Intelligible Artificial Intelligence

Daniel S. Weld[1,2]
[1]Microsoft Research
Redmond, Washington

Gagan Bansal[1]
[2]Paul G. Allen School of Computer Science & Engineering
University of Washington
Seattle, Washington

ABSTRACT

Since Artificial Intelligence (AI) software uses techniques like deep lookahead search and stochastic optimization of huge neural networks to fit mammoth datasets, it often results in complex behavior that is difficult for people to understand. Yet organizations are deploying AI algorithms in many mission-critical settings. In order to trust their behavior, we must make it intelligible — either by using inherently interpretable models or by developing methods for explaining otherwise overwhelmingly complex decisions by local approximation, vocabulary alignment, and interactive dialog.

KEYWORDS

HCI, artificial intelligence, machine learning, explanation, interpretability

1 INTRODUCTION

Artificial Intelligence (AI) systems have reached or exceeded human performance on many circumscribed tasks. As a result, they are increasingly deployed in mission critical roles — credit scoring, predicting if a bail candidate will commit another crime, selecting the news we read on social networks, and soon we may have self-driving cars. Unlike other mission critical software, it's very hard to test AI systems, which are extraordinarily complex. AI decisions are context specific, often based on thousands or millions of factors. Typically, AI behaviors are generated by searching vast action spaces or learned by the opaque optimization of mammoth neural networks operating over inhuman amounts of training data. Almost by definition there is no deterministic method for accomplishing the AI's task.

Unfortunately, much computer-produced behavior is *alien* — it can fail in unexpected ways. This lesson is most clearly seen in the performance of the latest deep neural network image analysis systems. While their accuracy at object-recognition on naturally occurring pictures is extraordinary, imperceptible changes to the input images can lead to erratic predictions, as shown in Figure 1. Why is the recognition system so brittle, making different predictions for apparently identical images? Unintelligible behavior isn't limited to machine learning — many AI programs perform search-based lookahead and inference whose complexity exceeds human abilities to verify. However, if we don't understand a system's behavior, it is dangerous to trust it.

Yet it's crucial that we be able to trust deployed systems, and this requires improving robustness and developing ways to make their reasoning intelligible [7]. As a result, researchers have increasingly focused on ways for enabling AI system to *explain* their learned models and reasoning — so that humans can understand and trust their operation. Such explanations may help reveal situations where biased training data has resulted in a learned model that unfairly discriminate against certain social groups [14]. Furthermore, in cases where the data and learned-models are accurate or where search algorithms have provably-correct properties, explanations can give humans useful insights about the domain at hand [6].

But it's remarkably hard to specify what makes an explanation "intelligible" or to navigate the tension between a concise account and an accurate one. We discuss desiderata for explanations later in this article, but in brief, a good explanation should allow the human to see what factors *caused* the AI's action and to predict how changes in the situation would have led to alternative behaviors.

We focus on two high-level approaches to building intelligible AI: 1) ensure that the underlying reasoning or learned model is inherently *interpretable* (aka transparent, intelligible), *e.g.*, learning a linear model over a small number of well-understood features, and 2) generate post hoc explanations of more complex, *inscrutable* reasoning, such as the predictions of complex neural networks or deep-lookahead search. Using an interpretable model has the benefit of veracity — in theory, the human can see exactly what the model is doing. But this approach has the drawback that interpretable methods may not perform nearly as well as more complex methods, such a deep neural networks. Conversely, the post-hoc approach has the benefit that it promises to apply to whichever AI technique is currently delivering the best performance, but it raises additional concerns since the explanation is inherently *different* from the way the AI system is actually operating. How can a person trust that the explanation reflects the essence of the underlying decision and isn't sweeping important details under the rug? We argue that the answer is making the explanation system *interactive* so the user can drill down until they are satisfied with their understanding.

Since the key challenge for designing intelligible AI is communicating a complex computational process to a human, it requires interdisciplinary skills, including HCI as well as AI and machine learning expertise. Furthermore, since the nature of explanation has been long studied by philosophy and psychology, these fields should also be consulted.

The next section reviews the common methodology for training and evaluating machine learning models, enumerating reasons why one might wish for more understanding than is provided by traditional performance metrics. Next, we describe the benefits and limitations of GA^{2}M, a class of interpretable ML models. Section 4 discusses methods and challenges for generating explanations of inscrutable models, including *blackbox* models such as those provided by remotely managed cognitive services. We close by sketching a vision for *interactive* explanation systems.

2 WHY DOUBT LEARNED MODELS?

We've argued that if AI is interpretable or can explain itself, then this will increase humans' understanding of the system. But what type

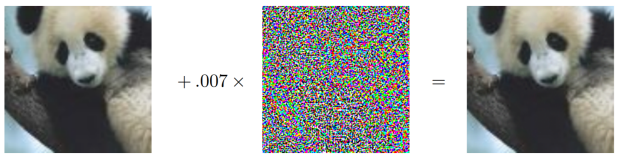


Figure 1: Figure 1 from Goodfellow *et al.* [10], demonstrating adversarial example generation applied to the GoogLeNet [39] image recognizer, trained on ImageNet. Adding an imperceptibly small vector changes GoogLeNet's classification of the image.

of understanding is desired? Different people will likely have different questions about the system, *e.g.*, depending on their relationship to the AI system. The concerns of someone whose mortgage application was denied due to a FICO score probably differ from the developer or data scientist debugging such an ML system. In Section 5 we return to the issue of customized explanations, but for now we focus on a developer's concerns in the context of machine learning (a single type of AI). Quoting Lipton, a learned model "may be considered trustworthy in the sense that there is no expected cost of relinquishing control" [24]. It is natural, then, to ask what might *cause* a learned model to act unexpectedly, incurring a cost.

