Updates in Human-AI Teams: Understanding and Addressing the Performance/Compatibility Tradeoff

Gagan Bansal Besmira Nushi[†] Ece Kamar[†] Daniel S. Weld Walter S. Lasecki[‡] Eric Horvitz[†]
University of Washington, Seattle [†]Microsoft Research, Redmond [‡]University of Michigan, Ann Arbor

Abstract

AI systems are being deployed to support human decision making in high-stakes domains such as healthcare and criminal justice. In many cases, the human and AI form a team, in which the human makes decisions after reviewing the AI's inferences. A successful partnership requires that the human develops insights into the performance of the AI system, including its failures. We study the influence of *updates* to an AI system in this setting. While updates can increase the AI's predictive performance, they may also lead to behavioral changes that are at odds with the user's prior experiences and confidence in the AI's inferences. We show that updates that increase AI performance may actually hurt *team* performance. We introduce the notion of the *compatibility* of an AI update with prior user experience and present methods for studying the role of compatibility in human-AI teams. Empirical results on three high-stakes classification tasks show that current machine learning algorithms do not produce compatible updates. We propose a re-training objective to improve the compatibility of an update by penalizing new errors. The objective offers full leverage of the performance/compatibility tradeoff across different datasets, enabling more compatible yet accurate updates.

Introduction

A promising opportunity in AI is developing systems that can partner with people to accomplish tasks in ways that exceed the capabilities of either individually (Wang et al. 2016; Kamar 2016; Gaur et al. 2016). We see many motivating examples: a doctor using a medical expert system (Wang et al. 2016), a judge advised by a recidivism predictor, or a driver supervising a semi-autonomous vehicle. Indeed, economists expect human-AI teams to handle many such tasks (Gownder et al. 2017). Despite rising interest, there is much to learn about creating effective human-AI teams and what capabilities AI systems should employ to be competent partners.

We study human-AI teams in decision-making settings where a user takes action recommendations from an AI partner for solving a complex task. The user considers the recommendation and, based on previous experience with the system, decides to accept the suggested action or take a different action. We call this type of interaction *AI-advised human decision making*. While there exist other important forms of human-AI collaboration (including human-advised AI decision making and more general collaborative decision making involving a mix of initiatives and emergent team behaviors), we focus on a specific interplay where the goal is to create AI systems that recommend actions to assist humans with decisions in high-stakes domains (Angwin et al. 2016; Bayati et al. 2014). The motivation for AI-advised human decision making comes from the fact that humans and machines have complementary strengths and abilities. While both human experts and machine-learned models are not perfect on a task like medical diagnosis or classifying objects in images, researchers have shown that their ideal combination could significantly improve performance (Wang et al. 2016; Kamar, Hacker, and Horvitz 2012). AI systems offer added benefits by speeding up decision making when humans can identify tasks where the AI can be trusted and no more human effort is needed (Lasecki et al. 2012a; 2012b).

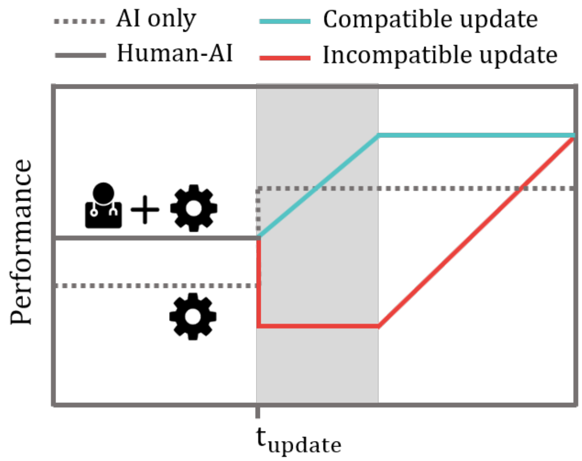


Figure 1: Schematized view of human-AI teams in the presence of AI updates. Human-AI teams perform better than either alone, but when the AI is *updated* its behavior may violate human expectations. Even if updates increase the AI's *individual* performance, they may reduce *team* performance by making mistakes in regions where humans have learned to trust the AI.

It might be expected that improvements in the performance of AI systems lead to stronger team performance, but, as with human groups, individual ability is only one of many factors that affect team effectiveness (DeChurch and

Mesmer-Magnus 2010; Grosz 1996). Even in a simple collaboration scenario, in which an AI system assists a human decision maker with predictions, the success of the team hinges on the human correctly deciding when to follow the recommendation of the AI system and when to override. Unless the particular domain and the interaction allows the human to validate the correctness of the machine recommendation efficiently and effectively, extracting benefits from collaboration with the AI system depends on the human developing insights (i.e., a mental model) of when to trust the AI system with its recommendations. If the human mistakenly trusts the AI system in regions where it is likely to err, catastrophic failures may occur. Human-AI teams become especially susceptible to such failures because of discrepancies introduced by system updates that do not account for human expectations. The following example and Figure 1 illustrate this situation.

Example (PATIENT READMISSION). *A doctor uses an AI system that is 95% accurate at predicting whether a patient will be readmitted following their discharge to make decisions about enlisting the patient in a supportive post-discharge program. The special program is costly but promises to reduce the likelihood of readmission. After a year of interacting with the AI, the doctor develops a clear mental model that suggests she can trust the AI-advised actions on elderly patients. In the meantime, the AI's developer trains and deploys a new 98% accurate classifier, which errs on elderly patients. While the AI has improved by 3%, the doctor is unaware of the new errors, and as a result of this outdated mental model, takes the wrong actions for some elderly patients.*

This example is motivated by real-world AI applications for reducing patient readmissions and other costly outcomes in healthcare (Bayati et al. 2014; Wiens, Guttag, and Horvitz 2016; Caruana et al. 2015), and motivates the need for reducing the cost of disruption caused by updates that violate users' mental models. The problem with updates extends to numerous AI-advised human decision-making settings; similar challenges have been observed during over-the-air updates in the Tesla autopilot (O'Cane 2018), and analogous issues arise in a variety of other settings when AI services being consumed by third-party applications, are updated.

