Intuitively Assessing ML Model Reliability through Example-Based Explanations and Editing Model Inputs

Harini Suresh
hsuresh@mit.edu
MIT CSAIL

Kathleen M. Lewis
kmlewis@mit.edu
MIT CSAIL

John V. Guttag
guttag@mit.edu
MIT CSAIL

Arvind Satyanarayan
arvindsatya@mit.edu
MIT CSAIL

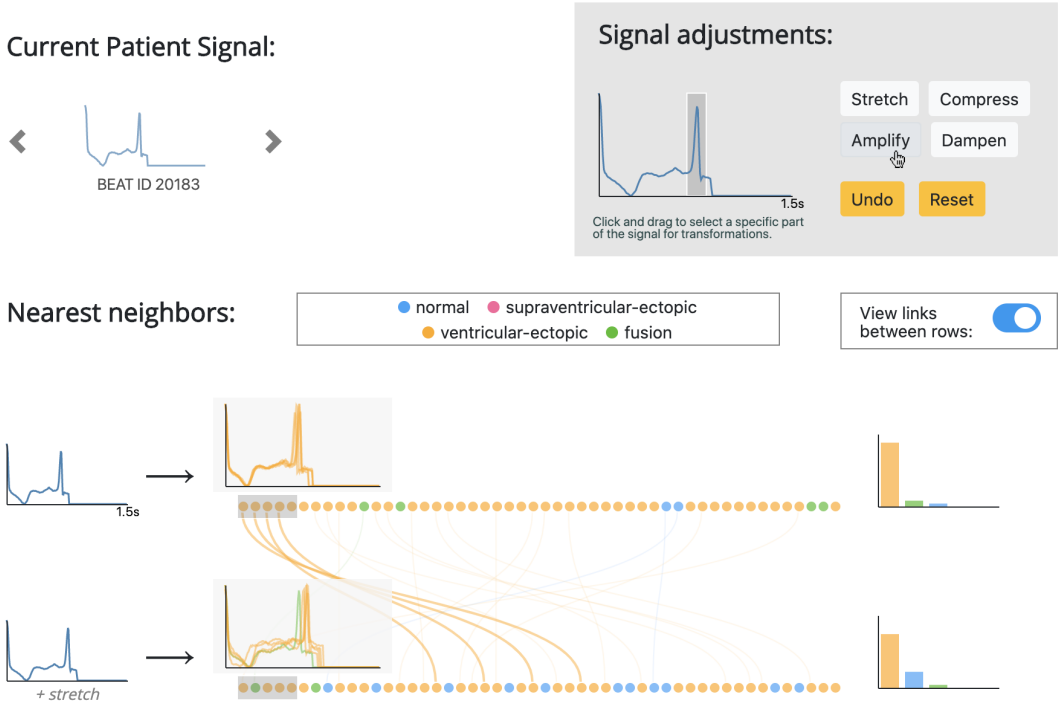


Figure 1: An example of the proposed interface for an electrocardiogram (ECG) case study. The output of the machine learning model consists of raw and aggregate information about the input's nearest neighbors. With the editor in the top right, the user can apply meaningful manipulations to the input and see how the output changes.

ABSTRACT

Interpretability methods aim to help users build trust in and understand the capabilities of machine learning models. However, existing approaches often rely on abstract, complex visualizations that poorly map to the task at hand or require non-trivial ML expertise to interpret. Here, we present two interface modules to facilitate a more intuitive assessment of model reliability. To help users better characterize and reason about a model's uncertainty, we visualize raw and aggregate information about a given input's nearest neighbors in the training dataset. Using an interactive editor, users can manipulate this input in semantically-meaningful ways, determine the effect on the output, and compare against their prior expectations. We evaluate our interface using an electrocardiogram beat classification case study. Compared to a baseline feature importance interface, we find that 9 physicians are better able to align the model's uncertainty with clinically relevant factors and build intuition about its capabilities and limitations.

CCS CONCEPTS

• **Applied computing** → *Health care information systems*; • **Human-centered computing** → **Visualization**; *Empirical studies in interaction design.*

KEYWORDS

interpretability, machine learning, visualization, k-nearest neighbors, example-based explanations

1 INTRODUCTION

Machine learning (ML) systems are being developed and used for a broad range of tasks, from predicting medical diagnoses [24] to informing hiring decisions [3]. While some of these systems have demonstrated excellent performance on in-domain accuracy using test or benchmark datasets, many are intended to be part of a larger sociotechnical process involving human decision-makers. In these cases, in-domain accuracy is not enough to guarantee good outcomes – the people using a particular system must also understand its limitations and modulate their trust appropriately [49]. *Model interpretability*, which is intended to help people understand how a particular ML model is working in general or on a case-by-case basis, can play an important role in this.

Many of the existing approaches to model interpretability, however, are intended for people with a non-trivial amount of ML expertise, and are often only used in practice by ML developers [6]. While tools for developers are certainly needed, the people who will actually be interpreting and acting upon the model predictions to guide decisions are often a distinctly different set of users. Even methods that are intended to be simpler and more understandable to such users, such as reporting feature weights or displaying more information about the model and dataset, have not improved decision-making in experimental studies [8, 30, 43, 53].

In response, we introduce two interface modules to facilitate more intuitive assessment of model reliability. The first module, motivated by the effectiveness of examples [44] and case-based reasoning [1] in human problem-solving, uses k-nearest neighbors (KNN) to visualize the model's output grounded in examples familiar to the user. Our visualization comprises three components: a unit visualization of individual neighbors that encodes their class and distance from the input, an overlaid display of the raw input of each neighbor, and a histogram of the neighbors' class labels. These views give the user an insight into the model's certainty with respect to the given input. By interactively examining individual neighbors, the user can investigate questions like whether the variation in the neighbors is natural given the domain, or if it indicates that the prediction is unreliable; whether the commonalities amongst neighbors align with domain knowledge of what should or should not be similar; or whether seeing examples reveals limitations or biases in the data of which they were previously unaware.

The second module allows users to interactively modify the input and see how the model's output changes. The user can modify the input by applying *domain-specific* transformations that correspond to meaningful manipulations of the data. Users can modify the input to test hypotheses about the model's reasoning, check that it aligns with domain expectations, and ensure that it is not overly sensitive to small input modifications that should be class-preserving.

Together, the KNN and editor modules provide a rich suite of information that facilitates intuitive assessment of a model's reasoning and reliability. We evaluate the effectiveness of our proposed interface modules through a medical case study of classifying electrocardiogram (ECG) heartbeats with different types of irregularities. This case study allows us to perform an application-grounded evaluation with users who closely resemble real-world decision-makers, and who have some degree of prior knowledge and investment in the domain. We conducted think-aloud studies with 9 physicians/medical students, observing the way they interacted with our interface as well as a feature importance baseline. By visualizing the nearest neighbor waveforms overlaid on one another, participants were able to intuitively understand prediction reliability by assessing the variation amongst neighbors — for example, distinguishing between variation that was understandable because it reflected natural ambiguities in the task versus variation that was more indicative of the model not learning the right features of the waveform. In contrast, when participants used the baseline interface and did not see nearest neighbors, they often rationalized incorrect predictions. Moreover, by exploring neighbors from different classes, participants were consistently able to relate the model's uncertainty to clinically-relevant concepts to guide decision-making. Finally, participants used the editor module to form hypotheses about the model's reasoning and test them, using the results to investigate how the model worked and whether its reasoning was clinically sensible.

To prevent ML systems from being misused or mistrusted, it is crucial that they provide their users (who may not, themselves, be familiar with ML) with information that is both relevant to the task at hand and understandable to them. In this paper, we find that our proposed interface is a promising direction for providing such users the ability to visualize and probe the model's output, intuitively understand the reliability of its predictions, and build effective trust over time.

2 RELATED WORK

2.1 Interpretability Methods for Human Understanding

The growing field of ML interpretability aims to provide information that helps people understand how a particular model works, either on a global or case-by-case level [18]. Such efforts can serve a number of different goals, such as aiding in decision-making, helping debug or improve a system, or building confidence in the model [21]. A major area of focus has been on developing methodologies for computing such explanations [14]. We note that these methods are mostly intended for settings where the data and domain are complex enough to warrant models that are not inherently interpretable on their own, and these are the settings we focus on in this paper.

Some methods try to visualize the internals of a particular model to reason about how it is operating [12, 34, 58]. This can be useful for theoretical ML understanding and model development, but amy be too abstract and complicated to help people without knowledge of such models and how they work. Others try to produce explanations more grounded in the features of the data, such as a ranking of features important for the prediction or a decision-tree approximating the model's logic [16, 33, 46]. However, a growing body of work that has tried to empirically measure the efficacy of many of these methods has shown that they often do not actually affect or improve human decision-making [2, 8, 30, 43], and in practice are primarily used for internal model debugging [6].

To understand the discord between proposed interpretability methods and their suitability for real-world users, we can draw

from well-established theories in cognitive psychology that describe how people think about problems and organize information using different "cognitive chunks" [35]. For example, a physician might think about decisions in terms of concepts that are higher-level than individual features, or relate features to each other in more complex ways than linearly ranking them by importance. This idea manifests in theories of HCI stating that effective and engaging interfaces should allow users to view and interact with them in a way that feels *direct* — i.e., the visualizations and interactive mechanisms available to users should align with their cognitive chunks. Specifically, Hutchins et al. [23] describe "the gulf of execution," arising from a gap between the available mechanisms of an interface and the user's thoughts and goals, and "the gulf of evaluation," arising from a gap between the visual display of an interface and the user's conceptual model of the domain. In this paper, our aim is to narrow both of these gaps.

To this end, example-based (also referred to as instance-based) interpretability methods, which produce explanations in terms of other input examples, are of particular interest. Research in cognitive psychology and education supports the idea that people often use past cases to reason about new ones when solving problems [1] and that utilizing examples can help people understand complex concepts, build intuition, and form better mental models [44, 45].

Different types of example-based explanations for ML models have been proposed. Many of these are computed post-hoc, i.e., they are generated after a prediction is made to try and explain that prediction. For example, counterfactual examples [19, 56] use gradient-based methods to generate the closest example(s) to the input that are predicted to be a different class (defining appropriate measures of "closeness" is an open question). Influence functions [28] try to trace a model's predictions back to the data it was trained on, identifying the examples that were most influential to the prediction. Normative explanations [9] present users with a set of training examples from the predicted class. There are also limitations of some of these approaches; for example, technical constraints make quickly generating influential examples quite difficult in practice [5, 6], and hidden assumptions about actionability in counterfactual explanations can be misleading [4].