2.1 Traditional Machine Learning Methodology

Since practitioners have been evaluating machine learning for many decades, it's natural to wonder how current methodology could fall short. To see this, we review traditional ML performance evaluation to see what might be missing.[1] A developer uses machine learning because they want to create a model that can predict the future. Suppose the objective is to predict the chance that a patient will soon die from pneumonia; we'll call this the *target*, denoted y. The prediction should be made based on easily observable input *features*, like age, presence of a cough, *etc.*., so the developer next decides on this set, denoted $X = \{x_1, \ldots, x_n\}$. The input and target features can be numbers or discretely typed.

Supervised learning starts with a labeled corpus of data, $\langle X, y \rangle$, such as a large database of medical cases. The developer starts by splitting this data into *training* and *test* sets. They also choose a class of *models* (mathematical functions), such as decision trees or deep neural networks; these models are parameterized, and once the parameter values are chosen, the model will be a predictor — a function mapping the input features to the output class. Machine learning *training* is an optimization process, where the learning algorithm searches for a set of model parameters that minimizes a *loss function*, such as the number of misclassified training examples.

The developer performs the optimization on the training set and then evaluates the performance (*e.g.*, accuracy, precision, recall, *etc.*) on the test set only after committing to the features, model class, optimization algorithm *etc.*. Since the model has never seen the test data, it can be used to give an independent estimate of the learned model's performance.

[1] Our treatment here is by necessity an over-generalization. Furthermore, there are alternative methodologies, such as *machine teaching* [34] which surmount some of the issues we discuss. Yet the approach described here is in wide practice.

2.2 Model Assessment

This methodology seemingly gives an impartial and quantitative assessment of the learner. If the learned model's performance on the test set is low, then one knows that there is a problem. But what if test-set performance is extremely high, say 100%. Isn't that enough to earn trust? Why might one also wish to understand *why* the model is making each prediction? There are at least five reasons, of which three are technical:

- **Inadequate Loss Function** In some situations, even 100% perfect performance may not be good enough, if the performance metric is flawed or incomplete due to the difficulty of specifying it explicitly. For example, as Lipton observed [24], "an algorithm for making hiring decisions should simultaneously optimize for productivity, ethics and legality," but how does one express this trade off? Other examples include balancing training error while uncovering causality in medicine and balancing accuracy and fairness in recidivism prediction [14]. In the case of recidivism prediction, a simplified loss function such as accuracy combined with historically biased training data may result uneven performance for different groups (*e.g.*, people of color).
- **Overfitting** A learned model may be optimized to perform well on a held-out validation set. However, repeated testing may cause the model to overfit on this set. Most practitioners know that they shouldn't repeat a measurement on the test set, but it often seems necessary and human nature finds a way to justify it, *e.g.*, "The new pipeline for computing input features fixes a bug." The irreversible slide towards overfitting is often imperceptible. Only a few members of the development team may even know of the risk.
- **Distributional Drift** A deployed model may perform poorly in the wild, especially if there is a difference between the train (or test) distribution and that encountered during deployment. In fact, the deployment distribution may change over time. For example, there may be a feedback loop as a result of deployment. Indeed, this is common in adversarial domains such as spam detection, online ad pricing, and search engine optimization.

These problems can lead to learned models which appear to perform well on the test set, but behave poorly, perhaps disastrously, during deployment. Making the AI intelligible could help users detect these technical problems. But intelligible AI is important for social reasons too:

- **User Acceptance.** Studies have shown that users are more likely to accept algorithmic decisions if they are accompanied by an explanation [19].
- **Legal.** A final reason stems from the legal system, for example the European Union's GDPR legislation decreeing the right to an explanation [11] or the AI owner's concerns about liability.

Figure 2: The dashed blue shape indicates the space of possible mistakes made by humans. The red shape denotes the AI's mistakes; it's smaller size indicates a net reduction in the number of errors. The gray region denotes AI-specific mistakes a human would never make. Despite reducing the total number of errors, a deployed model may create new areas of liability (gray), necessitating explanations [5].

A related reason is assessing legal liability. A deployed model (e.g., self-driving cars) may introduce new areas of liability by causing accidents unexpected from a human operator, such as the "AI-specific error" region in Figure 2. Auditing such situations, *e.g.*, for the purpose of liability, requires explaining the model's decisions.

In the rest of this paper, we focus on two broad approaches towards intelligible AI: machine learning of inherently interpretable models and methods for explaining the predictions of opaque, "black-box" learned models.

3 INHERENTLY INTERPRETABLE MODELS

If the domain contains a moderate number of features that have intuitive meanings known to the people involved, then it is sometime possible to build a model which is considered *interpretable*. Few computing researchers have tried to formally define what is an interpretable model [8], but one suggested criterion is *human simulatability* [24] — can a human easily predict the model's output for a given input. Under this definition, sparse linear models are more interpretable than dense or non-linear models.

Philosophers, such as Hempel and Salmon have long debated the nature of explanation. Lewis [23, p 217] summarizes "To explain an event is to provide some information about its causal history." But many causal explanations may exist. The fact that event C causes E is best understood relative to an imagined counterfactual scenario, where absent C, E would *not* have occurred; furthermore, C should be minimal — an intuition known to early scientists, such as William of Occam and formalized by Halpern and Pearl [13].