Despite these problems, developers have almost exclusively optimized for AI performance. Retraining techniques largely ignore important details about human-AI teaming, and the mental model that humans develop from interacting with the system. The goal of this work is to make the human factor a first-class consideration of AI updates. We make the following contributions:

- We define the notion of compatibility of an AI update with the user's mental model created from past experience. We then propose a practical adjustment to current ML (re)training algorithms — an additional differentiable term to the logarithmic loss — that improves compatibility during updates, and allows developers to explore the performance/compatibility tradeoff.
- We introduce an open-source experimental platform[1] for studying how people model the error boundary of an AI teammate in the presence of updates for a an AI-advised decision-making task. The platform exposes important design factors (*e.g.*, task complexity, reward, update type) to the experimenter.
- Using the platform, we perform user studies showing that, humans develop mental models of AI systems across different conditions, and that more accurate mental models improve team performance. More importantly, we show that updating an AI to increase accuracy, at the expense of compatibility, may *degrade* team performance. Moreover, experiments on three high-stakes classification tasks (recidivism prediction, in-hospital mortality prediction, and credit-risk assessment) demonstrate that: (i) current ML models are not inherently compatible, but (ii) flexible performance/compatibility tradeoffs can be effectively achieved via a reformulated training objective.

AI-Advised Human Decision Making

In our studies, we focus on a simple, but common, model of human-AI teamwork that abstracts many real-world settings, *e.g.*, a 30-day readmission classifier supporting a doctor (Bayati et al. 2014), a recidivism predictor supporting judges in courts (Angwin et al. 2016). In this setting, which we call *AI-advised human decision making*, an AI system provides a *recommendation*, but the human makes the final *decision*. The team solves a sequence of tasks, repeating the following cycle for each time, t.

S1: The environment provides an input, x^t.

S2: The AI (possibly mistaken) suggests an action, $h(x^t)$.

S3: Based on this input, the human makes her decision, u^t.

S4: The environment returns a reward, r^t, which is a function of the user's action, the (hidden) best action, and other costs of the human's decision (e.g., time taken).

While interacting over multiple tasks, the team receives repeated feedback about performance, which lets the human learn when she can trust the AI's answers. The cumulative reward R over T cycles records the team's performance.

Trust as a Human's Mental Model of the AI

Cognitive psychology research shows that when people interact with any complex system, they create a mental model, which facilitates their use of the system (Norman 1988). Just as for other automated systems, humans create a mental model of AI agents (Kulesza et al. 2012). In AI-advised human decision making, valid mental models of the reliability of the AI output improve collaboration by helping the user to know when to trust the AI's recommendation. A perfect mental model of the AI system's reliability could be harnessed to achieve the highest team performance. A simple definition for such a model would be $m : x \rightarrow \{T, F\}$, indicating which inputs the human trusted the AI to solve

[1] Available at `https://github.com/gagb/caja`

correctly. A more complex model might compute a probability and include additional arguments, such as the AI's output, $h(x)$. In reality, mental models are not perfect (Norman 2014): users develop them through limited interaction with the system, and people have cognitive limitations. Furthermore, different team members may have access to different information about the situation. For example, a doctor may know things about a patient that are missing from electronic health records (*e.g.*, an estimate of the patient's compliance with taking medications), while an AI system may have access to the most recent results and trends in physiological state that are not tracked by physicians. In summary, users learn and evolve a model of an AI system's competence over the course of many interactions. In the experimental section, we show that these models can greatly improve team performance. Next, we study the problem of updating an AI system within the context of AI-assisted human decision making, and introduce the notion of compatibility.

Compatibility of Updates to Classifiers

Developers regularly update AI systems by training new models with additional or higher-quality training data, or by switching to an improved learning algorithm. Such updates presumably improve the AI's performance on a validation set, but the patient readmission example highlights how this is not always sufficient: updates can arbitrarily change the AI's error boundary, introduce new errors which violate user expectations and decrease team performance.

In software engineering, an update is *backward compatible* if the updated system can support legacy software. By analogy, we define that an update to an AI component is *locally compatible* with a user's mental model if it does not introduce new errors and the user, even after the update, can safely trust the AI's recommendations.

Definition (LOCALLY-COMPATIBLE UPDATE). *Let $m(x)$ denote a mental model that dictates the user's trust of the AI on input x. Let $A(x,u)$ denote whether u is the appropriate action for input x. An update, h_2, to a learned model, h_1, is locally compatible with m iff*

$$\forall x, [m(x) \wedge A(x, h_1(x))] \Rightarrow A(x, h_2(x))$$

In other words, an update is compatible only if, for every input where the user trusts the AI and h_1 recommends the correct action, the updated model, h_2, also recommends the correct action. In the rest of this paper, we focus on situations where a classifier's predictions are actions. For instance, in the patient readmission example, if a classifier predicts that the patient will be readmitted in the next 30 days, the suggested action from the classifier would be to include the patient in a special post-discharge program.

Globally Compatible Updates

When developers are building an AI system that is used by many individuals, it may be too difficult to track individual mental models or to deploy different updated models to different users. In this situation, an alternative to creating locally compatible updates, is a *globally compatible update*. To make this notion precise, we observe that a developer who is updating a classifier with new training data goes through the following steps:

1. Collect initial training data D_1.
2. Train a model h_1 on D_1 and deploy h_1.
3. Collect additional data to create D_2, where $D_1 \subset D_2$.
4. Train h_2 on D_2.
5. If the performance of h_2 is higher than h_1, deploy h_2.

