of Machine Learning on Work Design. In *Proceedings of the 8th International Conference on Human-Agent Interaction (HAI '20), November 10–13, 2020, Virtual Event, NSW, Australia.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3406499.3415070

1 INTRODUCTION

The evolution of work design—long interlinked to technology—has recently been accelerated by the increased capabilities of artificial intelligence (AI), machine learning (ML) in particular. There are many different ML techniques that can support the automation of a broad range of activities, including many decision-making tasks that until recently were the exclusive domain of humans. By automation, we mean the capability of a system to perform some tasks without human involvement. For example, ML approaches are being applied to tasks ranging from credit-card fraud detection [6], to detecting skin cancers [13], to advising judicial decisions [4]. As ML-based systems become able to handle a greater range of decision-making tasks, they can be used for more kinds of work.

HAI '20, November 10–13, 2020, Virtual Event, NSW, Australia

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/3406499.3415070

Much of the rhetoric around work and AI focuses on people being replaced by automated systems [1]. However, this view of the relationship between people and machines is too simplistic, because automatable tasks rarely stand in isolation [2, 7]. As a result, analysts expect that "technological disruptions such as robotics and machine learning—rather than completely replacing existing occupations and job categories—are likely to substitute specific tasks previously carried out as part of these jobs" [34, p. 19].

For instance, consider the work of a "computer user support specialist", a job we will use as a running example in this paper. It may soon be (if it is not already) feasible to develop an automated system to answer at least some computer users' support questions [19]. However, to be functional, such a system needs to fit the complex work of an organization. Someone must identify that there is a problem, collect relevant information to input to the system, explain the system's diagnosis to the user, implement the fix and so on. All of this surrounding work needs to adapt to an automated computer-support system (and vice versa). The research question we address in this conceptual paper is: what are the implications of different relationships between human and ML-based automated systems for designing work that includes multiple tasks ?

2 BACKGROUND

In this section, we draw on a conceptual framework for task automation to analyze how ML-based systems might have an impact on the design of human jobs. We first present a model for analyzing jobs then discuss different levels of automation. This discussion provides a basis for our exploration of the impact of different approaches to automation for work that includes multiple tasks.

2.1 Task design

We start by presenting our perspective on human work. In their jobs, most workers do a variety of different actions that might be more or less susceptible to automation. As noted above, a job is therefore not the right level of analysis at which to understand the impacts of technology. We follow the job analysis approach [32] in considering a job "an aggregation of tasks assigned to a worker" [33, p. 825]. In turn, a "task represents certain processes in which the worker, through his or her actions, transforms inputs into outputs meaningful to the goals of the job by using tools, equipment, or work aids" [33, p. 825]. The Employment and Training Administration of the U.S. Department of Labor has a database called O*Net that provides detailed information about jobs, including the comprised tasks. For example, the top three tasks (of 16) given for a "computer user support specialist" are:

(1) Answer user inquiries regarding computer software or hardware operation to resolve problems.
(2) Oversee the daily performance of computer systems.

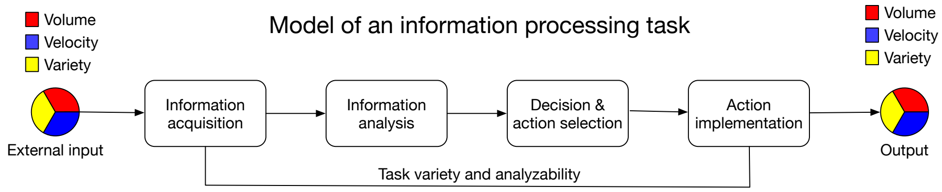

Figure 1: A four-stage model of human information processing for a task integrated with 3V at input/output and task variety and analyzability frameworks.

(3) Read technical manuals, confer with users, or conduct computer diagnostics to investigate and resolve problems or to provide technical assistance and support.

In summary, the design of work is defined as "the content and organization of one's work tasks, activities, relationships and responsibilities" [23, p. 662].

We will next recap prior work on automation of individual tasks as a basis for our analysis of multiple tasks. Research has offered different frameworks for analyzing tasks for automatability. For instance, Koorn et al. [16] classified tasks into 8 categories (e.g., creative, system supervision, information exchange) and differentiated them by ease of automation and routineness. In contrast, our goal is to examine in more detail the implications of different degrees of automation for performing a task, which calls for a more detailed framework. To analyze tasks, we draw on our prior work [8], in which synthesized different task models.

We first used a model from Parasuraman et al. [22] that suggests decomposing information-processing tasks into a "simple four-stage view of human information processing" (p. 287): 1) information acquisition; 2) information analysis; 3) decision and action selection; and 4) action implementation (see Figure 1). Since we are focusing on applications of ML, we restrict our analysis to information-processing tasks, i.e., we do not consider the impact of robots on physical work. Tasks can thus be characterized by considering their inputs, outputs and the nature of the mapping between them.