Others compute example-based explanations by modifying the inference process of a trained model to produce predictions based directly on similar training examples. This type of method first appeared in Caruana et al. [13] and Shin and Park [50]. They both propose utilizing a trained neural network model to improve a k-nearest neighbors classifier, either through weighting input features according to the neural network when computing example similarity [50], or through computing similarity in the embedding space of a neural network [13]. The class label making up the majority of nearest neighbors is the prediction, and the nearest neighbor examples can be used as an explanation. Of particular relevance to our case study, Caruana et al. [13] are motivated by the potential benefits of example-based explanations in clinical settings: "*[b]ecause medical training and practice emphasizes case evaluation, most medical practitioners are adept at understanding explanations provided in the form of cases.*" Recently, [41] extended this methodology to compute neighbors using embeddings from multiple layers of a neural network, demonstrating additional benefits for improving the model's robustness and confidence estimates.

In our proposed interface, we compute neighbors using the method reflected in [13], though this could be easily extended to calculate neighbors in a weighted input space as in [50], or to use embeddings from multiple layers of the neural network as in [41]. This prior work has focused on developing optimal ways to utilize the trained neural network to inform the k-nearest neighbors algorithm, implying that the nearest neighbors would then serve as an explanation. Here, we focus on the relatively unexplored part of this claim, investigating how the resultant output should be presented to the user in an interactive interface to narrow the gulfs of execution and evaluation. We explore a specific case study to more clearly define the ways in which this type of example-based explanation can improve trust and understanding for real users.

2.2 Interactivity and Visualization for Interpretability

For explanations to be useful in practice, figuring out how to present the information to a user is a critical step. In a literature review of interpretability systems and techniques, Nunes and Jannach [40] found that the vast majority of papers presented explanations in a natural-language-based format (e.g., a list of feature weights). Other types of visualizations include simple charts (e.g., bar plots indicating feature importances) [46] or highlighting/denoting sections of the input (e.g., displaying important pixels of an image in a different color or opacity) [30, 52]. With respect to example-based explanations, the visualizations used in papers are often a table of features if the data is tabular [37, 56, 57] or a list of images if the data is image-based [9, 27, 28]. Here, we explore visual encodings that convey more information and allow for more interaction than listing examples.

Other work specifically focuses on visualizations of latent embeddings within a neural network model. Many of these utilize 2 or 3D plots to visualize distance between different examples in the embedding space [7, 20, 32]. Liu et al. [32] add an additional visualization of 1D vectors of examples corresponding to user-defined concepts, and Boggust et al. [7] provide the ability to compare embeddings of two different models by viewing and interacting with the two plots side-by-side. Particularly relevant to our work, some of the visualizations of text embeddings proposed in [20] aim to display a given word's nearest neighbors in the embedding space. They plot the nearest neighbors as points along a 1D axis that encodes distance, and provide the ability to compare the nearest neighbors across different embeddings.

We also explore the benefits of interactive input modification within our interface. Prior work in interpretability for ML systems has studied interactivity primarily from the angle of using human feedback to modify or filter the information that is shown [10, 26, 29, 51]. Here, our goal is instead to provide users with a way to test hypotheses about model behavior and explore limitations. The tool described in [57] similarly includes a feature for modifying the input to observe how a model's output changes, though in their case, it is intended primarily for users familiar with ML.

Like these prior works, our interface aims to facilitate understanding by allowing users to visualize and interact with examples from the data. However, while these prior visualization tools are intended for general exploration of what a model has learnt, or for

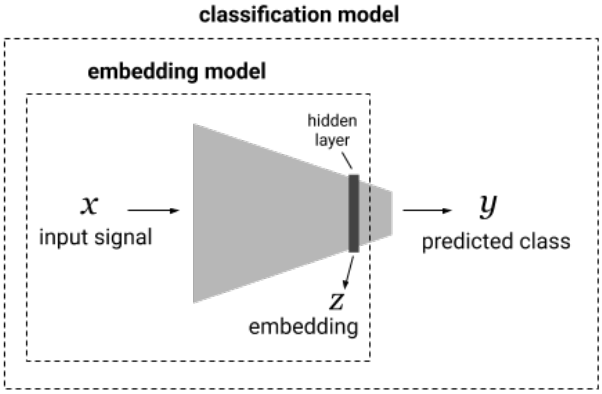


Figure 2: To compute nearest neighbors, we extract an embedding model from the original classification model, where the output is a learned representation (i.e., the activation of one of the model's hidden layers). We can then use this to embed the training data examples and rank them by similarity to the input in this learned embedding space. In other words, we are replacing the final layers of the original classification model with a nearest neighbors classifier.

uncovering underlying structure in data, the goal of our interface is to help users assess the reliability of predictions on a case-by-case basis.

3 TWO INTERFACE MODULES FOR INTUITIVELY ASSESSING ML MODEL RELIABILITY

We propose two interface modules: a display of the model's output in terms of an aggregate and an individual-level view of nearest neighbors, and an editor with which users can interactively modify the input and observe how the model's output changes. These utilize general ideas that can be customized to different domains. In Sec. 3.3, we present a concrete instantiation for ECG beat classification.

3.1 The KNN Module

In the k-nearest neighbors (KNN) module, we display the model's output for a particular example in terms of nearest neighbors from the training set.

The nearest neighbors are computed using a similar method to [13, 41, 50]. Given a neural network model trained to perform the classification task (*the classification model*), we first define an *embedding model*, whose output is the activations of one of the model's hidden layers (see Figure 2). We use this to embed all the training examples. Then, for a given new input example, we first embed it and then use KNN to find the most similar training examples in this learned representation space. These similar examples, along with their class labels, are visualized in the KNN module. The class label associated with the majority of nearest neighbors is the model's prediction.

Computing nearest neighbors in the learned embedding space of the classification model provides the advantage of harnessing the classification model's representational capacity. In other words, since the learned embedding space of the classification model encodes higher level features relevant to the task, these are then taken into account when calculating similar examples. Given our goal of narrowing the *gulf of evaluation* [23] for users — providing ways for them to understand the model's output in terms of higher-level concepts that align with how they think about the task — this step is particularly important. Producing predictions based directly off nearest neighbors then facilitates example-based explanations by visualizing the neighbors.

The KNN module has three main components: an aggregate view of the neighbors' class labels, a unit visualization of individual neighbors that encodes their class and distance from the input, and an overlaid display of the raw input examples associated with each neighbor. These components support the following use cases, which we will describe in more detail in the ECG case study:

(1) Visualizing the distribution of and variation amongst neighbors can help explain when the model is (or is not) reliable and, over time, help users build a broader sense of model limitations.
(2) Comparing neighbors from the other non-majority classes can help characterize why the model is uncertain, and how that uncertainty should guide decision-making.
(3) Seeing nearest neighbors that do not align with prior expectations of the data can help reveal limitations in the training data and prompt further questioning.

3.2 The Editor Module

A range of prior work interviewing interpretability stakeholders has suggested that to build effective trust, users need the ability to confirm that the model is using sensible logic and that its reasoning aligns with their expectations [6, 11, 21, 31, 55]. To this end, the editor module allows users to apply transformations to the input and re-run the modified input through the model to see how the output changes. Users can apply transformations that they expect to be class-preserving, for example, and ensure that the model's output does not drastically change.

The available transformations should help narrow the *gulf of execution* [23] in the interface by providing transformations that align with users' existing ways of thinking about the data and task. For example, in a dataset of natural images, it does not make sense to invert the colors because that is not something that would occur naturally, and does not reflect thought processes of people analyzing images. We also would not want to provide transformations like editing individual pixels, which operate at a much lower-level than a person looking at an image would consider. To come up with transformations that are data-specific (meaning they reflect how users think about a modifying a specific type of data, like images or ECG signals), relevant to the task (meaning they reflect higher-level factors that users consider important to the task at hand), and aligned with the target users' level of understanding, we emphasize the importance with working with domain experts and other intended end users to design them.

Class	Percentage of Examples	Test Set Accuracy
Normal	89.3%	99.6%
Supraventricular Ectopic	2.7%	70.5%
Ventricular Ectopic	7.1%	95.7%
Fusion	0.8%	70.4%
Overall	–	98.3%

Table 1: The classes used in the ECG beat classification task, along with their representation in the dataset and the model's test set performance on that class.

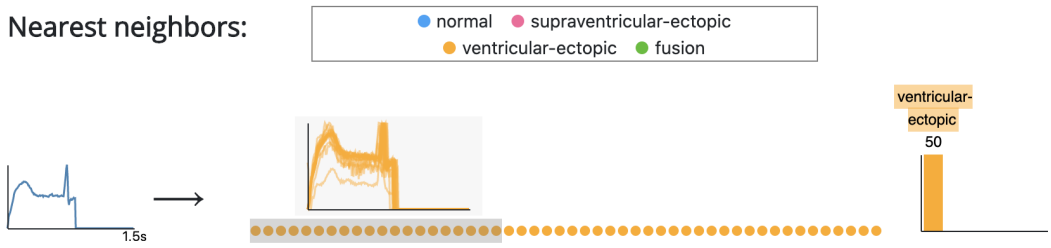


Figure 3: An example of the KNN module for a particular ECG beat. On the left is the input signal. On the right is a histogram of class labels for the 50 nearest neighbors from the training data (here, all 50 are in the class ventricular ectopic). The label and count appear above the bar on hover. In the center, each dot represents an individual nearest neighbor, ordered by similarity to the input. The plot above overlays the signals in the selected region on each other. The brush selection can be moved or redrawn elsewhere.

We envision the following use cases for the editor module, which we expand upon in the following sections with examples from the ECG case study:

(1) Checking alignment with domain knowledge through testing hypotheses about how the model's output should change upon various transformations.
(2) Assessing prediction reliability by seeing if the output is overly sensitive to small transformations that should be class-preserving.
(3) Understanding how the model works by observing which transformations lead to a significant shift in the class distribution.

3.3 ECG Beat Classification Case Study

Thus far, we have described the motivation for the interface modules generally, but here we will instantiate and evaluate each with a specific clinical case study of classifying electrocardiogram (ECG) beats. We chose this application because we wanted to be able to perform an application-grounded evaluation of our system using a realistic task that people (i.e., physicians) were familiar with [15]. Using simplified or proxy tasks (e.g., asking people to classify images) can be more straightforward, but also can lead to less reliable results since the task at hand is not something the participants are familiar with or have prior conceptions about doing. ECG beat classification, in particular, is an area where machine learning has been widely applied and yielded good performance [25, 47, 59]. It is also generally applicable in the medical domain since most physicians are familiar with reading ECG beats.