Similar steps can be formulated for a model update where the training data does not change ($D_2 = D_1$) but h_2 belongs to a different model class.

Definition (GLOBALLY-COMPATIBLE UPDATE). *An updated model, h_2, is globally compatible with h_1, iff*

$$\forall x, A(x, h(x)) \Rightarrow A(x, h_2(x))$$

Note that a globally compatible update is locally compatible for *any* mental model. While global compatibility is a nice ideal, satisfying it for all instances is difficult in practice. More realistically, we seek to minimize the number of errors made by h_2's that were not made by h_1, since that will hopefully minimize confusion among users. To make this precise, we introduce the notion of a *compatibility score*.

Definition (COMPATIBILITY SCORE). *The compatibility score $\mathcal{C}$ of an update h_2 to h_1 is given by the fraction of examples on which h_1 recommends the correct action, h_2 also recommends the correct action.*

$$\mathcal{C}(h_1, h_2) = \frac{\sum_x A(x, h_1(x)) \cdot A(x, h_2(x))}{\sum_x A(x, h_1(x))} \qquad (1)$$

If h_2 introduces no new errors, $\mathcal{C}(h_1, h_2)$ will be 1. Conversely, if all the errors are new, the score will be 0.

Dissonance and Loss

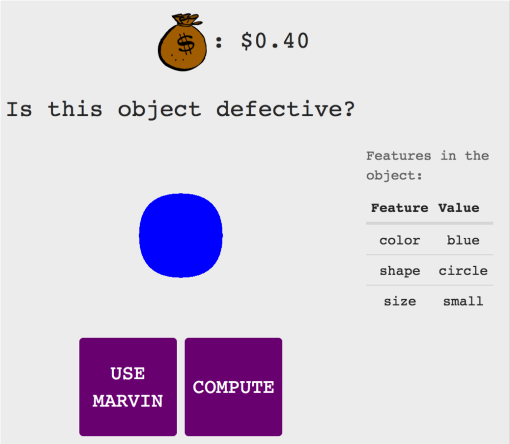
To train classifiers, ML developers optimize for the predictive performance of h_2 by specifying, and minimizing, a classification loss L that penalizes low performance. The equation below shows the negative logarithmic loss (also known as log loss or cross-entropy loss) for binary classification – a commonly used training objective in ML.

$$L(x, y, h_2) = y \cdot \log p(h_2(x)) + (1-y) \cdot \log(1 - p(h_2(x)))$$

Here, the probability $p(h(x))$ denotes the confidence of the classifier that recommendation $h(x)$ is true, while y is the true label for x (*i.e.*, $A(x,y) =$ *True*). The negative log loss, like many other loss functions in machine learning, depends only on the true label and the confidence in prediction – it ignores the previous versions of the classifier and, hence, has no preference for compatibility. As a result, retraining using different data can lead to very different hypotheses, introduce new errors, and decrease the compatibility score. To alleviate this problem, we define a new loss function L_c expressed as the sum of classification loss and *dissonance*.

Definition (DISSONANCE). *The dissonance $\mathcal{D}$ of h_2 to h_1 is a function $\mathcal{D} : x, y, h_1, h_2 \rightarrow \mathcal{R}$ that penalizes a low compatibility score. Furthermore, $\mathcal{D}$ is differentiable.*

$$\mathcal{D}(x, y, h_1, h_2) = \mathbb{1}(h_1(x) = y) \cdot L(x, y, h_2) \qquad (2)$$



: $0.40

Is this object defective?

Features in the object:

Feature	Value
color	blue
shape	circle
size	small

Figure 2: Screenshot of the CAJA platform for studying human-AI teams.

Recall that $\mathcal{C}(h_1, h_2)$ is high when both h_1 and h_2 are correct (Eqn 1). Dissonance expresses the opposite notion: measuring if h_1 is correct ($\mathbb{1}$ denotes an indicator function) and penalizing by the degree to which h_2 is incorrect. Equation 3 defines the new loss.

$$L_c = L + \lambda_c \cdot \mathcal{D} \tag{3}$$

Here, λ_c encodes the relative weight of dissonance, controlling the additional loss to be assigned to all new errors. We refer to this version as *new-error dissonance*. Just as with classification loss, there are other ways to realize dissonance. We explored two alternatives, which we refer to as *imitation* and *strict imitation* dissonance. Eqn 4 describes the imitation dissonance which measures the log loss between the prediction probabilities of h_1 and h_2:

$$\mathcal{D}'(x, y, h_1, h_2) = L(x, h_1, h_2) \tag{4}$$

Eqn 4 is used in model distillation (Ba and Caruana 2014; Hinton, Vinyals, and Dean 2015), where the aim is to train a shallower, less expensive model by imitating the probabilities of larger, accurate model. Unfortunately, $\mathcal{D}'$ has the effect of nudging h_2 to mimic h_1's mistakes as well as its successes. Eqn 5 describes the strict imitation dissonance, which follows a similar intuition but it only adds the log loss between h_1 and h_2 when h_1 is correct.

$$\mathcal{D}''(x, y, h_1, h_2) = \mathbb{1}(h_1(x) = y) \cdot L(x, h_1, h_2) \tag{5}$$

Compared to dissonance, $\mathcal{D}$, $\mathcal{D}''$ still puts a larger emphasis on matching h_1's predictions (*vs.* the true labels, y), which we worried would hurt accuracy. Our experiments (*e.g.*, Figure 5) confirm this intuition and show the effect of varying λ_c on the performance/compatibility tradeoff.

Platform for Studying Human-AI Teams

How might we study the impact of AI accuracy, updates, compatibility, and mental models on the performance of AI-advised human decision making teams? Ideally, we would conduct user studies in real-world settings, varying parameters like the length of interaction, task and AI complexity, reward function, and the AI's behavior. All human-subjects research is challenging, but our setting poses special perplexities. Testing in real settings reduces or removes our ability to directly control the performance of the AI. Furthermore, it may largely measure experts' differing experience in the domain, rather than their interactions with the AI. The importance of mental models for team success varies among domains and the interaction designed between the AI system and humans. When humans do not have an easy way to validate machine correctness, extracting value out of AI assistance depends on the ability of the human developing a mental model of the AI system.