- Inputs are the information acquired for the task. To analyze these, we drew on a second model, the definition proposed for big data. Specifically, inputs are characterized by the volume, velocity and variety [17] of the information acquired, as shown to the left in Figure 1. For example, some tasks like answering user computer-support queries might have a high volume of requests in total, arriving at a high rate during certain times of the day (velocity) with a high variety of different queries, some more common, but with lots of exceptions.
- Outputs can also be characterized by the 3Vs, as shown to the right in Figure 1. In this case, by variety we consider a number of possible actions to be selected among. The decision could be binary (e.g., cancer/no cancer for a radiological screening) or of very high dimensionality (e.g., hundreds of possible replies in a customer-support setting or for a more complicated medical diagnosis). Again, we also need to consider the distribution of the outputs, whether some outputs are more common than others, i.e., the proportion of exceptions [24].
- Third, we considered the complexity of the decision rules that connect inputs and outputs, which covers the steps of information analysis and decision & action selection. These rules could be very regular [i.e., high analyzability, 24] or very irregular (low analyzability).

The above discussion has considered inputs, outputs and the mapping as static, but there could also be a dynamic aspect. For example, the nature of the inputs and outputs could change over time rather than being static and pre-given. Tasks are most likely repeated, so the information acquired as inputs could include feedback from prior rounds. And the mapping rules could evolve as system learns or as inputs and outputs change.

2.2 Levels of task automation

We next consider how a new technology might enable a task to be executed by a worker in a different way or to be completely automated. We first consider which components of the task are automated. Information acquisition has to be at least partly automated for the task to be automatable, but the performance of a task might also rely on information held by a person. Similarly, information analysis and decision selection could be done by humans, automation or some mix, as is the case for action implementation. These combinations yield a number of patterns of automation. To analyze these, we draw on a fourth model from Pacaux et al. [21], who identified 10 levels of automation, some with sublevels. Level 1 is no automation, i.e., all four steps performed by a human, while level 10 is total automation, i.e., all four steps performed by a machine without human intervention. Similar to [28] we develop a framework with four levels: no automation, decision support, blended decision making and full automation.

- No automation means that the steps of information analysis, decision & action selection and action implementation are performed by a human. We leave open the possibility that information acquisition and action implementation are supported by a system but the system does not take action autonomously, only when directed by a human. For instance, computer user support can be done entirely by a human, possibly receiving problem reports via a system.
- Decision support means that information analysis is automated, meaning that the system can recommend a few or one action to take. However, the human makes the final selection. In this case, information acquisition must also be automated to provide the input information for analysis, while action

implementation might or might not be automated. For computer user support, this level of automation means that a system reads the problem report and makes a suggestion about the diagnosis or resolution (e.g., based on a model learned from a corpus of prior reports and resolutions), but leaves the choice of a fix to the human technician.

- Blended decision making means that the system makes and implements a final decision for a subset of cases, either deferring some decisions to a human to make or having certain cases delegated to it. In this case, both information acquisition and action implementation must also be automated for at least the subset of cases that will be handled by the system. For computer user support, a system might automatically provide or even implement a solution to selected problem reports, e.g., ones for which it is a high confidence solution or if directed by the human technician.
- Finally, full automation means that the system handles all cases by itself, with the human's role being to set parameters for the system to follow, monitor performance and perhaps intervene if needed.

2.3 Drivers and limits to automation

We recognize two technological drivers that support an increase in automation from one level to the next. The first driver is digitization: increasingly more data and interactions are digital. The greater penetration of digitized data implies that the data-acquisition and action-implementation steps of our task model are increasingly done via a system, increasing the range of tasks executable through machine processing. This trend can be expected to continue for the near future. For example, consider our computer user-support specialist answering user inquiries. These inquiries could be made face-to-face or over the phone. However, if they are submitted via a computer system (digitally), then they can be processed by the system, opening the potential for automation. Similarly, if the users' computer systems are networked, an automated system could act on them directly to address problems, expanding automation to include action implementation.

The second issue is about what can be automated in the intervening steps of information analysis and decision & action selection. In the past, decision-making was automated with a set of rules: if some parameter or combination of parameters have particular values, then a particular decision is taken. ML systems provide new capabilities for complex pattern recognition. Rather than having to make explicit "if-then" rules, an ML system can learn the appropriate outputs given a large set of training examples (input-output pairs) [12]. For example, given a sufficient volume of problems and solutions, a system can learn what solution to suggest given a user query. As a result, an ML-based system can learn to identify solutions that were not coded ex-ante by humans and thus handle less analyzable mappings between inputs and outputs. This ability to learn from experience is what sets ML-approaches apart from other forms of AI. On-going development of ML technology suggests that systems will be capable of learning increasingly more complex connections between inputs and outputs. Further, systems may be trained on large general corpora and then transfer that learning to learn a specific task more quickly. For instance, an efficient approach to image classification is to start with a pre-trained model.