Following this logic, we suggest (instead of human simulatability) a better criterion is the ability to answer counterfactuals, known as "What-if" questions. Specifically, we say that a model is interpretable to the degree that a human can predict how a *change* to a feature, *e.g.* a small increase to its value, will *change* the model's output. Note that if one can simulate the model, predicting its output, then one can predict the effect of a change — but not vice versa.

Linear models are especially interpretable under this definition because they allow answering counterfactuals. For example, consider a naive Bayes, unigram model for sentiment analysis, whose objective is to predict the emotional polarity (positive or negative) of a textual passage. Even if the model is large, combining evidence from the presence of thousands of words, one can see the effect of a given word by looking at the sign and magnitude of the corresponding weight. This answers the question "What-if the word had been omitted?" Similarly, by comparing the weights associated with two words, one can predict the effect on the model of substituting one for another.

Unfortunately, linear models have limited utility because they often result in poor accuracy. We next discuss another class of interpretable models that perform better in practice.

3.1 Generalized Additive Models

Generalized additive models (GAMs) are the class of machine learning models that relate a set of features to a target with a linear combination of (potentially-nonlinear) single-feature models called *shape functions* [26]. For example, if y represents the target and $\{x_1, \ldots .x_n\}$ represents the features then a GAM model takes the form $y = \beta_0 + \sum_j f_j(x_j)$, where the f_is denote the shape functions and the target y is computed by summing up single-feature *terms*. Popular shape functions include non-linear functions such splines and decision trees. If the shape functions are restricted to be linear, then a GAM reduces to be a linear model.

In general, GAMs are more expressive than linear models and have been shown to achieve high accuracy on many real-world problems [6]. In fact, the performance of GAMs can be further improved by including terms that model interaction between features. GA^{2}M models are an extension of GAM models that in addition to the single-feature terms include terms for pairwise interactions between features. GA^{2}M models take the following form:

$$y = \beta_0 + \sum_j f_j(x_j) + \underbrace{\sum_{i \neq j} f_{ij}(x_i, x_j)}_{\text{pairwise terms}} \quad (1)$$

Caruana et al. observed that for domains that contain a moderate number of semantic features GA^{2}M models achieve performance that is competitive with uninterpretable models, such as random forests and neural networks, whilst remaining intelligible [6]. Lou et al. observed that among many methods available for learning GA^{2}M models, the version where the shape functions are bagged shallow regression trees learned via gradient boosting achieve the highest accuracy[26]. Other approaches include spline regressions where a piece wise polynomial function is fit via least squares estimate. They further noticed that while spline regression produces smoother line plots, tree-based methods better model abrupt changes in prediction such as discrete thresholds.

Both GAM and GA^{2}M are considered interpretable because the model's learned behavior can be easily understood by examining or visualizing the contribution of terms (individual or pairs of features) to the final prediction. For example, Figure 3 depicts the contribution (log odds) of a subset of terms to total risk for a GA^{2}M model trained to predict a patient's risk of dying due to pneumonia. A positive contribution increases risk, whereas a negative contribution decreases risk. For example, 3(a) clearly shows how the patient's age affects predicted risk. While the risk is low and steady for young patients (e.g., age < 20), it increases rapidly for older patients (age > 67). Interestingly, the model shows a sudden increase at age 86; perhaps this results from less aggressive care by doctors for patients "whose time has come." Even more surprising is the sudden drop for patients over 100. This might be another social effect — once a patient reaches the magic "100" they get more aggressive care. One benefit of an interpretable model is their ability to highlight these issues, spurring deeper analysis.

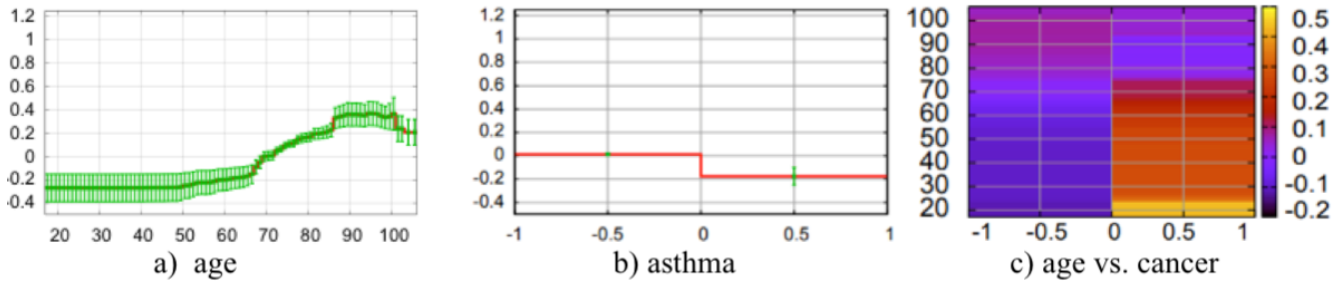


Figure 3: A part of Figure 1 from [6] showing 3 (of 56 total) components for a GA^{2}M model, which was trained to predict a patient's risk of dying from pneumonia. The two line graphs depict the contribution of individual features to risk: a) patient's age, and b) boolean variable asthma. The y-axis denotes its contribution (log odds) to predicted risk. The heat map, c, visualizes the contribution due to pairwise interactions between age and cancer rate.