To control for human expertise and the centrality of mental modeling, we developed the CAJA platform, which supports parameterized user studies in an assembly line domain that abstracts away the specifics of problem solving and focuses on understanding the effect of mental modeling on team success. CAJA is designed such that *no* human is a task expert (nor can they become one). In fact, the true label of decision problems in the platform is randomly generated so that people cannot learn how to solve the task. However, humans can learn when their AI assistant, Marvin, succeeds and when Marvin errs. Alongside, the human has access to a perfect problem-solving mechanism, which she can use (at extra cost) when she does not trust Marvin.

Specifically, CAJA is a web-based game, whose goal is to make classification decisions for a fixed number of box-like objects. For each object, the team follows the steps S1-S4 to decide whether the object is "defective" or not. In S1 a new object appears (*e.g.*, blue square), in S2 the AI recommends a label (*e.g.*, not-defective), in S3 the player chooses an action (*e.g.*, accept or reject the AI recommendation), and in S4 the UI returns a reward and increments the game score. The objects are composed of many features, but only a subset of them are made *human-visible*. For example, visual properties like shape, color, and size are visible, but the contents are not. In contrast, the AI has access to all the features but may make errors. At the beginning of the game, users have no mental model of the AI's error boundary. However, to achieve high scores, they must learn a model using feedback from step S4. Figure 2 shows a screenshot of the game at step S3.

CAJA allows study designers to vary parameters, such as the number of objects, number human-visible features, reward function, AI accuracy, and complexity of perfect mental model (number of clauses and literals in the error boundary and stochasticity of errors). Further, it enables one to study the effects of updates to AI by allowing changes to these parameters at any time step. In the next section, we use CAJA to answer various research questions.

Experiments

We present experiments and results in two parts. First, using our platform, we conduct user studies to understand the impact of mental models and updates on team performance. Second, we simulate updates for three real-world, high-stakes domains and show how the retraining objective

	Accept	Compute
AI right	\$0.04	0
AI wrong	-\$0.16	0

Table 1: Reward matrix for the user studies. To mimic high-stakes domains, penalty for mistakes is set to high.

enables an explorable tradeoff between compatibility and performance that is not available in the original models.

User Studies. In user studies, we hired MTurk workers and directed them to the CAJA platform.[2] We informed them of the purpose of the study and provided a set of simple instructions to familiarize them with the task and the user interface: form a team with an AI, named Marvin, and label a set of 100 objects as "defective" or "not defective". Following AI-advised human decision making, to label an object, a worker can either accept Marvin's recommendation, which is initially correct 80% of the time, or use the "compute" option, which is a surrogate for the human doing the task herself perfectly but incurring an opportunity cost. Table 1 summarizes the reward function used in our studies. The matrix is designed in a way that it imitates a high-stakes scenario, i.e., the monetary penalty for a wrong decision is much higher than the reward for a correct decision. We found this design choice to be a good incentive for workers to learn and update their mental model on Marvin. Note that the expected value of a pure strategy (*e.g.*, always "Compute" or always "Accept," without considering the likelihood of Marvin's correctness) is zero. The only way to get a higher score is by learning when to trust Marvin. While the subjects are told Marvin's accuracy and the payoff matrix, they can only learn Marvin's error boundary gradually by playing the game. These design choices allow us to study the impact of mental models while controlling for human problem solving expertise — every player is able to solve problems perfectly, at a fixed cost, using "compute."

Q1: *Do better mental models of AI lead to higher team performance?*

To answer this, we conducted human studies that measured team performance across different conditions of complexity of task and error boundary. For each condition, we hired 25 MTurk workers, and filtered spammers by deleting data from workers in the bottom quartile. We varied the task complexity by varying the number of human-visible features. The complexity of error boundary f, expressed as a logical formula, is varied by changing the number of conjuncts and literals in f. For example, we tried one conjuncts containing two literals, one conjunct with three literals, and two conjuncts with two literals. Since many features can be used as literals, we chose them randomly to create different but isomorphic error boundaries. For example, worker A gets $f_A = (blue \cap square)$ and worker B gets $f_B = (red \cap circle)$. Figure 3a shows that, for one conjunct and two literals, team performance decreases with the number of features. We observed a similar behavior for other error boundaries (results omitted for space), and for the rest of these studies we set the number of conjuncts to one. Figure 3a shows that, as the number of features increases, team performance decreases because it becomes harder to create a mental model.