Nevertheless, the possibility for automation still depends on the nature of the task, particularly the proportion of exceptions in the inputs and outputs and the stability and analyzability of the mapping between them [24].

- Stable, routine tasks, those with high analyzability and few exceptions, have little or no need for information analysis or decision and action selection, meaning that a worker can just implement the actions. Such tasks are also very automatable, as long as information acquisition and action implementation can be done by the system.
- If the task has low analyzability, but few exceptions, then analysis is hard, but the selection of actions is from limited range. These tasks may be increasing amenable to automation with the capability of ML-systems to learn more complex patterns.
- For tasks with high analyzability but many exceptions, analysis may be easy, suggesting automation, but large number of choices for action may be problematic for ML, both in ensuring that the training data are complete and for achieving the necessary precision in decision making.
- Non-routine tasks are both low in analyzability and high in exceptions, suggesting that automation will be difficult.
- Finally, unstable tasks for which the inputs, outputs or the mapping evolve over time are also challenging to automate.

In addition to these technological drivers, we note a number of managerial drivers to substitute machine for human labour, such as increased productivity, cost-cutting, addressing labour shortages and a desire to regulate processing.

2.4 Challenges created by automation using machine learning

Finally, we consider some challenges created by the distinctive characteristics of ML systems that are unlike prior systems for supporting or automating work and that create new challenges for automation. We note that not all approaches to AI share these features, hence our focus here on ML.

- A first difference is that ML performance depends heavily on the quantity and quality of data available for the training [3, 15]. Furthermore, as systems are reliant on data, they often exhibit hybrid agency, combining human and machine actions; human to generate an initial dataset and then further ML-based actions, meaning that initial human biases in the data may be amplified. [26].
- Second, the results of ML are most often probabilistic: e.g., when classifying an unknown case, an ML system likely provides probabilities that the unknown case fits one of the known categories rather than a definitive answer.
- Finally, many ML techniques are opaque (deep learning in particular): unable to provide a human-understandable explanation of why a particular output was selected [12].

As a result, ML-based systems behave quite differently than programmed ones or even other approaches to AI (e.g., rule-based systems). These differences can cause problems for use and users.

The application of an ML system is an algorithmic phenomenon, but our ability to control the technology is limited: an unwanted behavior is hard to fix if it is the result of training rather than programming and the precise reason for the answer can not be easily pinpointed [10, 12]. For example, engineers at Google were embarrassed when their image-labeling-system labeled a black user as a "gorilla", but reportedly the only solution so far has been to eliminate the term "gorilla" from the labels [29]. Designing interfaces that work with the strengths and limitations of ML is an open challenge [11]. ML systems can thus display on a small scale the problems [10] described as arising from the complexity of large-scale interconnected systems.

A general concern with automation is the ability of those interacting with the automated systems to understand what the systems are doing and to intervene if needed. For instance, there have been many studies of pilots interacting with autopilot systems that largely automate the job of flying. One outcome of this work is the identification of the problem of automation surprises [27], when the human operator loses track of the state of the automated system and so is surprised by unexpected or inappropriate actions or has difficulty taking over in a crisis. The often opaque nature of ML may exacerbate this problem if it makes it harder for a user to understand what the system is doing.

3 AUTOMATION OF MULTIPLE TASKS

The contribution of this paper is to continue the analysis of the different patterns of relationship between human and ML-based automated systems identified above to understand the impact of ML-based systems across multiple tasks. Our analysis above [8] has considered an individual task. But jobs are collections of tasks, not just one, and furthermore, people doing a job typically have to interact with others. As a result, the impact of using ML for a task will propagate beyond the boundaries of the task itself.

To analyze multiple tasks, we consider how a particular task is interdependent with others, defined as "the extent to which the inputs, processes, or outputs of the tasks affect or depend on the inputs, processes, or outputs of other tasks within the same job" [33, p. 826]. For example, the second task in the list for a computer user support specialist is to monitor system performance. It may be that handling problem reports from users is helpful to see when system performance has changed, because the kinds of problems change. If handling problem reports were entirely automated, the specialists would need to develop new ways to get information about the systems. We can also consider interdependencies between tasks that compose different jobs. An isolated task might be automated with few consequences, while one that interacts with many other jobs will be more problematic. While this perspective is quite common in studies of organizational design, it is interesting to note that the O*Net database does not explicitly record task interdependencies or what other jobs a job interacts with.