3.3.1 Task, Dataset, and Model. The specific task we implemented was classifying a single ECG heartbeat into one of four categories: normal, supraventricular ectopic, ventricular ectopic, or fusion. The latter three classes are different types of arrhythmias, or heart rhythm problems.

We used a preprocessed version of the MIT-BIH Arrhythmia Dataset [36] available on Kaggle [17]. Each sample in the dataset is an individual heartbeat sampled at a frequency of 125 Hz, and padded to a maximum length of 1.5 seconds. The available dataset contains a fifth class, "unknown," which we exclude here.

We replicated the convolutional neural network (CNN) classification model from [25], without data augmentation (we were interested to see whether our visualizations could elucidate, for example, that certain classes were underrepresented). The model was trained for ten epochs to a final overall accuracy of 98.3%. The breakdown of classes and performances on each is in Table 1.

3.3.2 The KNN Module for ECG Beat Classification. For the ECG beat classification task, we use the same CNN classification model described above, and we define the embedding model to output the activations from the final hidden layer (a 32-dimensional vector). As in prior work, we use Euclidean distance in this space to rank the embeddings of the training examples by their similarity to a particular input. We retrieve the 50 nearest neighbors for visualization.

Figure 3 shows an example ECG beat in the interface. Throughout the interface, color encodes class labels (e.g., orange waveforms, dots, and bars correspond to ventricular ectopic examples). The aggregate view is a histogram of class labels present in the nearest

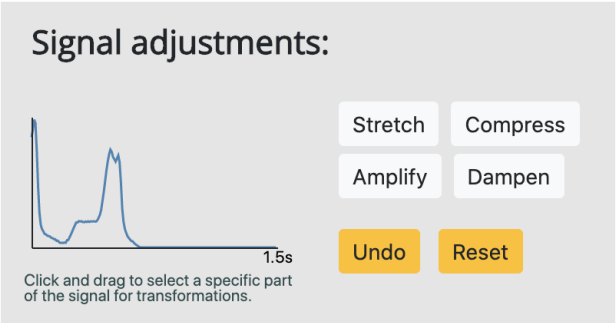


Figure 4: The editing toolbar allows users to apply specific transformations or combinations of transformations to the input signal. The transformations can be applied to the entire signal, or to a specific user-selected region. This allows users to select and transform clinically-meaningful segments of the signal (e.g., "stretch the QRS complex").

neighbors, ordered by class frequency to identify the majority class and distribution of other classes. The exact count of each class appears on hover for each bar in the histogram.

The unit visualization of individual neighbors is a series of dots arrayed horizontally and ordered by similarity to the input. Users can see, for example, within the nearest neighbors if certain classes are more similar to the input. When prototyping this component, we also considered designs that encoded the absolute similarity (e.g., placing two neighbors that were relatively similar nearer to each other). However, we decided against this, since the absolute similarity (i.e., Euclidean distance in the learned embedding space) is not a value that is meaningful or familiar to the user. Additionally, the distribution of these values is more complicated to visualize, since the distances between neighbors are inconsistent. In our prototypes, for example, there were often clusters of points that densely overlapped and did not facilitate selecting and viewing individual examples.

To visualize the raw input examples, users can brush over specific segments of the ordered dots. The brush is initialized to the first five neighbors since these represent the most similar examples. Because the ECG data is signal-based, we choose to visualize the neighbors by overlapping signals on a single plot that appears above the brush. This allows users to visually assess consistency amongst the neighbors – for example, if the neighbors are very consistent, the overlaid plot will look very similar to a single signal, while if they are very varied, the overlaid plot will appear comparably noisy. Outliers are also visible, since they appear as a distinct waveform that does not follow in the same pattern as the other signals. In the example beat in Figure 3, for example, we can see that most of the waveforms follow the same general pattern, with some variance in their height, and that there is one that is significantly shorter than the others. By moving and adjusting the brush to cover specific segments of the neighbors, users can home in on examples from specific classes or individual outliers.

3.3.3 The Editor Module for ECG Beat Classification. For the ECG beat classification task, the editor consists of four transformations which we arrived at through discussion with a cardiologist: amplify, dampen, stretch, and compress. These transformations can be applied to the entire input signal, or to specific user-defined regions using the brushing functionality. Together, they allow for a large space of possible adjustments to the input signal. There are certainly other options that could be explored here, such as detecting certain important sections of the signal (e.g., "P wave" or "QRS complex") to transform instead of having users select them themselves. Although some combinations of transformations might lead to a signal that is not realistic, our tool is intended for users who are familiar with what beats should look like, so we assume that they will realize when this happens and undo or reset.

Once the transformation has been applied, a new row appears below the original output, displaying the new output. The color encoding as well as highlighting on hover enables tracing how the class distribution changes overall, while links between neighbors that are shared across rows enables tracking how individual examples shift in similarity. The editing toolbar is pictured in Figure 4, and an example of the output after several transformations is in Figure 5.

3.4 Design Goals and Use Cases

Throughout the interface, our design goals are to narrow the *gulf of evaluation* and the *gulf of execution* for users by providing, respectively, visualizations and input modifications that reflect meaningful, higher-level concepts that align with their existing ways of thinking about the task. In doing so, we are able to provide more a more intuitive interactive interface with which to assess the model's reliability, understand why it is uncertain, and check whether its reasoning aligns with domain knowledge. Here, we expand upon several specific ways that a user can interact with both modules to these ends:

3.4.1 Assessing consistency among nearest neighbors to understand prediction reliability and data limitations. Users can assess the reliability of the prediction in multiple ways. First, the aggregate distribution of class labels can convey the model's uncertainty in the prediction (i.e., the majority class label). For example, if 45 neighbors are normal, this conveys more certainty about the prediction than if only 25 neighbors are normal, and the rest are spread out across other classes.

Second, by viewing the class labels of the unit visualization representing individual neighbors, users can see how similar the neighbors from non-majority classes are to neighbors from the majority class. For example, if there are 40 neighbors labeled normal and 10 neighbors labeled fusion, are those 10 the most similar to the input? Or do they appear closer to the latter end of the nearest neighbors? If the neighbors from the non-majority class are the 10 most similar, this might indicate further unreliability of the 'normal' prediction.

Third, visualizing the variance or consistency amongst the waveforms themselves can give insight into whether the input example is well-represented in the training data and whether the model is picking up on sensible high-level features common in the neighbors. For example, if the overlaid plot of nearest neighbors shows examples that are very consistent and similar to the input in semantically meaningful ways (see Figure 6a for an example), it implies

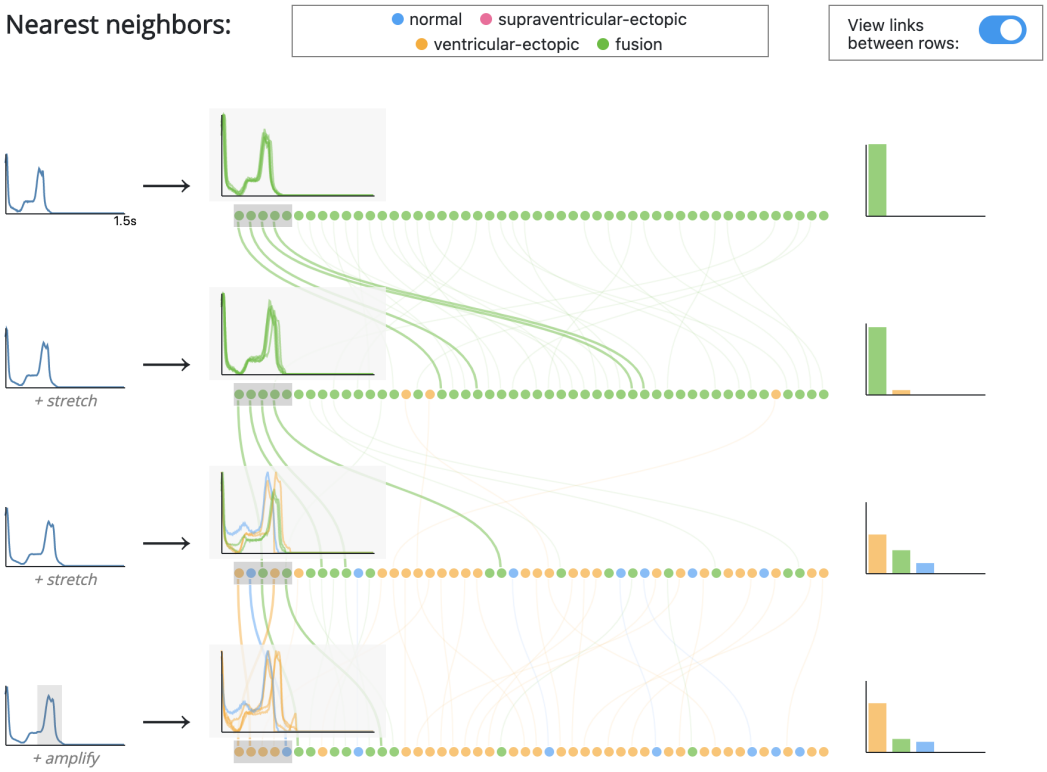
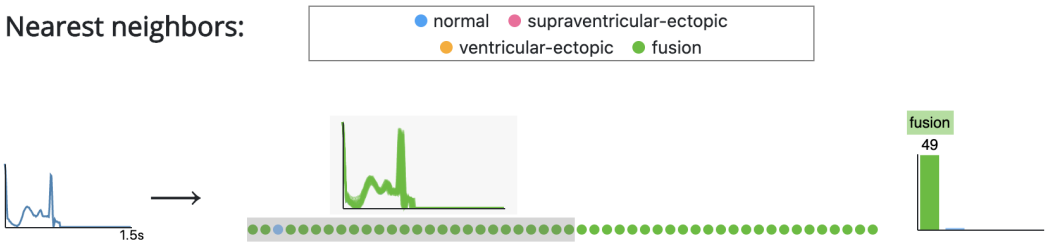


Figure 5: As transformations are applied, new rows appear below the original output with the transformed input and the model's new output. Links between each row indicate neighbors that are shared. The links corresponding to neighbors within a row's selection are more visible, while the rest are more transparent. Users can get a general sense of how much the nearest neighbors change (by assessing the overall density of links) as well as the specific movements of particular neighbors or sets of neighbors.