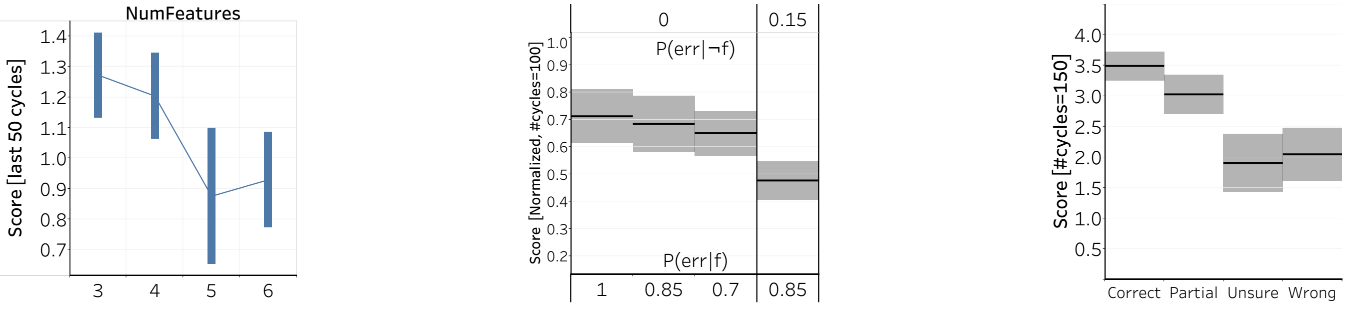
Next, we conducted a study (Figure 3b) to understand the impact of *stochasticity* in the error boundary on team performance. Stochasticity is defined using two conditional probabilities: $P(err|f)$ and $P(err|\neg f)$. That is, the probability of error if f is satisfied, and if it is not satisfied. To vary stochasticity, we chose the following four pairs of probabilities: (0.7, 0), (0.85, 0), (1.0, 0), and (0.85, 0.15). For the first three pairs, the errors are "one-sided": since $P(err|\neg f)$ is 0, the classifier makes a mistake only if the formula is satisfied. In the last pair, errors are "two-sided": with a probability of 0.15, the classifier makes a mistake even if the formula is not satisfied. We fix the number of features to three and the number of literals to two. Figure 3b shows that as errors become more stochastic, it becomes harder to create a mental model, deteriorating team performance. The y-axis shows the score normalized by the score of the optimal policy because, as we vary stochasticity, the optimal policy's score changes.

Finally, in order to have a closer view of the quality of the workers' mental models, we ask them to self report when they thought Marvin was wrong. We manually labeled these reports as correct, partial, unsure, and wrong, without looking at their team performance. The label denotes how the worker's mental model compared to the true error boundary f. For example, correct denotes that the mental model and f were the same, wrong denotes no match, partial denotes an incomplete match, and unsure denotes that the worker was skeptical of their mental model. Figure 3c compares team performance on these groups. Workers with the correct mental model score the highest, followed by workers with a partially correct model. These observations confirm that better mental models contribute positively to team performance.

Q2: *Do more compatible updates lead to higher team performance than incompatible updates?*

To study the impact of updates, we set the number of cycles to 150, and at the 75th cycle, update the classifier to a version that is 5% more accurate (80% → 85%). Then, we divide the participants into three groups: same error boundary, compatible error boundary, and incompatible error boundary. The same error boundary group receives an update improving accuracy, but the error boundary is unchanged. For the two other groups, the number of literals (features) in the error boundary changes from two to three. The update for the compatible error boundary group introduces no new errors; for example, if before the update the error boundary was $blue \cap square$, after the update it may change to $small \cap blue \cap square$. For the incompatible error boundary group, the error boundary introduces new errors violating compatibility. Figure 4 summarizes our results. We also show the performance of workers if no update was introduced (dashed line). It uses the no-update setting from experiments in Q1, and extrapolates from there assuming that

[2] Workers were paid on average \$20/hr, over the minimum wage in line with ethical guidelines for requesters (Dynamo 2017).



(a) (b) (c)

Figure 3: (a) Team performance decreases as we increase the number of human-visible features. (b) Team performance decreases with the stochasticity of errors. The decrease is much higher for two-sided errors. (c) Better mental models result in higher team performance. Wrong and Unsure mental models have the lowest performance.

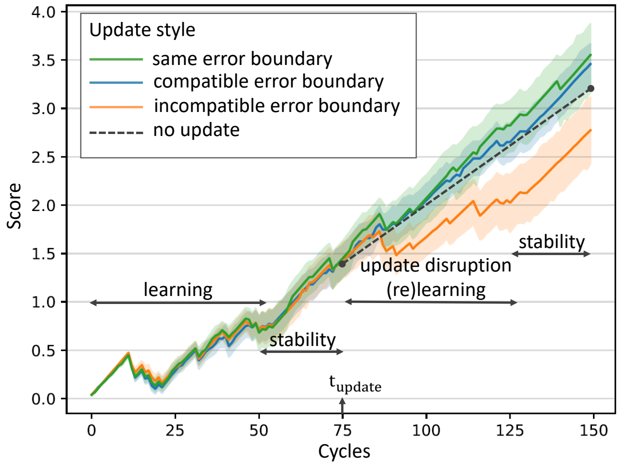


Figure 4: Team performance for different update settings. Compatible updates improve team performance, while incompatible updates hurt team performance despite improvements in AI accuracy.

the worker's mental model is already stable at the 75th cycle, meaning that the human-AI team has reached the maximum performance for the original setting and no further improvements are expected. The graph demonstrates two main findings on the importance of compatibility. First, a more accurate but incompatible classifier results in lower team performance than a less accurate but compatible classifier (no update). Second, compatible updates improve team performance. Moreover, the figure shows different stages during the interaction: the user learning the original error boundary, team stabilizes, update causes disruption, and performance stabilizes again. A central insight in the update stage is that the incompatible error boundary condition sacrifices the team score while workers have to relearn the new boundary. This insight shows that compatible updates not only improve team performance but they can also reduce the cost of retraining users after deploying system updates.