To analyze interdependencies, we adopt a coordination theory approach [9, 18]. Malone and Crowston [18] analyzed group action in terms of actors performing interdependent tasks to achieve some goal. These tasks might require or create various resources. The actors face coordination problems arising from dependencies that constrain how tasks can be performed. In coordination theory, dependencies are conceptualized as arising because of the use of common resources among tasks. The key point in coordination theory is that the dependencies create problems (or possible synergies) that may require additional work to manage. The necessary tasks of managing dependencies are what Malone and Crowston [18] called coordination mechanisms. As the pattern of dependencies among tasks changes, we expect to see corresponding shifts in the needed coordination mechanisms. We next introduce each kind of dependency and associated coordination mechanisms.

- First, a shared-input dependency emerges among activities that use a common resource [like Thompson's pooled dependency, 31]. For these, the resource must be allocated to a particular user (if it is not shareable), e.g., through a schedule or first-come-first-served. If we consider as a resource the actor, either a human or a machine, who carries out the steps in the tasks, we can think of the resource assignment coordination mechanism instead as a task assignment mechanism that identifies which actor should work on which task, e.g., first-come-first-served, bidding in a market or some kind of matching. For instance, a problem report might be assigned to the first available computer support technician, technicians might themselves pick which to work on from the list of reports or a manager (or system) might match reports to technicians based on some model of expertise.
- Second, producer-consumer or flow dependencies match Thompson's sequential dependency [31]: one task produces a resource that a second uses. Flow dependencies including three sub-dependencies: the need to manage the usability of the resource as well as the timing and location of its availability. Considering usability, we might consider whether the producer of the resource adapts to the needs of the user or vice versa. For timing, we might consider whether the producer tells the user when to work or vice versa. For computer support, usability is a factor for both the quality of problem reports and suggested fixes, as discussed below.
- Third, a shared-output or fit dependency occurs when two activities collaborate in the creation of an output (in the case where the output is identical, there is potential synergy, since the duplicate work can be avoided; for instance, for computer support it is common to maintain a database of past reports to avoid solving the same problem twice).
- A final possible relation between two tasks is when one is a subtask of the other, that is, when the work to accomplish some goal is decomposed into smaller tasks to be performed. From a coordination theory perspective, the additional work needed to identify which subtasks to perform (i.e., planning) is another kind of coordination mechanism.

The above presentation starts with the dependencies, but coordination can also be analyzed by identifying a coordination mechanism (e.g., task assignment) and looking for the dependency that it manages. A final point is that coordination mechanisms are themselves tasks, so adding a coordination mechanism may create new dependencies that themselves must be managed. For instance, a task assignment mechanism may require information from another task (e.g., the skills needed), implying a flow dependency from that task to the task assignment that must be managed.

The coordination theory perspective leads us to consider two possible impacts of automation. First, we consider how automating one task might impact another task with which it is interdependent. Second, we consider the implications of automating coordination mechanisms themselves.

3.1 Automation of interdependent tasks

To illustrate the first situation, we consider issues when the output of one task is the input to the next (a flow or producer-consumer dependency). For example, our computer user support specialist consumes (or elicits) problem reports as input and produces recommended solutions as an output. The input and outputs are provided from and to some other task: the input to the support task comes from a customer who's encountered a problem and the output goes back to that customer to implement.

As discussed above, flow dependencies imply three sub-dependencies: usability, timing and transfer. We analyze the impacts of automation by considering how these dependencies are affected.

- To manage usability, coordination theory suggests approaches such as standardization, producers asking consumers what input they want, or for consumers to give feedback on the output to refine the solution. Considering the first flow dependency in our example, from the computer user to the user support system, one option to ensure that the input is usable by an automated system is to require them to be presented in a standard format. However, the need for standardized inputs may be problematic if the information is hard to express, e.g., if it is based on tacit knowledge, requiring some interaction to ensure the input is usable. We noted above that extracting useful problem reports is an important skill for a computer user support specialist, i.e., in practice the consumer of a problem report needs to work to ensure the report is usable. However, it is much more challenging to develop an automated system that can interact to elicit information from a user.
  Considering next the flow from the system back to the user, the automated system needs to know what output is needed and to present results in a usable format. In this case, guidance on addressing the problem could be given in a standard format. More likely though, it would be desirable to customize the advice to the skills and knowledge of the user, which is again more challenging.
- Regarding timing, the second sub-dependency, the consumer needs to know when the input is ready to process. Options to manage this dependency include having the producer explicitly inform the consumer or for the consumer to monitor the producer to notice when the output is ready. Automation of the consumer might be advantageous, as it might be possible to process the output on demand. In the case of computer user support that means automatically processing a problem report when it is submitted rather than waiting for a technician to be available.
- Third, transfer of the problem report and solution must be done electronically rather than physically, which we already identified as a precondition for automation.
- Finally, we note that flows of information from producer to consumer may be implicit rather than explicit. As we noted, handling customer problem reports may be a way to learn about the status of systems, e.g., identifying problems by when the problem reports change. As a result, automating problem resolution may remove an important source of information for managing systems. And conversely, if system oversight is automated, user-support specialists may be unaware of ongoing system problems that affect users. In other words, coordination can be achieved both through explicit communication and through visibility of work in other work spaces. This combination of implicit and explicit mechanisms, also referred as stigmergic coordination [5], often plays an under-appreciated role in supporting coordination. Being emergent and sometimes informal, stigmergic coordination could be difficult to map while designing and implementing a ML-system. However, to automate a task without considering its role as implicit coordination nexus could lead to a lack of communication flow into the organization and to the disconnection among some tasks.