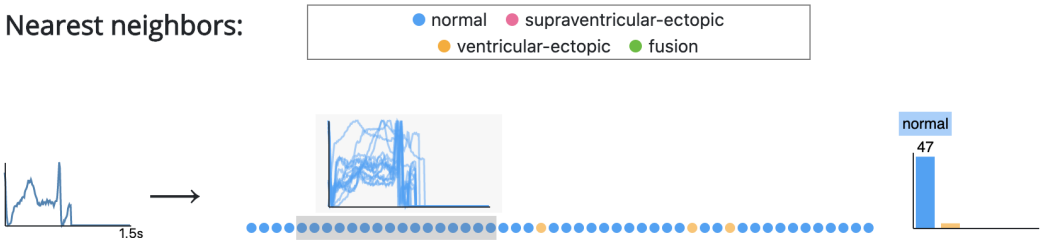
that the input is well-represented in the training data and that the model is picking up on the right concepts for this input. The user can look at the commonalities amongst neighbors to further understand what the model is picking up on and whether it aligns with their expectations of the domain. On the other hand, if the plot of nearest neighbor signals shows examples that are non-overlapping or not similar to the input (see Figure 6b for an example), it implies that either examples like the input are not well-represented in the training data, or that the model is not learning the right features and therefore not finding those similar examples.

3.4.2 Investigating neighbors from non-majority classes to characterize prediction uncertainty. Typically, a classification model outputs a probability score indicating its certainty in its prediction. Probability scores can alert the user to some uncertainty in the model, but they don't give the user any additional information to understand why the model is uncertain.

In the KNN module, one way the model's certainty is conveyed is through the aggregate distribution of class labels. Beyond this, though, the user can further investigate why the model is uncertain by viewing and comparing examples from non-majority classes. Brushing over specific selections of dots representing individual neighbors allows the user to compare neighbors from different classes on the same plot, or analyze them separately. Take the example in Figure 7. 30 of the neighbors have the class label supraventricular ectopic, and 20 have the label normal (the counts of each class are visible upon hovering over the bars in the aggregate histogram). Looking at the ordered dots representing individual neighbors, we can see that most of the closer neighbors are supraventricular ectopics, while the later neighbors are mostly normal. This might give us more confidence in the model's prediction of supraventricular ectopic. Then, in Figure 7a, we can see that brushing over the first 15 neighbors reveals that most of them follow the same general pattern, and it looks very similar to the input. The 3 normal neighbors in this selection also seem to follow this pattern — so some of the model's uncertainty is arising from the fact that in the training data, there are normal beats that can look very similar to supraventricular beats. In Figure 7b, we can see that brushing over the last 15 neighbors reveals that most of them follow the same general pattern, but have a more elevated T-wave (the spike at the beginning of the signal) than the supraventricular ectopic neighbors. We might reason, then, that the model is split between supraventricular and normal, and one of the factors driving the uncertainty is whether or not the input has a significant T-wave.



(a)



(b)

Figure 6: For the ECG data, we visualize individual nearest neighbors by overlaying those in the user's selection on top of one another. This allows for assessing the variance or consistency among the neighbors, comparing neighbors from different classes, getting a sense of the "average" waveform, and detecting if an output might be out-of-distribution with respect to the training data. (a) shows an example where the neighbors are very consistent, and (b) shows an example where they are much noisier.

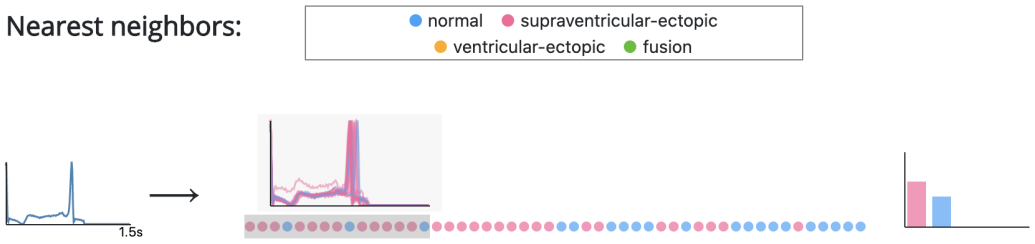
The user can then use their domain knowledge to reason about how to proceed. In this example, for instance, the user might examine the input and decide that the T-wave is significantly depressed, making the input more similar to the supraventricular ectopic examples, and more confidently proceed with supraventricular ectopic as the correct class. Or, they might decide that the different classes present in the neighbors reflect legitimate ambiguities about what the correct beat type is, and choose to deliberate further, consult a second option, or run additional tests.

3.4.3 Comparing nearest neighbor examples and labels against domain expectations to prompt critical questioning around the data. If, when looking at neighboring examples, the way the data looks or is labeled does not align with the user's expectations, it might prompt questions from the user about the details of the data and how it was collected or labeled, areas that are too often not engaged with after a model's deployment. Crucially, seeing the signals themselves facilitates this type of critical thinking for people who are likely more familiar with the data and what it should look like than more abstract representations like feature weights.

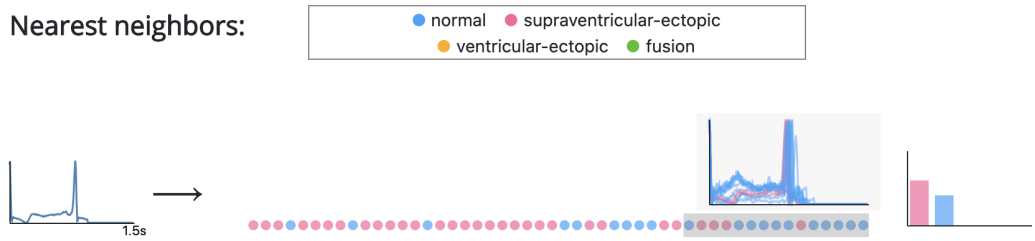
In the ECG case study, for example, the data was annotated by physicians who had access to additional information about the beats preceding and following the input. As a result, there are some examples in the dataset that look extremely similar but are labelled differently (presumably because the difference in their label was due to the information available during annotation that the model does not see). In some cases, this leads to nearest neighbors that have different classes but look very similar (see Figure 8). Viewing the neighbors for a particular example can prompt questions about how the data was annotated and the subsequent limitations of the model, which would likely not arise if users were not able to view and compare specific similar examples.

3.4.4 Applying input transformations to check if model reasoning aligns with domain knowledge. Checking if the model's reasoning aligns with prior expectations of domain experts is crucial for building trust, especially in the clinical domain [11, 55]. The editor module allows users to form hypotheses about how particular transformations should change the model's output, and build confidence and intuition around the model's reasoning by seeing if these hypotheses hold. For example, the beat in Figure 9 is initially classified as supraventricular ectopic. The user might hypothesize that since one indicator of supraventricular ectopic beats is narrowness, and this particular beat is narrow, that this is what the model is picking up on. Therefore, stretching the beat should change the model's output, making it lean more towards normal. The user can then use the editor to apply this transformation to test their hypothesis. In this case, the model's output does change to reflect more normal neighbors, confirming both the original hypothesis and the fact that its reasoning in the case aligns with the user's expectations from a clinical perspective.

3.4.5 Applying transformations to assess the model's sensitivity to small perturbations. Even aside from specific hypotheses about how a particular series of transformations should change the output,



(a)



(b)

Figure 7: In this example, 30 of the nearest neighbors are supraventricular ectopic, and 20 are normal. Rather than just concluding that the prediction is "60% confident," the user can home in on the non-majority examples to characterize where the uncertainty is coming from and how to take it into account. (a) shows brushing over and viewing the first 15 neighbors. Most of them follow the same general pattern that looks very similar to the input. The 3 normal neighbors in this selection also seem to follow this pattern – meaning that some of the model's uncertainty is arising from the fact that normal beats can look very similar to supraventricular beats. (b) shows viewing the last 15 neighbors. Most of them are normal, following a similar pattern, but with a more elevated T-wave (the spike at the beginning of the signal) than the supraventricular ectopic neighbors. This implies that one of the factors driving the uncertainty is whether or not the input has a significant T-wave. Based on this understanding of the uncertainty, the user can use their domain knowledge to decide how to proceed.

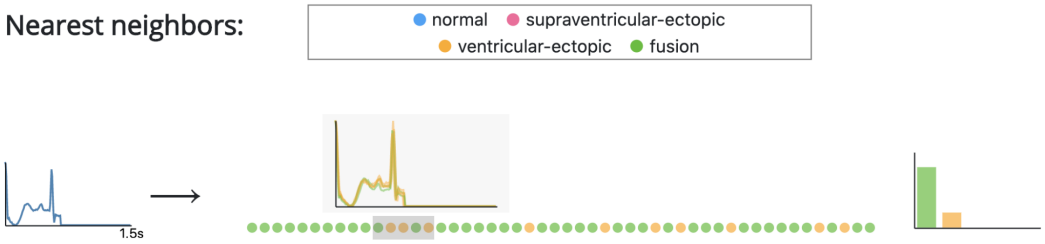


Figure 8: An example of neighbors that look similar but have different labels, due to a discrepancy in the additional information available during annotation versus at test-time. Alerting users to such cases through viewing nearest neighbors can help prompt questions about the data, the annotation process, and limitations of the model.

a user can still gauge the reliability of a particular prediction by performing ad-hoc sensitivity analyses. If the output changes drastically when the input is slightly tweaked, this can alert users to the fact that the prediction is precarious and encourage them not to be overly reliant on it. On the other hand, if the output is relatively stable, this can be an additional indicator of model reliability.

4 USER EVALUATIONS

We ran a set of think-aloud studies to evaluate how well our design goals came across using the ECG beat classification case study[1]. In particular, we sought to study the two following questions:

[1]These studies were certified by our institution as exempt from IRB review under Category 3 (benign behavioral intervention).