Classifier	Dataset	ROC h_1	ROC h_2	$\mathcal{C}(h_1, h_2)$
LR	Recidivism	0.68	0.72	0.72
	Credit Risk	0.72	0.77	0.66
	Mortality	0.68	0.77	**0.40**
MLP	Recidivism	0.59	0.73	0.53
	Credit Risk	0.70	0.80	0.63
	Mortality	0.71	0.84	0.76

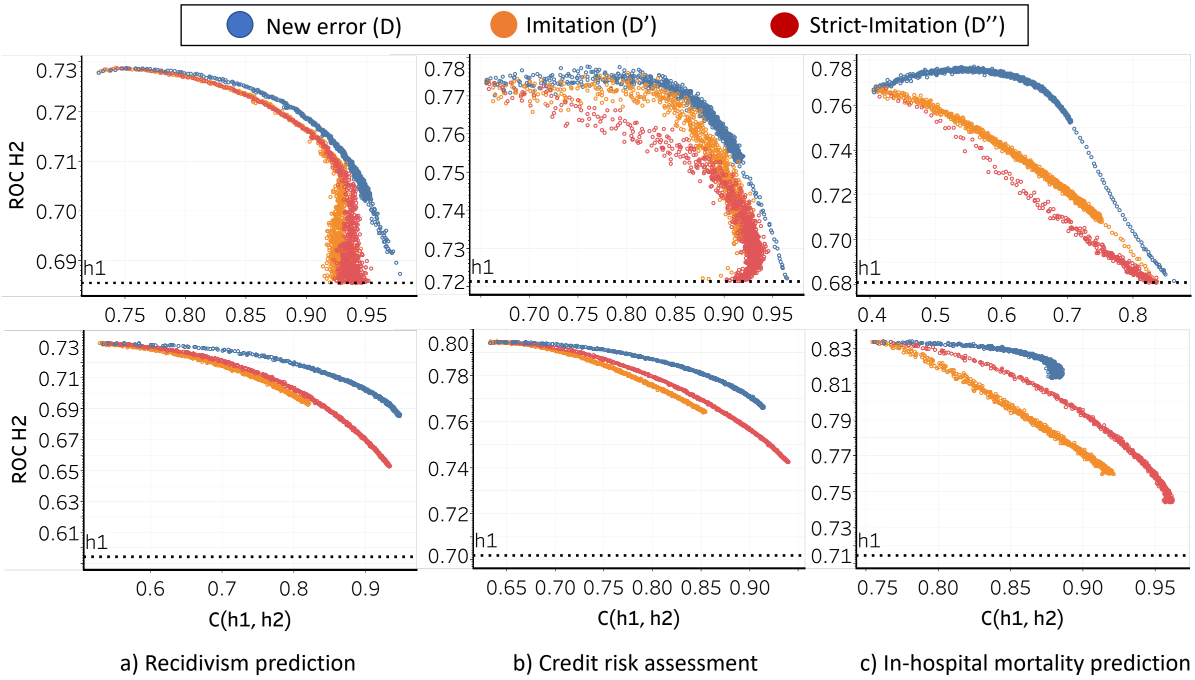
Table 2: Although training on a superset of data increases classifier performance, compatability can be suprisingly low.

Experiments with High-Stakes Domains

Datasets. To investigate whether a tradeoff exists between performance and compatibility of an update, we simulate updates to classifiers for three domains: recidivism prediction (Will a convict commit another crime?)(Angwin et al. 2016), in-hospital mortality prediction (Will a patient die in the hospital?) (Johnson et al. 2016; Harutyunyan et al. 2017), and credit risk assessment (Will a borrower fail to pay back?)[3]. We selected these high-stakes domains to highlight the potential cost of mistakes caused by incompatible updates in human-AI teams.

Q3: *Do current ML classifiers produce compatible updates?* For this experiment, we first train a classifier h_1 on 200 examples and note its performance. Next, we train another classifier h_2 on 5000 examples and note its performance and compatibility score. We train both classifiers by minimizing the negative log loss. Table 2 shows the performance (area under ROC) and compatibility averaged over 500 runs for logistic regression (LR) and multi-layer perceptron (MLP) classifiers. We find that training h_2 by just minimizing log loss does not ensure compatibility. For example, for logistic regression and the in-hospital mortality prediction task, the compatibility score is as low as 40%. That is, 60% of the instances where h_1 was correct are now violated.

[3] `https://community.fico.com/s/explainable-machine-learning-challenge`



a) Recidivism prediction

b) Credit risk assessment

c) In-hospital mortality prediction

Figure 5: Performance vs. compatibility for a logistic regression and multi-layered perceptron classifiers. The reformulated training objective (L_c) offers an explorable performance/compatibility tradeoff, generally more forgiving during the first half of the curves. The training objective based on new-error dissonance performs the best, whereas the ones based on imitation and strict-imitation dissonance perform worse since they imitate probabilities of a less accurate, and less calibrated model (h_1).

Q4: *Does there exist a tradeoff between the performance and the compatibility of an update to AI?*

For Q4 (and Q5), we learn the second classifier h_2 by minimizing L_c. As L_c depends also on the first classifier, we make its prediction available to the learner. We vary λ_c and summarize the resulting performance and compatibility scores across different datasets for the logistic regression and multi-layer perceptron classifiers in Figure 5 and for different definitions of dissonance (discussed in Q5). The figure shows that there exists a tradeoff between the performance of h_2 and its compatibility to h_1. This tradeoff is generally more flexible (flat) in the first half of the curves. This shows that, at the very least, one can choose to train via L_c and deploy a more compatible update without significant loss in accuracy. Although such updates are not fully compatible, they might still be relevant to be picked by the developer if the update is supported by efficient explanation techniques that can help users to better understand how the model has changed. In these cases, a more compatible update would also reduce the effort of user (re)training. In the second half, the tradeoff becomes more evident. High compatibility can sacrifice predictive performance. Look-up summaries similar to graphs shown in Figure 5 are an insightful tool for ML developers that can guide them select an accurate yet compatible model based on the specific domain requirements.

Q5: *What is the relative performance of the different dissonance functions?*

Figure 5 compares the performance of the new-error dissonance function ($\mathcal{D}$) with the imitation-based dissonances ($\mathcal{D}'$ and $\mathcal{D}''$). As anticipated, $\mathcal{D}$ performs best on all three domains. The definitions inspired by model distillation, $\mathcal{D}'$ and $\mathcal{D}''$, assume that h_1 is calibrated, and more accurate. Therefore, h_2 needs to remain faithful to only the correct regions of a less accurate model h_1. If these assumptions are violated, h_2 overfits to non-calibrated confidence scores of h_1, which hurts performance.