3.2 Automation of coordination mechanisms

We next consider the possible implications of automating coordination mechanisms. We will focus on two dependencies that raise particularly interesting points, mechanisms for coordinating shared inputs and for coordinating flow dependencies.

3.2.1 Shared inputs. When multiple tasks have shared inputs, meaning they use the same resources, a task or resource assignment coordination mechanism is needed to decide which task gets which resource. Following the analysis we developed above, the task of resource assignment might be done manually, with decision support or entirely automated. The matching of task and resource can be improved by learning from past data.

We can apply the analysis performed above for individual tasks to the task-assignment task to assess its suitability for automation. For the assignment to be automatable, the analyzability of the assignment should be high with few exceptions. For instance, for Uber, drivers and riders are essentially interchangeable, meaning that the assignment is easily analyzable based on location. In contrast, the fit between a programmer and a programming task (to take another example) depends on many factors, which may be why these systems are implemented as decision support rather than fully automated.

In management studies, resource allocation is often studied together with the power balance and structure among different parts or individuals of the organizations. If automated resource allocation becomes more common, it could shift power inside organizations or among workers. For example, Uber drivers perceive having little or no control over the assignments they receive or the implications of the system's decisions for their pay [20].

3.2.2 Flow dependency. A flow or producer/consumer dependency exists when the output of a task becomes the input of another task. In the prior section, we considered the implications for coordination when one of the tasks was automated. In this section, we discuss how the needed coordination mechanisms themselves might be automated. As noted above, a producer/consumer dependency involves three sub-dependencies.

(1) To assure usability, we can rely on standardized solutions or the direct involvement of the users to explicitly express their needs. The coordination mechanism needed to determine usability can increasingly be automated thanks to the amount of available data, the possibility to integrate different sources and the velocity with which a certain output can be re-arranged according to the other variables. However, such recommendations have a range of known problems, e.g., the possibility of bad recommendation due to biases in the data or the "cold start" problem in making recommendations for a new user [25, 30].
(2) Considering timing, one approach to managing prerequisite dependencies is to signal the consumer that it can start; another is for the consumer to actively monitor the producer. The activity of notifying, as well as sequencing and tracking, can be done by a person or can be automated by a system. As with other automated notifications, an issue could be that a system overwhelms a human with notifications or requires responses more quickly than a human can provide them [14].
(3) Finally, considering transfer, with information products, it is easy for a system to manage the movement of an information product from one actor to another. (With robots, it is increasingly possible to automate such movement for physical goods as well.)

4 DISCUSSION

The patterns of automation discussed above differ in the level of automation, from individual support to complete automation. Much of the rhetoric has focused on the final case, but research is also needed on how to effectively do the former, hence our focus on elucidating patterns of interaction and their implications for work.

Despite the differences, the patterns have a number of commonalities. First is a need for a sufficient volume and quality of training materials and a sufficient regularity of the relationship between inputs and outputs. If the task has many exceptions, an ML might not be able to learn them, suggesting a decision-support or blended-decision- making pattern. A particular challenge to the blended-decision-making pattern is the ability of the ML to know when it does not know and should defer the case to the human. Finally, if the task is not stable, automation will be challenging.

A second common issue has been transparency of decision making. In all of the models, there is a need for human workers to maintain awareness of the system's performance. Human understanding of the whole system is needed to ensure that first, the agents (human and AI) collaborate in a safe manner, comprehending each other's intentions and actions; second, the automation of certain activities does not obstruct implicit coordination with other tasks; and third, humans still pay attention to the tasks, even when not directly performing them. All of these issues are tied up in the need to design a usable interface between the system and humans that informs without overwhelming.

A third issue as automation increases is to identify the circumstances under which is reasonable for humans to be able to veto the machine. These interventions may decrease the reproducibility of the task (and so face managerial opposition), but they also acknowledge that the automated system may not have complete information. Visibility and agency are tightly coupled, because the former is necessary to be able to implement the latter. We note that a human worker may technically have the authority to make the final decision (i.e., the system nominally follows the decision-support pattern) but face obstacles to exercising the authority, resulting in complete automation in practice (i.e., the human reduced to the "voice of the system" [10, p. 934]). The pressure to follow the system could be from internal management or external forces. For example, a doctor who decides to ignore the advice of a medical expert system could feel at risk for a malpractice suit for not following its encapsulated "best practice". In such a situation, a doctor might feel forced to cede authority to the system or view doing so as an easier option. Or even more simply, the design of the system may make it practically impossible to exert control, e.g., by not exposing sufficient detail about the situation or not providing effective controls to override a decision.