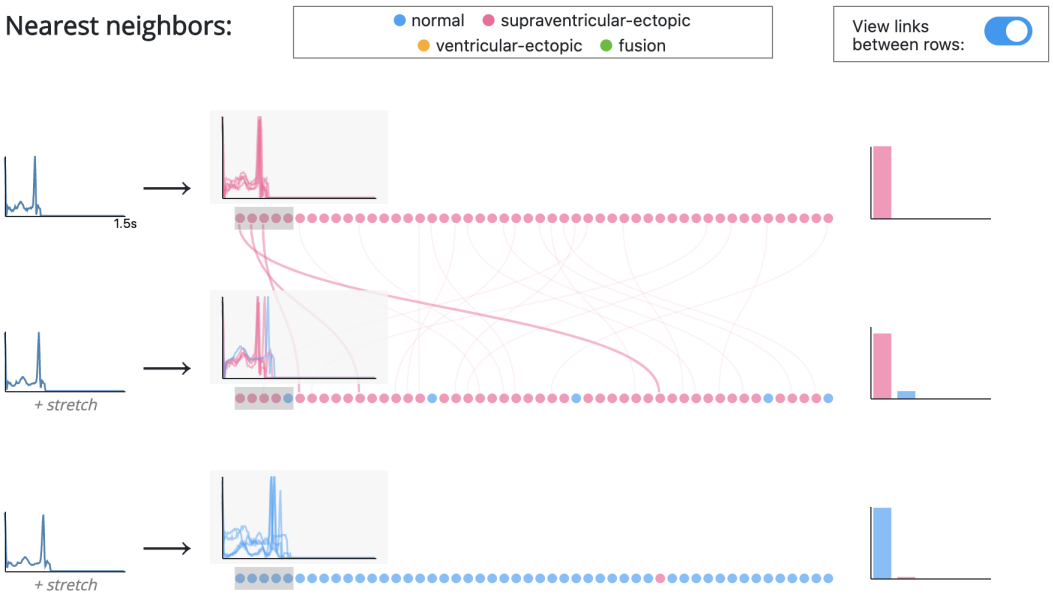


Figure 9: Here, the input beat is initially classified as supraventricular ectopic. One indicator of a supraventricular ectopic beat is narrowness, so we would expect it to become more normal upon stretching. A user could use the editor to stretch the beat out, then, to test out such a hypothesis as a way to check whether the model's reasoning aligns with clinical expectations.

(1) Can visualizing the nearest neighbors waveforms and their classes help physicians understand a prediction's reliability in terms of clinically-meaningful concepts that they are familiar with?
(2) Can editing inputs further help physicians test out hypotheses about the model's reasoning and confirm if it aligns with domain expectations?

4.1 Participants

Our participants all had a medical background but varied in experience, and were recruited through our personal networks. There were 9 total participants: 3 fourth year medical students (P1-P3), 5 first year residents (P4-P8), and 1 internal medicine physician with five years of experience (P9).

4.2 Study Design

4.2.1 Baseline. The baseline interface we used displayed output from the machine learning model that included the predicted class, the predicted probability of that class, and a feature importance-based explanation. Feature importance-based explanations are widely proposed and referenced [6, 16], and constitute a natural alternative to example-based explanations.

The particular feature-importance based explanation we implemented was LIME [46], which we chose because it is well-known, commonly used, and open source. To explain a specific input example, LIME finds a local neighborhood of points surrounding it, gets the predicted class for each, and computes a linear boundary to classify that set of points. The parameters of that boundary can be thought of as an approximation to the model's behavior in that specific local neighborhood. Because the function is linear, the coefficients associated with each input feature can then be extracted and ordered by importance.

In our implementation (pictured in Figure 10), each input example consists of one ECG beat, so each coefficient corresponds to one measurement in that signal. To visualize these feature importances, we added a second plot of the input signal with a highlighting layer corresponding to important sections, which is in line with existing visualizations of proposed feature importance-based explanations for ECGs [39, 54]. Specifically, we plot the feature importance values that are both above the 80th percentile and part of a continuous segment of neighboring important features, to align more with physicians' existing ways of thinking about distinct important sections of an ECG signal.

4.2.2 Study Procedure. In order to study the effect of our interface modules independently, each participant saw three interfaces. The first two interfaces were randomly ordered between the KNN module (without the editor) and the feature-importance baseline to allow us to study whether visualizing nearest neighbors helped build participant intuition about the model. And then, to understand the impact of interactive editing of inputs, the third interface featured the KNN module *with* the editor.

Each interface was pre-populated with 12 input beats chosen from the test set and equally distributed among the four classes. We chose 30% of the beats such that they had incorrect predictions (for the baseline, the prediction is the class with highest probability; for the KNN Module, the prediction is the class that makes up the majority of nearest neighbors). The incorrect predictions were distributed to try and align with the model's actual performance

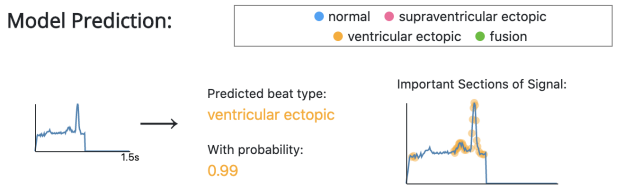


Figure 10: The feature importance-based baseline interface consists of the predicted beat class, the probability with which that class was predicted, and segments of the beat considered most important to the decision highlighted. The important segments were calculated via LIME, and we visualize continuous segments with importance values above the 80th percentile in the plot.

(e.g., we did not include incorrect predictions for normal beats since there are very few of those in reality; we included more incorrect predictions for supraventricular ectopic since the model's performance for that class is worse).

All studies took place over Zoom. Participants were informed that their participation was voluntary, that they could decline to continue at any point, and that their identities would remain anonymous in any research output. The studies were recorded (video + audio) with participant consent. Each took an average of 50 minutes, and participants were compensated with a $30 gift card.

At the start, participants were told which four categories of beats they would be working with including the more granular information about beat types included with the original dataset (e.g., there are multiple pathologies that fall under the umbrella of "ventricular ectopic"). We described that they would see ECG beats one-by-one, along with output from a machine learning model that had high overall performance. Participants were asked to imagine a scenario where their workplace had adopted such a tool for beat classification, and they were both trying to consider the model's output to make the best decision about a particular beat, as well as get a general sense for how the model worked.

Each interface condition was introduced as being backed by a different model to mitigate participants carrying over preconceptions from prior conditions. For each interface, participants were given a brief demo and were then sent a link to open the interface on their computer and asked to share their screen. They were prompted to click through the beats, and for each one, think aloud about how they were coming to a decision about the beat's class, how they were incorporating the model's output, and whether their perceptions about the model changed. At the end of each interface, we asked a few debriefing questions about their general thoughts on the model, the interface, and its strengths and weaknesses.

4.3 Observations

Our studies provide evidence that our interface modules successfully met our design goals. Visualizations of the neighboring signals allowed participants to reason about the model's output in terms of clinically-meaningful concepts. Moreover, participants were able to intuitively assess prediction reliability by examining variance in the overlaid visualizations of nearest neighbor signals. Inspecting the aggregate chart, ordering of neighbors, and neighboring signals all helped participants understand the model's uncertainty and relate it to relevant ambiguities in the task. When the nearest neighbors did not align with participants' prior expectations of the domain, they formulated questions about the data and the labeling process. And, finally, participants used the editor to confirm if the model's reasoning was sensible and guide decision-making. Here, we describe these observations in greater detail.

4.3.1 Nearest neighbors enable reasoning with clinically-relevant concepts. Visualizing nearest neighbors allowed participants to reason about the model in terms of clinically-relevant concepts. For example, participants would often notice a particular morphology present in the neighbors that helped them understand the model's reasoning and whether it was clinically sensible. One participant said, pointing to a pattern present in all the neighboring signals, *"Yeah, ventricular. It's this elevation and this space that's making it think ventricular"* [P4]. Another described, *"The model is right — with ventricular ectopic, the QRS spike should be broad, which is present in all the similar examples"* [P9]. Overall, six participants reasoned about the model using high-level clinically-relevant concepts that they observed in the neighbors, like *"depression in the signal"* [P9], *"slope right after the P-wave"* [P7], *"presence of a T-wave"* [P8], *"P-R interval"* [P5], or *"abnormal morphology"* [P3].

However, in a few cases, participants were not able to reason about the neighbors: this occurred either when they were unsure why the nearest neighbors were similar, or disagreed with their class labels. For example, one participant said, *"these [nearest neighbors] are supposed to be are ventricular ectopic... I think they're normal. I don't know what to make of this [output]"* [P2]. In part, this is likely due to the fact that when labeling a particular beat, the annotators of the data had additional information about surrounding beats that is not available in the dataset or used by the model. Without that additional information, it is sometimes unclear why a particular beat has the class label that it does. While the model's output was not very helpful in these cases, it did sometimes prompt additional questions about the data and labeling process. For example, one participant asked, *"Some of these normal ones look like they could be abnormal, so I'd want to know why they were called normal and what that was based on"* [P6]. Another participant asked a similar question and further hypothesized, *"Most likely this data was correctly annotated considering the multi-lead strip of ECG, but it's not using all that information here"* [P2]. An additional factor in some of these cases is that certain classes of beats are under-represented in the training data. Therefore, it is less likely that the neighboring beats for an input in this class will be very similar, since there are simply less of them to choose from. We were curious whether participants would notice this underrepresentation, but we found that it did not naturally occur to them during the study, and occasionally they expressed confusion as to why the neighbors were not as similar in these cases.

For the feature importance-based baseline, participants often had difficulty pulling out higher-level, clinically-relevant concepts from the explanations. For example, echoing a sentiment shared by many, one participant said, *"I don't see how these blue [highlighted] areas are super helpful here... what are they trying to get at?"* [P7]. Another

participant, who struggled trying to connect the explanation to the predicted class, said *"I don't understand how they go from this [pointing at highlighted areas] to saying that there's some aspect of a ventricular beat in there"* [P2]. Some others had difficulty figuring out what about the highlighted signal was important — for example, one participant asked, *"Why is it highlighted here, is it looking at the height of this, is it looking at width? And why only this part?"* [P1]. In some cases, the highlighted areas *did* align with participants' expectations, but they had difficulty connecting these sections back to the prediction. One participant noted, for example, *"Sometimes it was highlighting things I would also consider, but I still thought its prediction was wrong. I don't have any intuition on that. I guess it's finding some features. I would want to know what those features are, see whether they're useful, if they have any intuitive correlation"* [P2].

4.3.2 Visualizing variation helps intuitively assess prediction reliability. All participants said that they did not place as much weight on the model's prediction when the overlaid signals appeared very noisy and dissimilar from the input. Participants felt more confident in their answers when the overlaid signals were very consistent and similar. They were also able to reason about what kind of variation was acceptable given the task and domain (*"This input isn't as picture perfect so it makes sense that the model shows some variation in the overlaid examples"* [P4]) versus what was an indicator of unreliability (*"[The model's output] isn't giving me much information right now. If I was given this result I wouldn't just listen to the machine, I would want additional information"* [P4]).

In the baseline interface, when the predicted probability was very high, and the prediction aligned with their own, most participants felt reassured. When this was not the case, however, we found that it was difficult for participants to get a sense of how reliable the prediction was. As a result, they often rationalized incorrect predictions — even when it went against their initial instincts. For example, one participant saw an abnormal beat, started to say it was abnormal, but then changed her mind after looking at the predicted class, which (incorrectly) was normal: *"I don't think this is normal... well actually seeing that the machine thinks normal... I guess it has a small QRS and the T-wave has a normal slope. Okay, I'll put this in the normal category"* [P7]. Four participants [P2, P3, P4, P7] went through similar processes of rationalizing an incorrect prediction (after having also expressed an inclination towards the correct class) presented in the baseline interface.