Figure 3(b) illustrates another surprising aspect of the learned model — apparently a history of asthma, a respiratory disease, *decreases* the patients risk from pneumonia! This is counter-intuitive to any physician, who recognizes that asthma, in fact, should increase such risk. When Caruana et al. checked the data, they concluded that the lower risk could be explained by the fact that asthma patients typically have been receiving timely and aggressive therapy for lung issues. Therefore, although the model was highly accurate on the test set, it would likely fail, dramatically underestimating the risk to a patient with asthma who had not been previously treated for the disease. A domain expert can fix such erroneous patterns learned by the model by the setting the weight of the asthma term to zero. In fact, GAMs allow users to provide much more comprehensive feedback to the model by using a GUI to redraw a line graph for a term in the model [6]. An alternative remedy might be to introduce a new feature to the model, representing whether the patient had been recently seen by a pulmonologist. After adding this feature, which is highly correlated with asthma, the learned model would likely change its prediction and record that asthma increases risk from pneumonia. There are two more takeaways from this anecdote. First, the *absence* of an important feature in the data representation can cause *any* AI system to learn unintuitive behavior for another, correlated feature. Secondly, if the learner is interpretable, then this unintuitive behavior is immediately apparent, allowing appropriate skepticism (despite high test accuracy) and easier debugging.

Recall that GA^{2}M are more expressive than simple GAM models by including pairwise terms. Figure 3(c) depicts such a term for the features age and cancer. This explanation indicates that among the patients who have cancer, the younger ones are at higher risk. This is likely because the patients who develop cancer when they are young are probably critically ill. Again, since doctors can readily inspect these terms, they are made aware if the learner develops unexpected conclusions.

3.2 Limitations of GA^{2}M Models

GA^{2}Ms decompose their prediction into effects of individual terms which can be visualized. However, if the users are confused about the meaning of the terms, they won't understand the model nor be able to ask meaningful "What-if" questions. Furthermore, if there are too many features, the sheer complexity of the model may overwhelm the user. Lipton notes that the effort required to simulate some models (such as decision trees), may grow logarithmically with the number of parameters [24], but for GA^{2}M the number of visualizations to inspect may increase quadratically. Several methods may help users cope with this complexity — for example the terms might be ordered by importance; however, it's not clear how importance should be computed. Possible methods include: 1) conduct an ablation analysis to compute influence of terms on model performance, and 2) compute the maximum contribution of term as seen in the training samples. Alternatively, a human expert might be able to group the terms semantically to facilitate perusal.

However, when the number of features grows into the millions, which happens when dealing with classifiers over text, audio, image and video data, existing interpretable models don't perform nearly as well as inscrutable methods, like deep neural networks and giant boosted decision forests. Since these models may combine millions of features in complex, nonlinear ways, they are beyond human capacity to simulate. The next section discusses nascent methods for understanding these opaque learned models, but one should strongly consider interpretable models, such as GA^{2}M, for problems with a moderate number of semantically meaningful features.

4 EXPLAINING INSCRUTABLE MODELS

We now consider the case of an inscrutable, learned model; such as a neural network, where one has access to a myriad learned parameters but can't reasonably interpret them, or a blackbox model, such as an API like Microsoft Cognitive Services, which uses machine learning to provide image-recognition capabilities, but doesn't allow inspection of the underlying model. How can an AI system best convey aspects of this model to a human?

Grice introduced four rules characterizing cooperative communication, which hold for good explanations [12]. The maxim of quality: be truthful and only say things which are supported by evidence. The maxim of quantity: give as much information as is needed, and no more. The maxim of relation: only say things that are relevant to the discussion. The maxim of manner: avoid ambiguity; be as clear and as brief as possible.

Miller summarizes decades of work by psychological research, noting that explanations are *contrastive*, i.e. of the form "Why P rather than Q?" The event in question, P, is termed the *fact* and Q is called the *foil* [29]. Often the foil isn't explicitly stated, but it's

crucially important to the explanation process. For example, consider the question "Why did you predict that the image depicts an indigo bunting?" An explanation which points to the blue color is implicitly assuming that the foil is another bird, such as a chickadee. But perhaps the questioner is wondering why the recognizer didn't predict a pair of denim pants; in this case a better explanation might highlight the presence of wings and a beak. Clearly, an explanation targeted to the wrong foil will be unsatisfying, but the nature and sophistication of a foil can depend on the end user's expertise, and hence the ideal explanation will be different for different people [8]. For example, to verify that a ML system is fair, an ethicist might come up with more complex foils than a data scientist. Most implemented ML explanation systems have restricted their attention to elucidating the behavior of a binary classifier, where there is only one possible foil choice, but as we seek to explain multi-class systems addressing this issue will become essential.

This section summarizes two key challenges that must be addressed by practical systems that seek to generate explanations for complex, learned models where minimal counterfactuals are poorly defined. Specifically, we consider the simplicity / fidelity tradeoff and discuss the challenges of finding an appropriate vocabulary to express an explanation.

4.1 Explanation as an Approximation

A good explanation of an event is 1) simple and easy to understand, and 2) faithful (accurate), conveying the true cause of the event. Unfortunately, these two criteria are almost always in conflict. Consider the predictions of a deep neural network with millions of nodes — a complete and accurate trace of the network's prediction would be much too complex to understand, but *any* simplification sacrifices accuracy. Finding a good explanation, therefore, requires balancing the competing desires of comprehensibility and fidelity. Lakkaraju *et al.* [22] suggest formulating an explicit optimization of this form and propose an approximation algorithm for generating global explanations in the form of compact sets of if-then rules. Ribeiro *et al.* describe a similar optimization algorithm that balances precision and coverage in its search for summary rules [32].