A fourth issue, from a practice perspective, concerns the role of organization design in ML implementation projects. Implementing ML systems means redesigning the organizational processes and redefining communication and coordination flows both at implicit and explicit level. Shifting from a human to a system performing some task requires not only technological skills and competencies but also an organizational assessment of the nature of the task being redesigned, with a specific analysis of its dependencies, both the existing ones and those that will emerge with the automation. Moreover, applying ML to specific tasks might also lead to the design of tasks that were not needed before. For example, consider the design of autonomous vehicles: when the driver is human, there is a legitimate assumption that s/he will pay attention while driving, while with an autonomous vehicle, there will be the need to motivate the human-driver to pay attention (e.g., if hands are not on the wheel, sound an alarm).

4.1 Recommendations for practice

Our conclusion is that designers have to distinguish between the automation of single information-processing tasks and the automation of work, considered as an interlacement of interdependent tasks, and that there are different options for how the automation is implemented, as described above. Necessary conditions regarding the data to apply ML solutions to both tasks and work coincide: 1) input data must be digitized; 2) data volume should be high, including known correct answers to train from; 3) data variety should be at least medium to make ML attractive and feasible; and 4) data velocity should be high to justify automation. The lack of one of these conditions would suggest the application of different solutions to that automation problem.

Considering single-task automation, in decision-support systems, the ML enables the system to anticipate the emergence of an event, warning humans and possibly providing them a set of corrective actions to take. Even better, such systems might enable a more proactive role for humans: not simply reacting to an event, but able to address the issue while or before it occurs. In this scenario, task analyzability can be medium-low or medium because

Table 1: Mapping data and task characteristics to conditions for applicability of ML

Dimension	Variable	State	Implications for use of ML
Data	Type	Digitized	Necessary condition for ML: Data must be digital to be processed by ML
	Volume	High	Necessary condition for ML: Sufficient data required for ML training
	Variety	Medium	Low variety can be handled with a simpler algorithm (e.g. if-then); too high variety makes ML difficult to implement
	Velocity	High	Low velocity would not provide a payback for implementing ML
Tasks	Analyzability	High	Highly analyzable tasks are easier to automate
		Medium	Consider implementing as decision support to incorporate human judgement
	Variety (task exceptions)	Medium	High number of different actions may be problematic both for ML training and precision of recommendation
Task Relationships: Shared input dependence	Expressibility of task and resource characteristics	High	Features on which to match tasks to resources or workers need to be explicit
Task Relationships: Flow dependence	Expressibility of consumer needs	High	Inputs to and outputs from automated systems need to be made explicit; eliciting tacit information is difficult for a system
	Volume of data about consumer	High	System needs data about consumers if it is to infer needs rather require them to be given
	Dependence	Low	Automating a task requires changes to interdependent tasks (note that dependencies may not be explicit); difficult with high level of dependence
	Visibility of process	Medium to high	Workers need visibility into automated processes to monitor performance and avoid automation surprises

the uncertainty about activities will be addressed by the human intervention. Similarly, task variety can be higher since exceptions can be handled by humans. We note that many of our examples fall into this category, in part for the reasons above, and in part because it is easier to have a system advise a human on what action to take than for it to take action itself.

In blended decision-making systems, ML is able to autonomously take care of routine activities, executing all the actions needed to resolve an issue. The system will provide a fast and accurate solution to recurrent problems, allowing humans to focus their effort on the exceptions and less obvious issues. In this case, tasks analyzability should be medium-high or high because the ML system should be able to process the task information and execute actions without human intervention in a large part of the cases. However, task variety can be medium or medium-high since routine tasks (those with few exceptions) will be autonomously managed by the machine, while others are addressed to human attention.

In full task automation systems, humans intervene only to manage few exceptions, while the machine will autonomously collect and analyze information, make a decision and execute an action. In this case the designer will be in a scenario with high analyzability and low task-variety or she should try to adjust the organizational environment towards those conditions.

Considering next work automation across multiple tasks, in decision support systems, dependencies between the automated task and the rest of the work-activities pose less of a concern because the decision will be carried out by humans and their capabilities and behaviour can serve as a buffer for the interdependent tasks. The main issues with ML automation will thus concern the human-computer interaction for the individual task.