Even when they did not rationalize an incorrect prediction, participants often struggled with relating a probability score, or the important sections of a signal, to an intuitive notion of reliability when using the baseline interface. For instance, one participant thought out loud, *"I don't know, it seems high probability for a weird looking one like this. And I don't know if it makes sense what it's looking at here and calling important. I'm not confident about this"* [P1]. Six participants expressed similar difficulties in reasoning about the reliability of the prediction.

4.3.3 Nearest neighbors help characterize uncertainty and incorporate it into decision-making. When the class distribution of neighbors was split over multiple classes, participants were consistently able to home in on the differences using the overlaid plot of waveforms and align them with clinical concepts. For example, one participant viewed a beat where the neighbors were split between supraventricular ectopic and normal, noting *"For supraventricular ectopic one thing you look for is whether or not it has a P-wave. It's unclear in the input. These [brushing over the supraventricular ectopic examples] are probably saying it isn't a P-wave. And these [brushing over the normal examples] have the P-wave so they're probably saying that the input does also and that's why it should be normal"* [P5].

Participants were often able to relate the model's uncertainty to natural ambiguities in the task that humans also struggle with. For example, when one participant noticed some ventricular ectopic beats present in the nearest neighbors of a fusion beat, *"Given that fusion is itself a combination of ventricular ectopic and normal, it makes sense that there's uncertainty here, and that there are some yellow [ventricular ectopic] ones that look similar"* [P8]. Rather than leading to distrust in the model, the ability to contextualize its uncertainty helped rationalize and move forward with the model's output. Another participant said, regarding neighbors split across classes, *"I would be exactly split like the model is between supraventricular and ventricular ectopic. The fact that the model is also split between those two makes me feel better and I would do further testing to differentiate which one it is"* [P4].

Beyond reasoning about the presence of multiple classes in the nearest neighbors, participants were also able to harness that information along with their domain knowledge during decision-making. In many cases, upon viewing neighbors from the different classes, participants would realize that one of the classes was not actually similar to the input, and as a result, feel more confident in disregarding it. For example, one participant said of the beat in Figure 11, *"This is supraventricular ectopic. [The model] is calling it normal, but the normal ones don't look so similar. The pink ones [supraventricular ectopic] look more like it because they also don't contain a P-wave"* [P9]. In other words, they were able to relate variation in the neighbors to clinical concepts (normal neighbors with a P-wave, supraventricular ectopic neighbors without), hypothesize why the model is uncertain (it isn't sure whether the input example contains a P-wave), and use their own domain knowledge to determine how to proceed (the input does not actually have a P-wave, so go with supraventricular ectopic). Six participants also went through similar thought processes to characterize the model's uncertainty and then use their domain expertise to more confidently arrive at the correct answer.

In the baseline interface, when the model appeared less certain (i.e., a lower probability score), participants had difficulty reasoning about the uncertainty with the given explanation. Many said that they didn't know why the probability was relatively low. When prompted to reason about it, all participants tried to guess using their own knowledge as opposed to information from the feature importance-based explanation.

4.3.4 Editing inputs helps check model reasoning. Many participants used the editor to form hypotheses about what would happen to the output after applying certain transformations, and then test whether it actually happened. They used this as a way to "sanity check" the model's reasoning, and were more confident if it aligned with their expectations (and vice versa). For example, one participant described using the editor to feel more confident in the model's output for a particular beat (shown in Figure 12), which consisted

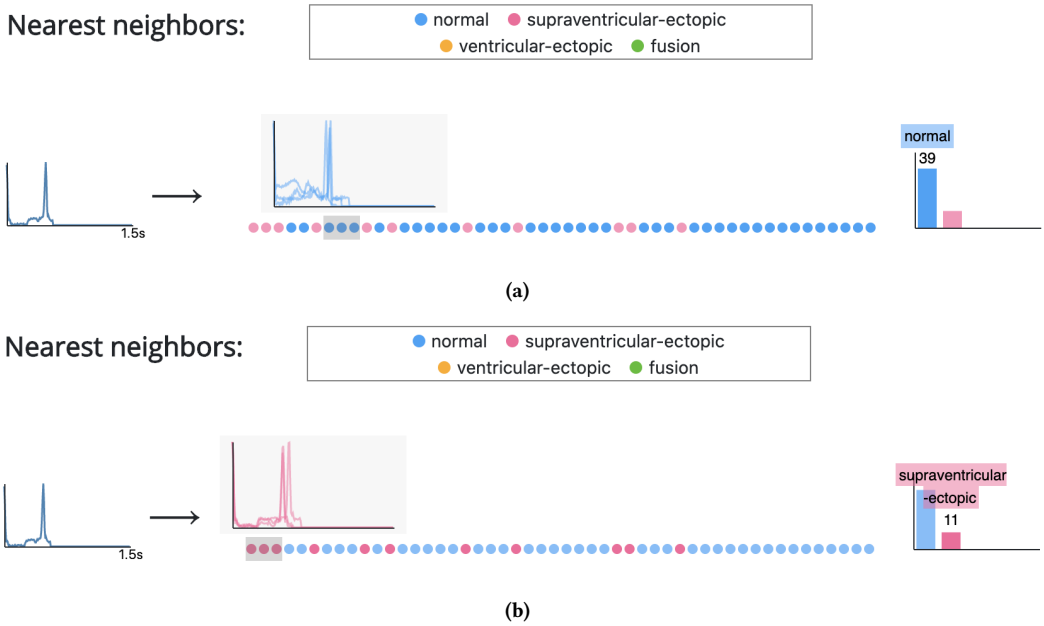


Figure 11: For this beat, one participant looked through some of the normal neighbors (a), comparing them to some of the supraventricular ectopic neighbors (b). They reasoned that the normal examples, though they made up the majority of neighbors, were not more similar in clinically-meaningful ways to the input than the supraventricular ectopic examples. As a result, they were able to arrive at the correct classification (supraventricular ectopic).

of mostly ventricular ectopic neighbors: *"I'm not that confident with ventricular ectopic, and this looks almost normal. It's a little narrow, which is what ventricular means, so I think that's why this is saying ventricular and if I were to stretch it it would be normal. [Stretches the signal] And that's exactly what happened. That makes me more confident that this is more ventricular ectopic rather than normal. Just because that's exactly what my thought was and that's exactly what happened when I did it"* [P4]. The same participant mentioned later on, *"This is how I think of things. If I can predict what's going to happen I'm more likely to be confident in the decision."*

Sometimes, however, participants applied a transformation, but were not able to reason about why the nearest neighbors changed as they did, and subsequently were not sure how to incorporate the observed change into their reasoning. This often happened for the under-represented beats, which, when transformed, would more often than not result in nearest neighbors that were skewed towards normal (i.e., the most highly-represented beat). In part, this is just because the model has learned about a much larger diversity of different normal beats, as opposed to, e.g., different supraventricular ecotpic beats. On one hand, this unexpected behavior prompted participants to rely less on the model's output in these cases — which, since the model is less accurate for these classes, is appropriate. At the same time, these instances were not able to offer them useful feedback on the model's reasoning.

Other participants did not form hypotheses about specific transformations, but applied several to try and gauge the sensitivity of the prediction to small changes as a way of assessing its reliability. Sometimes it served as positive reinforcement: *"Okay, this makes me more confident. When it's normal, and then you do all these [transformations], I think it should mostly stay normal, which it is. It's consistent so this all makes sense and I feel good with the machine"* [P1]. Other times it helped alert participants to the model's unreliability: *"Seeing it switch so quickly from supraventricular ectopic to normal does affect my perception of whether it [the model] is good at telling those apart"* [P3].

With respect to the model's reasoning more generally, some participants expressed an increased understanding in how the model worked after using the editor and observing what transformations tended to lead to a large change in the output. One participant noted, *"Doing these transformations is making me think about how this program works… I can tell that the narrowness of a beat affects the decision a lot for example"* [P8].

Participants did not typically use the editor when the neighbors were very consistent (both in terms of the shape of the signal and their class labels), because they did not feel the need to check the model's reasoning. Other times, they chose not to use the editor because they could not think of a specific hypothesis they wanted to test — this was particularly true for the participants who were medical students, who often expressed that they "didn't know enough" but that someone with more experience might know what to test.

4.4 Study Limitations

Several participants noted that the way the ECG beats were visualized was not realistic. For example, in practice, participants described that they would typically view a strip of beats from multiple leads, rather than one beat in isolation, and often with a grid

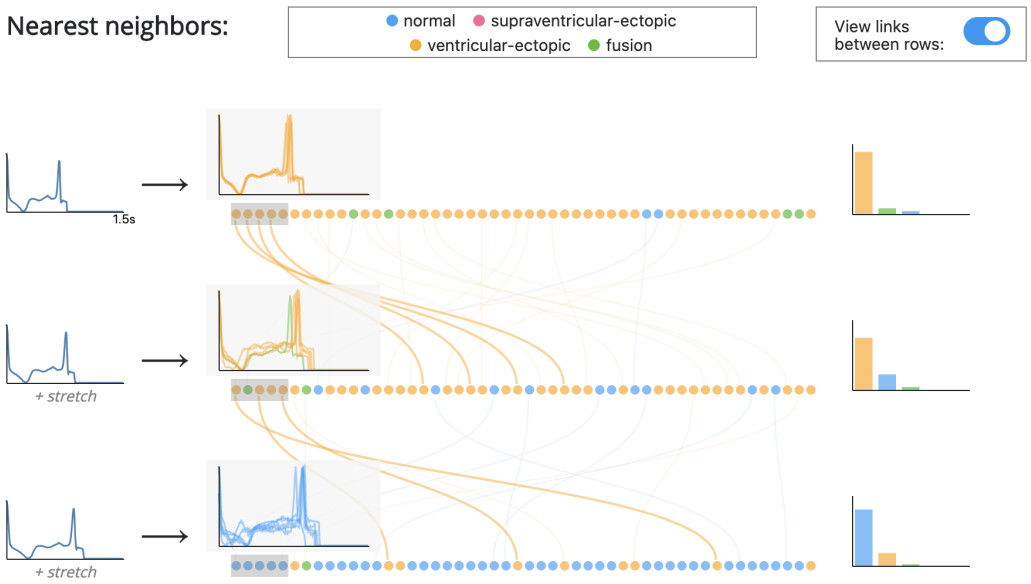


Figure 12: One participant viewing this example hypothesized that this beat was ventricular ectopic because it was narrow, and therefore, that stretching it would cause it to be more normal. After applying the stretching transformation and seeing that the nearest neighbors did change to be more normal, she felt more confident in the model's reasoning for this beat and in making the classification of ventricular ectopic.

overlaid to better measure distances. In some cases, this difference in display made participants more unsure about the class than they would have been if they had had their more familiar overlays. While the interface and task is simpler than it would be in a real clinical environment, in the current work we are more concerned with developing and evaluating the proposed interpretability and visualization techniques than developing a tool that could be deployed in a clinical setting (which would prompt an entirely different set of considerations).