Discussion and Directions

The AI-assisted human decision-making problem assumes that there are instances for which the AI is more efficient (*e.g.*, higher accuracy, faster, or low resource usage), and the human can recognize when the AI is capable of doing so. Earlier, we discussed that one way for humans to recognize when to follow the AI's recommendations is by creating mental models. However, depending on the domain and the type of interaction design, the importance of mental modeling for team performance may vary. For example, if the human can quickly validate the correctness of the recommendation, or the human expertise improves over time to leave no room for machine contribution, then mental modeling may not be needed. Otherwise, the accuracy of the mental model limits team performance. Thus, compatibility of updates becomes an essential determinant of team performance, and developers should factor it in system design supported by guiding tools exploring the performance/compatibility tradeoff.

Varying the value of λ_c results in numerous models on the performance/compatibility spectrum. The decision to select the appropriate model depends on several factors, including the user ability to create a mental model, the cost of disruption, and whether there exist other alternative approaches for minimizing disruption caused by updates. For example, if the cost of disruption (both the cognitive cost and mistakes) is high, then we may use a high value for λ_c. A more formal approach would be to set λ_c algorithmically. For example, a λ_c could be selected to maximize expected utility expressed using a computational user model and future rewards.

A developer can use other complementary approaches to minimize disruption caused by low compatibility. One approach is to retrain the user, for example, by leveraging mechanisms from interpretable AI to explain the updated model to users or to explain differences between h_1 and h_2. However, this may not always be practical: (1) in practice, developers may push updates frequently, and since retraining requires users additional time and effort, it may not be practical to subject experts to repeated re-training; (2) updates can arbitrarily change the decision boundary of a classifier, and as a result, require the user to re-learn a large number of changes; (3) re-training requires the developers to create an effective curriculum or generate a change summary based on the update. It is often impossible to compute such summaries in a human-interpretable way. For example, explaining the changes to a self-driving car may require the challenging task of mapping the feature representation used by the car (myriad of sensor data) to human-interpretable concepts. Nevertheless, backward compatibility does not preclude retraining; these techniques are complementary to each other. In fact, more compatible updates can be an efficient mechanism to simplify the re-training process by minimizing the divergence between two models deployed consecutively. Yet another complementary approach is to share the AI's confidence in the prediction. Well-calibrated confidence scores can help a user to decide when or how much to trust the system. Unfortunately, confidence scores of ML classifiers are often not calibrated (Nguyen, Yosinski, and Clune 2015) or a meaningful confidence definition may not exist due to the complexity of the task.

We formalized compatibility in terms of differences in model recommendations before and after an update, independent of mental models of users. An important future direction is to develop computational models of how people create and update mental models, and condition on the personalized experiences and cognitive capabilities of each user, drawing upon general findings about how people learn about phenomena via observation (Reber 1989). While this work distills trust as the essence of teamwork and presented results are applicable to a variety of use cases, promising extensions include developing blended studies in the real world that combine both factors of human problem solving and learned trust in AI.

Related Work

Prior seminal work explored the importance of mental models for achieving high performance in group work (Grosz and Kraus 1999), human-system collaboration (Rouse *et al.* 1992), and interface design (Carroll and Olson 1988). Our work builds upon these foundations and studies the problem for AI-advised human decision making. Other work (Hoff and Bashir 2015) highlights the connection between mental models and trust in systems. While many "layers" of trust exist, our work focuses on *learned trust*, which is built upon context and past experiences (Marsh and Dibben 2003). Previous work (Zhou et al. 2017) investigated factors that affect user-system trust, e.g., model uncertainty and cognitive load. The platform proposed in this work enables human studies that can analyze the effect of such factors.

The field of software engineering also considers the problem of backward compatibility, seeking to design components that, after updates, remain compatible with a larger software ecosystem (Bosch 2009; Spring 2005; Tsantilis 2009). Machine learning research has explored related notions. *Stability* expresses the ability of a model to not significantly change its predictions given small changes in the training set (Bousquet and Elisseeff 2001). *Consistency*, which has application in ML fairness, is a property of smooth classifiers, which output similar predictions for similar data points (Zhou et al. 2004). *Catastrophic forgetting* is an anomalous behavior of neural network models that occurs when they are sequentially trained to perform multiple tasks and forget to solve earlier tasks over time (Kirkpatrick et al. 2017). While these concepts are fundamental for analyzing changing trends in continuously learned models, they do not consider human-AI *team* performance nor prior user experience. Related to our proposed retraining objective is the idea of *cost-sensitive* learning (Elkan 2001), where different mistakes may cost differently; for example, false positives may be especially costly. However, in our case, the cost also depends on the behavior of the previous model h_1.

Conclusions

We studied how updates to an AI system can affect human-AI team performance and introduced methods and measures for characterizing and addressing the compatability of updates. We introduced CAJA, a platform for measuring the effect of AI performance and the effect of updates on team performance. Since humans have no experience with CAJA's abstract game, the platform controls for human problem-solving skill, distilling the essence of mental models and trust in one's AI teammate. Using CAJA, we presented experiments demonstrating how an update that makes an AI component more accurate can still lead to diminished human-AI team performance. We introduced a practical re-training objective that can improve the compatibility of updates. Experiments across three data sets show that our approach creates updates that are more compatible, while maintaining high accuracy. Therefore, at the very least, a developer can choose to deploy a more compatible model without sacrificing performance.

Acknowledgments

We thank M. Ribeiro, S. Tan, X. Zhang, R. Caruana, M. Czerwinski, J. Suh, M. Joshi, J. Bragg, and anonymous reviewers for helpful feedback. This work was was supported by

Microsoft Research, ONR grant N00014-18-1-2193, the future of Life foundation and the WRF/Cable Professorship

References

Angwin, J.; Larson, J.; Mattu, S.; and Kirchner, L. 2016. Machine bias: There's software across the country to predict future criminals and its biased against blacks. *ProPublica*.