In blended decision-making, some task outputs are autonomously processed by the machine without human intervention. In these cases, coordination should be carefully designed since the other tasks in the work-process will need to be interfaced with a machine-based output rather than a human being or artefact. At the same time the work process should also be able to function when humans make decisions and execute actions, since this scenario also happens. In this scenario the outputs will likely increase the data variety of the subsequent interdependent tasks.

Finally, in full task automation systems, the whole work process should be re-designed to have a minimum human intervention. In this scenario, the designer should focus on the hidden embedded tasks that humans perform while working and that should be made explicit when the automation is on full scale. Table 1 summarizes our recommendations mapping from data, task and task relationship characteristics to the possibility of applying ML.

5 CONCLUSION

The analysis of the articles in this paper was a pilot of the conceptual framework, with several possibilities for improvement. Many of the examples given are drawn from the popular press. A shortcoming of relying on such articles is that few discuss impacts of the system on workers in any detail. More detailed case studies are needed to establish these connections. A particular focus should be on the nature of the interface needed for the human worker to be able to interact effectively with the automation. A limitation of our analysis is that we have considered tasks atomically, as either automated or not, which does not illuminate possibilities for continuous interaction as humans carry out a task with system support, e.g., the way a modern spell checker autonomously and in real time corrects spelling mistakes as a person is typing. The framework developed in this paper may need to be extended to cover this kind of interaction. Finally, further analysis is needed of multiple intersecting tasks, not just pairs, e.g., when tasks are composed of multiple subtasks. As well, many tasks are performed by or with teams rather than solely by individuals, so research is needed to identify how a system can be an effective team member, performing a task that is interdependent with multiple human workers, not just one.

A final message of this paper is that, contrary to the rhetoric of an inevitable "march of automation", there are a variety of options for how automated systems can be used with differing impacts on jobs. The decision about which pattern to follow is partly driven by nature of the task and the work process in which it is embedded and partly by the increasing system capabilities. Designers should resist technological determinism, that is, to assume that technologies naturally evolve in given directions. They should be aware of how technological capabilities interact with managerial decisions about how technologies should be deployed, such as decisions about the desired locus of decision making. Designers should strive for a fit between system characteristics and the characteristics of the setting in which the automation is introduced. Our conceptual model can support designers in identifying different possible patterns of relationships between humans and machines, proposing a range of different scenarios for automation (not only the all-or-nothing cases) in which the deployment of the ML system is integrated with the work redesign.

REFERENCES

[1] David H. Autor. 2015. Why are there still so many jobs? The history and future of workplace automation. *Journal of Economic Perspectives* 29, 3 (2015), 3–30. https://doi.org/10.1257/jep.29.3.3

[2] David H Autor, Frank Levy, and Richard J Murnane. 2003. The skill content of recent technological change: An empirical exploration. *The Quarterly Journal of Economics* 118, 4 (2003), 1279–1333.

[3] Michele Banko and Eric Brill. 2001. Scaling to very very large corpora for natural language disambiguation. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics* (Toulouse, France). Association for Computational Linguistics, 26–33. https://doi.org/10.3115/1073012.1073017

[4] Richard Berk and Jordan Hyatt. 2015. Machine learning forecasts of risk to inform sentencing decisions. *Federal Sentencing Reporter* 27, 4 (2015), 222–228.

[5] Francesco Bolici, James Howison, and Kevin Crowston. 2016. Stigmergic coordination in FLOSS development teams: Integrating explicit and implicit mechanisms. *Cognitive Systems Research* 38 (2016), 14–22. Issue Supplement C.

[6] Philip K Chan and Salvatore J Stolfo. 1998. Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection.. In *KDD Conference*, Vol. 98. 164–168.

[7] Michael Chui, James Manyika, and Mehdi Miremadi. 2015. Four fundamentals of workplace automation. *McKinsey Quarterly* 29, 3 (2015), 1–9.

[8] Kevin Crowston and Francesco Bolici. 2019. Impacts of machine learning on work. In *Proceedings of the 52nd Hawai'i International Conference on System Sciences (HICSS-52)*. Wailea, HI. https://doi.org/10.24251/HICSS.2019.719

[9] Kevin Crowston and Charlie S Osborn. 2003. A coordination theory approach to process description and redesign. In *Organizing Business Knowledge: The MIT Process Handbook*, Thomas W Malone, Kevin Crowston, and George Herman (Eds.). MIT Press.

[10] Dionysios Demetis and Allen S Lee. 2018. When humans using the IT artifact becomes IT using the human artifact. *Journal of the Association for Information Systems* 19, 10 (2018), 5.