Another way to simplify the explanation of a learned model is to make it relative to a single input query. Such explanations, which are termed *local* [31], are akin to a doctor explaining the reasons behind her diagnosis of a single patient rather than trying to communicate all of her medical knowledge. Local explanations are common practice in AI systems. For example, early rule-based expert systems included explanation systems that augmented a trace of the system's reasoning — on a particular case — with background knowledge [38]. Recommender systems, which were one of the first deployed uses of machine learning, also induced demand for explanations of their specific recommendations; the most satisfying answers combined justifications based on the user's previous choices, ratings of similar users, and features of the items being recommended [30].

In many cases, however, even a local explanation is too complex to understand without approximation. In such a case the key challenge is deciding which details to omit in order to create a simple, *explanatory model*. This is a question long studied by psychologists, who determined that several criteria are prioritized for inclusion in an explanation: necessary causes (vs. sufficient), intentional actions (vs. those taken without deliberation), proximal causes (vs. distant), details that distinguish between fact and foil, and abnormal features [29].

According to Lombrozo, humans prefer explanations that are simpler (*i.e.*, contain fewer clauses), more general, and coherent (*i.e.*, consistent with what the human's prior beliefs) [25]. In particular, she observed the surprising result that humans preferred simple (one clause) explanations to conjunctive explanations — even when the probability of the conjunction was higher than the single clause) [25]. These results raise important questions about the purpose of explanations in an AI system. Is an explanation's primary purpose to convince a human to *accept* the computer's conclusions (perhaps by presenting a simple, plausible, but unlikely explanation) or to *educate* the human about the most likely true situation? Tversky, Kahneman, and other psychologists have documented many cognitive biases that lead humans to incorrect conclusions; for example, people reason incorrectly about the *probability* of conjunctions, with a concrete and vivid scenario deemed more likely than abstract situation which strictly subsumes it [18]. Should an explanation system exploit human limitations or seek to protect us from them?

Other studies raise an additional complication about how to communicate a system's uncertain predictions to human users; Koehler found that simply presenting an explanation for a fact makes people think that it is more likely to be true [19]. Furthermore, explaining a fact in the same way as previous facts have been explained amplifies this effect [36].

4.2 Locally-Approximate Explanations

In recent years, researchers have built systems that can generate post-hoc explanations of a learned classifier, including those typically considered noninterpretable, such as deep neural networks. Indeed, there are too many such systems to enumerate; many are domain- or classifier-specific. For example, Simonyan *et al.* describe methods for explaining CNN ImageNet classifiers by calculating class appearance models and saliency maps [35]. Recurrent neural models for translating text or answering questions often display heatmaps to show which parts of the text are getting the model's attention [2]. Ribeiro *et al.*'s LIME system [31] is an interesting example, because it has the additional benefit of explaining *black-box* models whose structure and parameters are completely unknown, such as models which are deployed as an API by a third-party.

While LIME can generate explanations for the predictions of any learned model, it requires the developer to provide two additional inputs: 1) a semantically meaningful set of features X' that can be computed from the original features, and 2) an interpretable learning algorithm, such as a linear classifier (or a GA^{2}M), which it will use to generate an explanation in terms of the X'.

The intuition is shown in Figure 4. Given an instance to explain, shown as the bold red cross, LIME randomly generates a set of similar instances and uses the blackbox classifier, f, to predict their values (shown as the red crosses and blue circles). These predictions are weighted by their similarity to the input instance and used to train a new, simpler, *interpretable* classifier, shown as the linear decision boundary, using X', the smaller set of semantic features. The

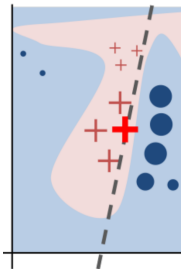
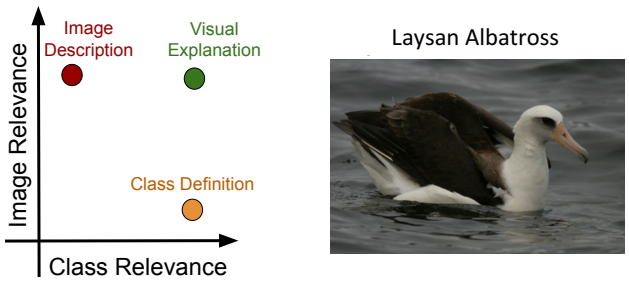


Figure 4: The intuition un structing an approximate, l "The black-box model's co known to LIME) is represer which cannot be approxim bold red cross is the instanc instances, gets predictions proximity to the instance b by size). The dashed line i locally (but not globally) fa



Description: This is a large bird with a white neck and a black beak in the water.

Class Definition: The *Laysan Albatross* is a large seabird with a hooked yellow beak, a black back, and a white belly.

Visual Explanation: This is a *Laysan Albatross* because this bird has a hooked yellow beak white neck and black back.