5 DISCUSSION

Throughout the user studies, we found that that our interface modules successfully achieved our design goals as participants' ability to reason about and interact with the model's output aligned with their existing ways of thinking. In particular, viewing nearest neighbor examples elucidated higher-level morphologies that the model was learning, more so than seeing important segments highlighted on the input as in our baseline. Indeed, this result is in line with prior work in cognitive psychology suggesting that people solve problems by utilizing knowledge about past cases [1, 45], and that it is a particularly relevant form of reasoning in the medical domain [48].

We also found that participants generally approached the model in a more critical way, investigating the nearest neighbor class distribution and waveforms to see why the model was uncertain, and connecting that uncertainty to clinical concepts. Importantly, our results support the idea that understanding that the model is uncertain or unreliable for particular cases does not damage trust in the model, but is actually necessary for building it. As Cai et al. [11] find from interviews with physicians, *"participants implicitly and explicitly understood that no tool (or person) is perfect,"* and Tonekaboni et al. [55] further describe that *"the acknowledgement of this challenge promotes trustworthiness."* In other words, when users can explore and understand the model's uncertainty, it further helps align the system with domain expectations that some cases are difficult and/or ambiguous, and that no system or tool is perfect. We found that the feature importance baseline did not facilitate the same investigation — going from important sections of the prediction to *why* the predicted probability is high required too big of a mental leap. This observation aligns with prior work studying the effect of feature-based explanations on doctors' diagnostic performance [8], in which participants expressed similar confusion about the clinical meaning of a predicted probability, and how to incorporate the explanation into decision-making. The input editor further allowed participants to engage in a back-and-forth questioning to confirm that their hypotheses about the model's reasoning were correct. Importantly, we found that participants described the transformations they were applying in terms of higher-level features corresponding to their domain knowledge. Several studies or frameworks of interpretability needs have described the need for users to "sanity check" a model's decision as a way to build trust [6, 11, 22, 31, 55], and we found that the input editor is one way to address this need. In particular, we posit that the interactivity of the editor — giving people the ability engage in back-and-forth probing of the model — is more effective to this end than, say, presenting a score representing the model's reliability.

At the same time, we note some limitations of our current interface, and what they imply for future work. We found that when the nearest neighbor waveforms looked significantly different than expected, participants had difficulty reasoning about why the model thought the neighbors were similar. We posit that part of participants' confusion is due to the fact that the nearest neighbors for some examples are lower quality, as the training data unevenly represented different classes (see Section 3.3.1). Supraventricular ectopic beats, for instance, made up only 2.7% of the training examples. As a result, the model was not able to distinguish this beat from others to the same level of detail or accuracy. Additionally, because there are simply fewer examples from this class, it is less likely that an example will have nearest neighbors that look very similar to it. This consideration of under-representation in the training data, however, did not naturally occur to participants upon seeing the lower-quality neighbors. This observation has a few implications for future work. First, it motivates additional data collection to train models and compute nearest neighbors with data that is more evenly distributed across classes and other important features. More immediately, it suggests the need to be transparent to users about under-representation in the data, and to explain its implications on quality of nearest neighbors. If a user is then presented with an output where the neighbors do not appear to make sense, they are better equipped to understand why this might be the case, rather than be confused about it. Indeed, we found that when we described this phenomena to participants after the conclusion of the study, they were able to understand why under-representation would affect the nearest neighbors — it had just not been on their radar previously. This direction is also supported by the results from Cai et al. [11], who found the need for an *"AI Primer"* for users to explain, in part, *"AI-specific behavior that may be surprising."*

In other cases, participants found it difficult to form hypotheses and apply corresponding transformations given the open-ended nature of the input editor. To imagine ways this problem could be further explored and addressed, we find it useful to compare the capabilities of the editor to interpretability methods that generate counterfactual examples (i.e., similar example(s) that are classified differently) [19, 38, 56]. In the latter, a modified version of the input is generated by automatically finding small transformations that lead to different predictions — while this doesn't require the user to generate their own hypothesis, it can also return unrealistic examples that do not permit further probing. With the editor, users who are familiar with the data are instead able to modify the input in ways they believe should be meaningful given their prior knowledge of and familiarity with the domain. However, as we found, this can make the interaction *too* open-ended in some cases. Here, we find a promising future direction in combining aspects of the two: for example, automatically determining types of transformations that lead to the most change in the output, and narrowing the user's space of hypotheses by communicating this on the interface akin to information scent [42].

Finally, while we demonstrate our interface using an ECG case study to ground our examples and user study in a real-world task, there is a significant opportunity for future work to investigate how these interface modules could be instantiated for other applications.

6 CONCLUSION

In this paper, we present two interface modules to facilitate intuitive assessment of a machine learning model's predictions. Our designs are guided by the motivation to provide users the ability to visualize and probe the model in ways that align with their existing domain knowledge and ways of thinking. Using the interface, users can explore a given input's nearest neighbors in the training data to better understand if and why the model is uncertain, and what high-level features the model is learning. They can further manipulate the input using domain-specific transformations to test hypotheses about the model's reasoning or ensure that it is not overly sensitive to small changes. Through think-aloud studies with 9 physicians/medical students, we demonstrate that our interface helped participants understand the model's reasoning and uncertainty in terms of domain-relevant concepts while building intuition around its capabilities and limitations. The interface modules we present indicate a promising direction for future work aiming to build ML systems that are useful and trustworthy to their real-world users.

REFERENCES

[1] Agnar Aamodt and Enric Plaza. 1994. Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. *AI Communications* 7, 1 (1994), 39–59. https://doi.org/10.3233/AIC-1994-7104

[2] Ajaya Adhikari, David M. J. Tax, Riccardo Satta, and Matthias Faeth. 2019. LEAFAGE: Example-based and Feature importance-based Explanations for Black-box ML models. In *2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, New Orleans, LA, USA, 1–7. https://doi.org/10.1109/FUZZ-IEEE.2019.8858846

[3] Ifeoma Ajunwa. 2016. The Paradox of Automation as Anti-Bias Intervention. *Forthcoming in Cardozo Law Review* (2016).

[4] Solon Barocas, Andrew D. Selbst, and Manish Raghavan. 2020. The hidden assumptions behind counterfactual explanations and principal reasons. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. ACM, Barcelona Spain, 80–89. https://doi.org/10.1145/3351095.3372830

[5] Samyadeep Basu, Philip Pope, and Soheil Feizi. 2020. Influence Functions in Deep Learning Are Fragile. *arXiv:2006.14651 [cs, stat]* (June 2020). http://arxiv.org/abs/2006.14651 arXiv: 2006.14651.

[6] Umang Bhatt, Alice Xiang, Shubham Sharma, Adrian Weller, Ankur Taly, Yunhan Jia, Joydeep Ghosh, Ruchir Puri, José M. F. Moura, and Peter Eckersley. 2020. Explainable machine learning in deployment. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency (FAT* '20)*. Association for Computing Machinery, Barcelona, Spain, 648–657. https://doi.org/10.1145/3351095.3375624

[7] Angie Boggust, Brandon Carter, and Arvind Satyanarayan. 2019. Embedding Comparator: Visualizing Differences in Global Structure and Local Neighborhoods via Small Multiples. arXiv:1912.04853 [cs.HC]

[8] Adrian Bussone, Simone Stumpf, and Dympna O'Sullivan. 2015. The Role of Explanations on Trust and Reliance in Clinical Decision Support Systems. In *2015 International Conference on Healthcare Informatics*. IEEE, Dallas, TX, USA, 160–169. https://doi.org/10.1109/ICHI.2015.26

[9] Carrie J. Cai, Jonas Jongejan, and Jess Holbrook. 2019. The effects of example-based explanations in a machine learning interface. In *Proceedings of the 24th International Conference on Intelligent User Interfaces*. ACM, Marina del Ray California, 258–262. https://doi.org/10.1145/3301275.3302289

[10] Carrie J Cai, Emily Reif, Narayan Hegde, Jason Hipp, Been Kim, Daniel Smilkov, Martin Wattenberg, Fernanda Viegas, Greg S Corrado, Martin C Stumpe, et al. 2019. Human-centered tools for coping with imperfect algorithms during medical decision-making. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–14.

[11] Carrie J Cai, Samantha Winter, David Steiner, Lauren Wilcox, and Michael Terry. 2019. " Hello AI": Uncovering the Onboarding Needs of Medical Practitioners for Human-AI Collaborative Decision-Making. *Proceedings of the ACM on Human-computer Interaction* 3, CSCW (2019), 1–24.

[12] Shan Carter, Zan Armstrong, Ludwig Schubert, Ian Johnson, and Chris Olah. 2019. Exploring Neural Networks with Activation Atlases. *Distill* 4, 3 (March 2019), 10.23915/distill.00015. https://doi.org/10.23915/distill.00015

[13] Rich Caruana, Hooshang Kangarloo, JD Dionisio, Usha Sinha, and David Johnson. 1999. Case-based explanation of non-case-based learning methods.. In *Proceedings of the AMIA Symposium*. American Medical Informatics Association, 212.

[14] Diogo V. Carvalho, Eduardo M. Pereira, and Jaime S. Cardoso. 2019. Machine Learning Interpretability: A Survey on Methods and Metrics. *Electronics* 8, 8 (July 2019), 832. https://doi.org/10.3390/electronics8080832

[15] Finale Doshi-Velez and Been Kim. 2017. Towards A Rigorous Science of Interpretable Machine Learning. *arXiv:1702.08608 [cs, stat]* (March 2017). http://arxiv.org/abs/1702.08608 arXiv: 1702.08608.