[11] Graham Dove, Kim Halskov, Jodi Forlizzi, and John Zimmerman. 2017. UX design innovation: Challenges for working with machine learning as a design material. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA) *(CHI '17)*. ACM, 278–288. https://doi.org/10.1145/3025453.3025739

[12] Paul Edwards. 2018. We have been assimilated: Some principles for thinking about algorithmic systems. In *IFIP WG 8.2 Working Conference on the Interaction of Information Systems and the Organization* (San Francisco, CA, USA). https://doi.org/10.1007/978-3-030-04091-8_3

[13] Andre Esteva, Brett Kuprel, Roberto A Novoa, Justin Ko, Susan M Swetter, Helen M Blau, and Sebastian Thrun. 2017. Dermatologist-level classification of skin cancer with deep neural networks. *Nature* 542, 7639 (2017), 115.

[14] Mahtab Ghazizadeh, John D. Lee, and Linda Ng Boyle. 2012. Extending the Technology Acceptance Model to assess automation. *Cognition, Technology & Work* 14, 1 (2012), 39–49. https://doi.org/10.1007/s10111-011-0194-3

[15] Alon Halevy, Peter Norvig, and Fernando Pereira. 2009. The unreasonable effectiveness of data. *IEEE Intelligent Systems* (2009), 8–12. Issue March/April.

[16] Jelmer Jan Koorn, Henrik Leopold, and Hajo A Reijers. 2018. A task framework for predicting the effects of automation. In *European Conference on Information Systems* (Portsmouth,UK). 141.

[17] Doug Laney. 2001. 3D data management: Controlling data volume, velocity and variety. *META Group Research Note* 6, 70 (2001), 1.

[18] Thomas W Malone and Kevin Crowston. 1994. The interdisciplinary study of coordination. *Computing Surveys* 26, 1 (1994), 87–119.

[19] Roel P Masongsong and Maria Amelia E Damian. 2016. Help desk management system. In *Proceedings of the World Congress on Engineering and Computer Science*, Vol. 1.

[20] Mareike Mohlmann and Lior Zalmanson. 2017. Hands on the wheel: Navigating algorithmic management and Uber drivers' autonomy. In *International Conference on Information Systems* (South Korea).

[21] M.-P. Pacaux, S. Debernard, A. Godin, B. Rajaonah, Françoise Anceaux, and Frédéric Vanderhaegen. 2011. Levels of automation and human-machine cooperation: Application to human-robot interaction. In *Proceedings of the 18th World Congress of The International Federation of Automatic Control*, Vol. 44. 6484–6492.

[22] Raja Parasuraman, Thomas B. Sheridan, and Christopher D. Wickens. 2000. A model for types and levels of human interaction with automation. *IEEE Transactions on Systems, Man, And Cybernetics—Part A: Systems and Humans* 30, 3 (2000), 286–297.

[23] Sharon K Parker. 2014. Beyond motivation: Job and work design for development, health, ambidexterity, and more. *Annual Review of Psychology* 65 (2014), 661–691.

[24] Charles Perrow. 1967. A framework for the comparative analysis of organizations. *American Sociological Review* 32 (1967), 194–208.

[25] Selwyn Piramuthu, Gaurav Kapoor, Wei Zhou, and Sjouke Mauw. 2012. Input online review data and related bias in recommender systems. *Decision Support Systems* 53, 3 (2012), 418 – 424. https://doi.org/10.1016/j.dss.2012.02.006

[26] Arun Rai, Panos Constantinides, and Saonee Sarker. 2019. Editor's comments: Next-generation digital platforms: Toward human–AI hybrids. *Management Information Systems Quarterly* 43, 1 (2019), iii–ix.

[27] Nadine B. Sarter, David D. Woods, and Charles E. Billings. 1997. Automation surprises. *Handbook of Human Factors and Ergonomics* 2 (1997), 1926–1943.

[28] A. Sasa, M. B. Juric, and M. Krisper. 2008-11. Service-oriented framework for human task support and automation. *IEEE Transactions on Industrial Informatics* 4, 4 (2008-11), 292–302. https://doi.org/10.1109/TII.2008.2008641

[29] Tom Simonite. 2018. *When it comes to gorillas, Google photos remains blind.* https://www.wired.com/story/when-it-comes-to-gorillas-google-photos-remains-blind/

[30] B. Smith and G. Linden. 2017. Two Decades of Recommender Systems at Amazon.com. *IEEE Internet Computing* 21, 3 (2017), 12–18.

[31] James D Thompson. 1967. *Organizations in Action: Social Science Bases of Administrative Theory.* McGraw-Hill.

[32] United States. Employment and Training Administration and U. S. Department of Labor. 1991. *The Revised Handbook for Analyzing Jobs.* JIST Works Incorporated.

[33] Chi-Sum Wong and Michael A Campion. 1991. Development and test of a task level model of motivational job design. *Journal of Applied Psychology* 76, 6 (1991), 825.

[34] World Economic Forum. 2016. *The future of jobs: Employment, skills and workforce strategy for the fourth industrial revolution.* World Economic Forum, Geneva, Switzerland.