Figure 5: A visual explanation, taken from [15]: "Visual explanations are both image relevant and class relevant. In contrast, image descriptions are image relevant, but not necessarily class relevant, and class definitions are class relevant but not necessarily image relevant."

user is given the interpretabl
this *explanation model* [27] is
of f, it hopefully is a good loc
the vicinity of the instance be

Ribeiro *et al.* tested LIME o
explained the predictions of a
classifier by converting the pi
"super-pixels," by running an
to identify regions in the inp
some these regions when gen
provides no formal guarantee
showed that LIME explanations helped people evaluate which of several classifiers would generalize better.

4.3 Choosing an Explanatory Vocabulary

Ribeiro *et al.*'s decision to explain image classification decisions in terms of pre-segmented regions of the photo illustrates the larger problem of determining an explanatory vocabulary. Clearly, it wouldn't make sense to try and identify the exact pixel that led to the decision — pixels are too low level a representation; they aren't semantically meaningful to people. In fact, the power of deep neural networks comes from the very fact that the hidden layers are trained to recognize latent features in a manner that seems to perform much better than previous efforts to define such features independently. Deep networks are inscrutable exactly because we don't know what those hidden features denote.

In order to explain the behavior of such models, however, we need to find some high-level abstraction over the input pixels that can be used to communicate the essence of the model. Ribeiro *et al.*'s decision to use an off-the-shelf image-segmentation system was pragmatic. On the plus side, the regions it selected are easily visualized and carry some semantic value. On the minus side, the choice of regions is done completely without regard to the way that the classifier is making the decision. If one wishes to explain a blackbox model, where there is no possible access to the classifier's internal representation, then there is likely no better option — any explanation will lack faithfulness.

However, if one has access to the classifier and is willing to tailor the explanation system to it, there are ways to choose a more meaningful vocabulary. One interesting method jointly trains a classifier with a natural-language, image captioning system [15]. The classifier uses training data that is labeled with the objects appearing in the image, while the captioning system is labeled with English sentences describing the appearance of the image. By training the systems jointly, the variables in the hidden layers may get aligned to semantically meaningful concepts even as they are being trained to provide discriminative power. This results in English descriptions of images that have both high image relevance (from the captioning training data) and high class relevance (from the object recognition training data) as shown in Figure 5.

While this method works well on many examples, the explanations sometimes include descriptions of details that aren't actually present in the image; newer approaches, such as phrase-critic methods, may create even better descriptions [16]. Another approach might determine if there are hidden layers in the learned classifier, that learn concepts correspond to something meaningful. For example, Zeiler and Fergus observed that certain layers may function as edge or pattern detectors [40]. Whenever one can identify the presence of such layers then it might be a good idea to prefer using them in the explanation. Bau *et al.*describe a automatic mechanism for matching CNN representations with semantically-meaningful concepts, using a large, labeled corpus of objects, parts, and texture Bau et al.; furthermore, using this alignment, their method can quantitatively score the interpretability of the CNN, potentially suggesting a way to optimize for intelligible models. Many challenges remain. As one example, it's not clear that there *are* good

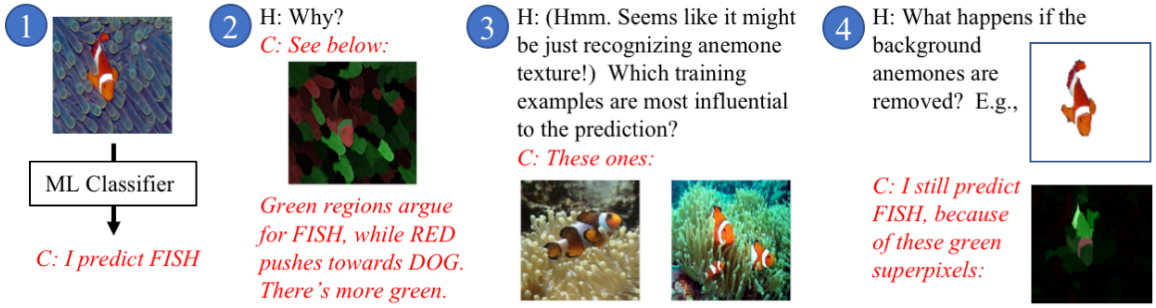


Figure 6: An example of an interactive explanatory dialog for gaining insight into a DOG/FISH image classifier. (For illustration, the questions and answers are shown in English, but our use of 'dialog' is meant to be suggestive. An interactive GUI might well be a better realization.)

ways to describe important, discriminative features, which are often intangible, *e.g.*, textures. A satisfying explanation may need to define new terms or combine language with other modalities, like patches of the image. Fortunately, research is moving quickly in this area.

4.4 Explaining Combinatorial Search

Most of the preceding discussion has focused on intelligible *machine learning*, which is just one type of artificial intelligence, but the same issues also confront systems based on deep-lookahead search. While many search algorithms have strong theoretical properties, true correctness depends on assumptions made when modeling the underlying actions [28], so a human might wish to question the agent's choices.

Consider a planning algorithm that has generated a sequence of actions for a remote, mobile robot. If the plan is short with a moderate number of actions, then the problem may be inherently intelligible, but larger search spaces will likely be cognitively overwhelming. In these cases, local explanations are a simplification technique that is helpful, just as it was when explaining machine learning. The vocabulary issue is likewise crucial — how does one succinctly summarize a complete search subtree abstractly? Depending on the choice of explanatory foil, different answers are appropriate [9]. Sreedharan et al. describe an algorithm for generating the minimal explanation that patches a human's partial understanding of a domain [37]. Many AI systems, such as AlphaGo [33], combine both deep search *and* machine learning; these will be especially hard to explain, since complexity arises from the interaction of combinatorics and a learned model.