Ba, J., and Caruana, R. 2014. Do deep nets really need to be deep? In *NIPS*, 2654–2662.

Bayati, M.; Braverman, M.; Gillam, M.; Mack, K. M.; Ruiz, G.; Smith, M. S.; and Horvitz, E. 2014. Data-driven decisions for reducing readmissions for heart failure: General methodology and case study. *PloS one* 9(10).

Bosch, J. 2009. From software product lines to software ecosystems. In *SPLC*, 111–119.

Bousquet, O., and Elisseeff, A. 2001. Algorithmic stability and generalization performance. In *NIPS*, 196–202.

Carroll, J. M., and Olson, J. R. 1988. Mental models in human-computer interaction. In *Handbook of human-computer interaction*. Elsevier. 45–65.

Caruana, R.; Lou, Y.; Gehrke, J.; Koch, P.; Sturm, M.; and Elhadad, N. 2015. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *KDD*, 1721–1730.

DeChurch, L. A., and Mesmer-Magnus, J. R. 2010. The cognitive underpinnings of effective teamwork: A meta-analysis. *Journal of Applied Psychology* 95(1):32.

Dynamo. 2017. *Guidelines for Academic Requesters*.

Elkan, C. 2001. The foundations of cost-sensitive learning. In *IJCAI*, volume 17, 973–978.

Gaur, Y.; Lasecki, W. S.; Metze, F.; and Bigham, J. P. 2016. The effects of automatic speech recognition quality on human transcription latency. In *Proceedings of the 13th Web for All Conference (W4A)*, 23. ACM.

Gownder, J.; Voce, C.; Koetzle, L.; LeClair, C.; Purcell, B.; Sridharan, S.; Garberg, C.; and Lynch, D. 2017. Techradar: Automation technologies, robotics, and AI in the workforce. Technical report, Forrester Research.

Grosz, B. J., and Kraus, S. 1999. The evolution of shared-plans. In *Foundations of rational agency*. Springer.

Grosz, B. J. 1996. Collaborative systems (AAAI-94 presidential address). *AI magazine* 17(2):67.

Harutyunyan, H.; Khachatrian, H.; Kale, D. C.; and Galstyan, A. 2017. Multitask learning and benchmarking with clinical time series data. *arXiv preprint arXiv:1703.07771*.

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Hoff, K. A., and Bashir, M. 2015. Trust in automation: Integrating empirical evidence on factors that influence trust. *Human Factors* 57(3):407–434.

Johnson, A. E.; Pollard, T. J.; Shen, L.; Li-wei, H. L.; Feng, M.; Ghassemi, M.; Moody, B.; Szolovits, P.; Celi, L. A.; and Mark, R. G. 2016. MIMIC-III, a freely accessible critical care database. *Scientific data* 3.

Kamar, E.; Hacker, S.; and Horvitz, E. 2012. Combining human and machine intelligence in large-scale crowdsourcing. In *AAMAS*, 467–474.

Kamar, E. 2016. Directions in hybrid intelligence: Complementing AI systems with human intelligence. In *IJCAI*.

Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N.; Veness, J.; Desjardins, G.; Rusu, A. A.; Milan, K.; Quan, J.; Ramalho, T.; Grabska-Barwinska, A.; et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences* 201611835.

Kulesza, T.; Stumpf, S.; Burnett, M.; and Kwan, I. 2012. Tell me more?: the effects of mental model soundness on personalizing an intelligent agent. In *CHI*, 1–10. ACM.

Lasecki, W.; Miller, C.; Sadilek, A.; Abumoussa, A.; Borrello, D.; Kushalnagar, R.; and Bigham, J. 2012a. Real-time captioning by groups of non-experts. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*, 23–34. ACM.

Lasecki, W. S.; Bigham, J. P.; Allen, J. F.; and Ferguson, G. 2012b. Real-time collaborative planning with the crowd.

Marsh, S., and Dibben, M. R. 2003. The role of trust in information science and technology. *Annual Review of Information Science and Technology* 37(1):465–498.

Nguyen, A.; Yosinski, J.; and Clune, J. 2015. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *CVPR*, 427–436.

Norman, D. 1988. *The psychology of everyday things*. Basic Books.

Norman, D. A. 2014. Some observations on mental models. In *Mental models*. Psychology Press. 15–22.

O'Cane, S. 2018. Tesla can change so much with over-the-air updates that its messing with some owners heads. `www.theverge.com/2018/6/2/17413732/tesla-over-the-air-software-updates-brakes`.

Reber, A. S. 1989. Implicit learning and tacit knowledge. *Journal of experimental psychology: General* 118(3):219.

Spring, M. J. 2005. Techniques for maintaining compatibility of a software core module and an interacting module. US Patent 6,971,093.

Tsantilis, E. 2009. Method and system to monitor software interface updates and assess backward compatibility. US Patent 7,600,219.

Wang, D.; Khosla, A.; Gargeya, R.; Irshad, H.; and Beck, A. H. 2016. Deep learning for identifying metastatic breast cancer. *arXiv preprint arXiv:1606.05718*.

Wiens, J.; Guttag, J.; and Horvitz, E. 2016. Patient risk stratification with time-varying parameters: a multitask learning approach. *JMLR* 17(1):2797–2819.

Zhou, D.; Bousquet, O.; Lal, T. N.; Weston, J.; and Schölkopf, B. 2004. Learning with local and global consistency. In *NIPS*, 321–328.

Zhou, J.; Arshad, S. Z.; Luo, S.; and Chen, F. 2017. Effects of uncertainty and cognitive load on user trust in predictive decision making. In *IFIP Conference on Human-Computer Interaction*, 23–39. Springer.