5 TOWARDS INTERACTIVE EXPLANATION

The best choice of explanation depends on the audience. Just as a human teacher would explain physics differently to students who have or not yet had calculus, the technical sophistication and background knowledge of the recipient affects the suitability of a machine-generated explanation. Furthermore, an ML developer will have very different concerns than someone impacted by the ML's predictions, and hence they may need different types of explanations. Ideally, the explainer might build a model of the listener's background over the course of many interactions, as have been proposed by the intelligent tutoring systems community [1].

Even with an accurate user model, it's likely that the explanation won't answer all of the listener's questions. We conclude that an explanation system should be *interactive*, supporting follow-up questions and actions from the user. This matches results from the psychology literature showing that explanation is best thought of as a social process, a conversation between explainer and explainee [17, 29]. This perspective highlights the motivations and backgrounds of the participants. It also recalls Grice's maxims, described before, especially those of quantity and relation.

We envision an interactive explanation system that supports a number of different follow-up and drill-down actions, after presenting the human it's initial explanation:

- Redirecting the answer by changing the foil: "Sure, but why didn't you predict class C?"
- Asking for more detail (*i.e.*, a more complex explanatory model), perhaps while restricting the explanation to a subregion of feature space: "I'm only concerned about women over age 50..."
- Asking "What made you believe this?" to which the system might respond by displaying the labeled *training examples* that were most influential in reaching that decision, *e.g.*, ones identified by influence functions [20] or nearest neighbor methods.
- Changing the vocabulary by adding (or removing) a feature to the explanatory model, either from a predefined set or by using methods from machine teaching [34].
- Perturbing the input example to see the effect on both prediction and explanation. In addition to aiding understanding of the model, this action is useful if an affected user wishes to contest the initial prediction "But officer, one of those prior DUI's was overturned...?"
- Attempting to repair the model. A data scientist may wish to add new training examples, correct an erroneous label in existing data, specify a new feature, or change learning parameters and architecture. Here we expect to use affordances from interactive machine learning [4] and explanatory debugging [21].

To make these ideas concrete, Figure 6 illustrates a possible dialog as a user tries to understand the robustness of a deep neural dog / fish classifier built atop Inception v3 [39]. 1) The computer correctly predicts that the image depicts a fish. 2) The user requests an explanation, which is provided using LIME [31]. 3) The user is concerned that the classifier is paying more attention to the background than

to the fish itself, and asks to see the training data that influenced the classifier; the nearest neighbors are computed using influence functions [20]. While there are anemones in those images, it also seems as if the system is recognizing clownfish. 4) To gain confidence, the user edits the input image to remove the background, resubmits to the classifier and checks the explanation. While we plan to add additional actions to the interactive explanation system, the system has already provided useful insights.

6 FINAL THOUGHTS

The increasing deployment of complex AI systems increases the need for humans to be able to audit and understand their decisions. Depending on the complexity of the models involved, two approaches may be appropriate: 1) using an inherently interpretable model, or 2) adopting an inscrutably-complex model and generating post-hoc explanations. When learning a model over a medium number of human-interpretable features, one may get an excellent balance of performance and intelligibility with approaches like GA^{2}Ms. However, for problems with thousands or millions of features, performance requirements likely force the adoption of inscrutable methods such as deep neural networks or boosted decision forests. In these situations post-hoc explanations may be the only way to facilitate human understanding.

Research on explanation algorithms is developing rapidly with work on both local (instance-specific) explanations and global approximations to the learned model. A key challenge for all these approaches is discovery or construction of an explanation vocabulary — essentially a set of features used in the approximate explanation model. Results from psychology show that explanation is a social process and is best thought of as a conversation. As a result we advocate increased work on *interactive* explanation systems that support a wide range of follow-up actions which further this conversation. To spur rapid progress in this important field, we hope to see collaboration between researchers in multiple disciplines.

Acknowledgements: We thank S. Ameshi, R. Calo, R. Caruana, M. Chickering, O. Etzioni, J. Heer, R. Kambhamapti, E. Kamar, P. Simard, Mausam, C. Meek, M. Michelson, S. Minton, G. Ramos, M. Ribeiro, M. Richardson, P. Simard, J. Suh, J. Teevan, and T. Wu for helpful comments and conversations. This work was supported in part by the Future of Life Institute grant 2015-144577 (5388) with additional support from NSF grant IIS-1420667, ONR grant N00014-15-1-2774, and the WRF/Cable Professorship.

REFERENCES

[1] J. R Anderson, F. Boyle, and B. Reiser. 1985. Intelligent tutoring systems. *Science* 228, 4698 (1985), 456–462.
[2] D. Bahdanau, K. Cho, and Y. Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *ICLR*.
[3] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba. 2017. Network Dissection: Quantifying Interpretability of Deep Visual Representations. In *CVPR*.
[4] M. Brooks, S. Amershi, B. Lee, S. M Drucker, A. Kapoor, and P. Simard. 2015. FeatureInsight: Visual support for error-driven feature ideation in text classification. In *VAST*.
[5] R. Calo. 2014. The case for a federal robotics commission. (2014). https://www.brookings.edu/research/the-case-for-a-federal-robotics-commission/.
[6] R. Caruana, Y. Lou, J. Gehrke, P. Koch, M. Sturm, and N. Elhadad. 2015. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission.
[7] T. Dietterich. 2017. Steps Towards Robust Artificial Intelligence. *AI Magazine* 38, 3 (2017).
[8] F. Doshi-Velez and B. Kim. 2017. Towards A Rigorous Science of Interpretable Machine Learning. *ArXiv* (2017). arXiv:1702.08608
[9] M. Fox, D. Long, and D. Magazzeni. 2017. Explainable Planning. http://arxiv.org/abs/1709.10256
[10] I. J. Goodfellow, J. Shlens, and C. Szegedy. 2014. Explaining and Harnessing Adversarial Examples. *ArXiv* (2014). arXiv:1412.6572
[11] B. Goodman and S. Flaxman. 2016. European Union regulations on algorithmic decision-making and a "right to explanation". *ArXiv* (2016). arXiv:1606.08813
[12] P. Grice. 1975. *Logic and Conversation.* 41–58.
[13] J. Halpern and J. Pearl. 2005. Causes and explanations: A structural-model approach. Part I: Causes. *The British journal for the philosophy of science* 56, 4 (2005), 843–887.
[14] M. Hardt, E. Price, and N. Srebro. 2016. Equality of opportunity in supervised learning. In *NIPS*.
[15] L. Hendricks, Z. Akata, M. Rohrbach, J. Donahue, B. Schiele, and T. Darrell. 2016. Generating visual explanations. In *ECCV*.
[16] L. A. Hendricks, R. Hu, T. Darrell, and Z. Akata. 2017. Grounding Visual Explanations. *ArXiv* (2017). arXiv:1711.06465
[17] D. Hilton. 1990. Conversational processes and causal explanation. *Psychological Bulletin* 107, 1 (1990), 65.
[18] D. Kahneman. 2011. *Thinking, fast and slow.* Farrar, Straus and Giroux, New York. http://a.co/hGYmXGJ
[19] Derek J Koehler. 1991. Explanation, imagination, and confidence in judgment. *Psychological bulletin* 110, 3 (1991), 499.
[20] P. Koh and P. Liang. 2017. Understanding black-box predictions via influence functions. In *ICML*.
[21] T. Kulesza, M. Burnett, W. Wong, and S. Stumpf. 2015. Principles of explanatory debugging to personalize interactive machine learning. In *IUI*.
[22] H. Lakkaraju, E. Kamar, R. Caruana, and J. Leskovec. 2017. Interpretable & Explorable Approximations of Black Box Models. *KDD-FATML* (2017).
[23] D. Lewis. 1986. Causal explanation. *Philosophical Papers* 2 (1986), 214–240.
[24] Zachary Chase Lipton. 2016. The Mythos of Model Interpretability. In *ICML Workshop on Human Interpretability in ML*.
[25] T. Lombrozo. 2007. Simplicity and probability in causal explanation. *Cognitive psychology* 55, 3 (2007), 232–257.
[26] Y. Lou, R. Caruana, and J. Gehrke. 2012. Intelligible models for classification and regression. In *KDD*.
[27] S. Lundberg and S. Lee. 2017. A unified approach to interpreting model predictions. *NIPS* (2017).
[28] J. Mccarthy and P. Hayes. 1969. Some Philosophical Problems from the Standpoint of Artificial Intelligence. In *Machine Intelligence*. 463–502.
[29] T. Miller. 2017. Explanation in artificial intelligence: Insights from the social sciences. *ArXiv* (2017). arXiv:1706.07269
[30] A. Papadimitriou, P. Symeonidis, and Y. Manolopoulos. 2012. A generalized taxonomy of explanations styles for traditional and social recommender systems. *Data Mining and Knowledge Discovery* 24, 3 (2012), 555–583.
[31] M. Ribeiro, S. Singh, and C. Guestrin. 2016. Why Should I Trust You?: Explaining the Predictions of any Classifier. In *KDD*.
[32] M. Ribeiro, S. Singh, and C. Guestrin. 2018. Anchors: High-Precision Model-Agnostic Explanations. In *AAAI*.
[33] D. Silver, A. Huang, C. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, et al. 2016. Mastering the game of Go with deep neural networks and tree search. *nature* 529, 7587 (2016), 484–489.
[34] P. Simard, S. Amershi, D. Chickering, A. Pelton, S. Ghorashi, C. Meek, G. Ramos, J. Suh, J. Verwey, M. Wang, and J. Wernsing. 2017. Machine Teaching: A New Paradigm for Building Machine Learning Systems. *ArXiv* (2017). arXiv:1707.06742
[35] K. Simonyan, A. Vedaldi, and A. Zisserman. 2013. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. *ArXiv* (2013). arXiv:1312.6034
[36] S. Sloman. 1997. Explanatory coherence and the induction of properties. *Thinking & Reasoning* 3, 2 (1997), 81–110.
[37] S. Sreedharan, S. Srivastava, and S. Kambhampati. 2018. Hierarchical Expertise-Level Modeling for User Specific Robot-Behavior Explanations. *ArXiv e-prints* (Feb. 2018). arXiv:1802.06895
[38] W. Swartout. 1983. XPLAIN: a system for creating and explaining expert consulting programs. *Artificial Intelligence* 21, 3 (1983), 285 – 325.
[39] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. 2016. Rethinking the Inception Architecture for Computer Vision. In *CVPR*.
[40] M. Zeiler and R. Fergus. 2014. Visualizing and understanding convolutional networks. In *ECCV*.