[16] Mengnan Du, Ninghao Liu, and Xia Hu. 2019. Techniques for interpretable machine learning. *Commun. ACM* 63, 1 (Dec. 2019), 68–77. https://doi.org/10.1145/3359786

[17] Shayan Fazeli. [n.d.]. *ECG Heartbeat Categorization Dataset.* https://www.kaggle.com/shayanfazeli/heartbeat

[18] Leilani H. Gilpin, David Bau, Ben Z. Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. 2018. Explaining Explanations: An Overview of Interpretability of Machine Learning. In *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*. IEEE, Turin, Italy, 80–89. https://doi.org/10.1109/DSAA.2018.00018

[19] Yash Goyal, Ziyan Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Counterfactual Visual Explanations. In *Proceedings of the 36th International Conference on Machine Learning*, Vol. 97. Long Beach, California, USA. http://proceedings.mlr.press/v97/goyal19a.html arXiv: 1904.07451.

[20] Florian Heimerl and Michael Gleicher. 2018. Interactive analysis of word vector embeddings. In *Computer Graphics Forum*, Vol. 37. Wiley Online Library, 253–265.

[21] Sungsoo Ray Hong, Jessica Hullman, and Enrico Bertini. 2020. Human Factors in Model Interpretability: Industry Practices, Challenges, and Needs. *Proceedings of the ACM on Human-Computer Interaction* 4, CSCW1 (May 2020), 1–26. https://doi.org/10.1145/3392878

[22] Sungsoo Ray Hong, Jessica Hullman, and Enrico Bertini. 2020. Human Factors in Model Interpretability: Industry Practices, Challenges, and Needs. *Proceedings of the ACM on Human-Computer Interaction* 4, CSCW1 (2020), 1–26.

[23] Edwin L Hutchins, James D Hollan, and Donald A Norman. 1985. Direct manipulation interfaces. *Human–computer interaction* 1, 4 (1985), 311–338.

[24] Fei Jiang, Yong Jiang, Hui Zhi, Yi Dong, Hao Li, Sufeng Ma, Yilong Wang, Qiang Dong, Haipeng Shen, and Yongjun Wang. 2017. Artificial intelligence in healthcare: past, present and future. *Stroke and vascular neurology* 2, 4 (2017), 230–243.

[25] Mohammad Kachuee, Shayan Fazeli, and Majid Sarrafzadeh. 2018. ECG Heartbeat Classification: A Deep Transferable Representation. In *2018 IEEE International Conference on Healthcare Informatics (ICHI)*. IEEE, New York, NY, 443–444. https://doi.org/10.1109/ICHI.2018.00092

[26] Been Kim. 2015. *Interactive and Interpretable Machine Learning Models for Human Machine Collaboration.* Ph.D. Dissertation. Massachusetts Institute of Technology, Cambridge, MA.

[27] Been Kim, Rajiv Khanna, and Oluwasanmi O Koyejo. 2016. Examples are not enough, learn to criticize! Criticism for Interpretability. In *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett (Eds.). Curran Associates, Inc., 2280–2288. http://papers.nips.cc/paper/6300-examples-are-not-enough-learn-to-criticize-criticism-for-interpretability.pdf

[28] Pang Wei Koh and Percy Liang. 2017. Understanding Black-box Predictions via Influence Functions. In *Proceedings of the 34th International Conference on Machine Learning*, Vol. 70. Sydney, Australia. http://arxiv.org/abs/1703.04730 arXiv: 1703.04730.

[29] Todd Kulesza, Margaret Burnett, Weng-Keen Wong, and Simone Stumpf. 2015. Principles of Explanatory Debugging to Personalize Interactive Machine Learning. In *Proceedings of the 20th International Conference on Intelligent User Interfaces - IUI '15*. ACM Press, Atlanta, Georgia, USA, 126–137. https://doi.org/10.1145/2678025.2701399

[30] Vivian Lai and Chenhao Tan. 2019. On Human Predictions with Explanations and Predictions of Machine Learning Models: A Case Study on Deception Detection. In *Proceedings of the Conference on Fairness, Accountability, and Transparency - FAT* '19*. ACM Press, Atlanta, GA, USA, 29–38. https://doi.org/10.1145/3287560.3287590

[31] Q Vera Liao, Daniel Gruen, and Sarah Miller. 2020. Questioning the AI: Informing Design Practices for Explainable AI User Experiences. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–15.

[32] Yang Liu, Eunice Jun, Qisheng Li, and Jeffrey Heer. 2019. Latent space cartography: Visual analysis of vector space embeddings. In *Computer Graphics Forum*, Vol. 38. Wiley Online Library, 67–78.

[33] Scott Lundberg and Su-In Lee. 2017. A Unified Approach to Interpreting Model Predictions. In *Advances in Neural Information Processing Systems 30 (NIPS 2017)*. https://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions

[34] Pablo Navarrete Michelini, Hanwen Liu, and Dan Zhu. 2019. Multigrid Backprojection Super–Resolution and Deep Filter Visualization. *Proceedings of the AAAI Conference on Artificial Intelligence* 33 (July 2019), 4642–4650. https://doi.org/10.1609/aaai.v33i01.33014642

[35] George A Miller. 1956. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological review* 63, 2 (1956), 81.

[36] George B Moody and Roger G Mark. 2001. The impact of the MIT-BIH arrhythmia database. *IEEE Engineering in Medicine and Biology Magazine* 20, 3 (2001), 45–50.

[37] Ramaravind K. Mothilal, Amit Sharma, and Chenhao Tan. 2020. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. ACM, Barcelona Spain, 607–617. https://doi.org/10.1145/3351095.3372850

[38] Ramaravind K Mothilal, Amit Sharma, and Chenhao Tan. 2020. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. 607–617.

[39] Sajad Mousavi, Fatemeh Afghah, and U Rajendra Acharya. 2020. HAN-ECG: An Interpretable Atrial Fibrillation Detection Model Using Hierarchical Attention Networks. *arXiv preprint arXiv:2002.05262* (2020).

[40] Ingrid Nunes and Dietmar Jannach. 2017. A Systematic Review and Taxonomy of Explanations in Decision Support and Recommender Systems. *User Modeling and User-Adapted Interaction* 27, 3–5 (Dec. 2017), 393–444. https://doi.org/10.1007/s11257-017-9195-0

[41] Nicolas Papernot and Patrick McDaniel. 2018. Deep k-Nearest Neighbors: Towards Confident, Interpretable and Robust Deep Learning. *arXiv:1803.04765 [cs, stat]* (March 2018). http://arxiv.org/abs/1803.04765 arXiv: 1803.04765.

[42] Peter Pirolli and Stuart Card. 1999. Information foraging. *Psychological review* 106, 4 (1999), 643.

[43] Forough Poursabzi-Sangdeh, Daniel G. Goldstein, Jake M. Hofman, Jennifer Wortman Vaughan, and Hanna Wallach. 2019. Manipulating and Measuring Model Interpretability. *arXiv:1802.07810 [cs]* (Nov. 2019). http://arxiv.org/abs/1802.07810 arXiv: 1802.07810.

[44] Alexander Renkl. 2014. Toward an Instructionally Oriented Theory of Example-Based Learning. *Cognitive Science* 38, 1 (Jan. 2014), 1–37. https://doi.org/10.1111/cogs.12086

[45] Alexander Renkl, Tatjana Hilbert, and Silke Schworm. 2009. Example-Based Learning in Heuristic Domains: A Cognitive Load Theory Account. *Educational Psychology Review* 21, 1 (March 2009), 67–78. https://doi.org/10.1007/s10648-008-9093-4

[46] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, San Francisco California USA, 1135–1144. https://doi.org/10.1145/2939672.2939778

[47] Giovanna Sannino and Giuseppe De Pietro. 2018. A deep learning approach for ECG-based heartbeat classification for arrhythmia detection. *Future Generation Computer Systems* 86 (2018), 446–455.

[48] Jerry W Sayre, Hale Z Toklu, Fan Ye, Joseph Mazza, and Steven Yale. 2017. Case reports, case series–from clinical practice to evidence-based medicine in graduate medical education. *Cureus* 9, 8 (2017).

[49] Andrew D. Selbst, Danah Boyd, Sorelle A. Friedler, Suresh Venkatasubramanian, and Janet Vertesi. 2019. Fairness and Abstraction in Sociotechnical Systems. In *Proceedings of the Conference on Fairness, Accountability, and Transparency* (Atlanta, GA, USA) *(FAT* '19)*. Association for Computing Machinery, New York, NY, USA, 59–68. https://doi.org/10.1145/3287560.3287598

[50] Chung Kwan Shin and Sang Chan Park. 1999. Memory and neural network based expert system. *Expert Systems with Applications* 16, 2 (1999), 145–155.

[51] Kacper Sokol and Peter Flach. 2020. One explanation does not fit all. *KI-Künstliche Intelligenz* (2020), 1–16.

[52] Pascal Sturmfels, Scott Lundberg, and Su-In Lee. 2020. Visualizing the Impact of Feature Attribution Baselines. *Distill* 5, 1 (Jan. 2020), 10.23915/distill.00022. https://doi.org/10.23915/distill.00022

[53] Harini Suresh, Natalie Lao, and Ilaria Liccardi. 2020. Misplaced Trust: Measuring the Interference of Machine Learning in Human Decision-Making. In *WebSci '20: 12th ACM Conference on Web Science, Southampton, UK, July 6-10, 2020*, Emilio Ferrara, Pauline Leonard, and Wendy Hall (Eds.). ACM, 315–324. https://doi.org/10.1145/3394231.3397922

[54] Geoffrey H Tison, Jeffrey Zhang, Francesca N Delling, and Rahul C Deo. 2019. Automated and interpretable patient ECG profiles for disease detection, tracking, and discovery. *Circulation: Cardiovascular Quality and Outcomes* 12, 9 (2019), e005289.

[55] Sana Tonekaboni, Shalmali Joshi, Melissa D McCradden, and Anna Goldenberg. 2019. What Clinicians Want: Contextualizing Explainable Machine Learning for Clinical End Use. In *Machine Learning for Healthcare Conference*. 359–380.

[56] Sandra Wachter, Brent Mittelstadt, and Chris Russell. 2018. Counterfactual Explanations without Opening the Black Box: Automated Decisions and the GDPR. *Harvard Journal of Law & Technology* 31, 2 (March 2018), 841–887. http://arxiv.org/abs/1711.00399 arXiv: 1711.00399.

[57] James Wexler, Mahima Pushkarna, Tolga Bolukbasi, Martin Wattenberg, Fernanda Viégas, and Jimbo Wilson. 2019. The what-if tool: Interactive probing of machine learning models. *IEEE transactions on visualization and computer graphics* 26, 1 (2019), 56–65.

[58] Matthew D Zeiler and Rob Fergus. 2014. Visualizing and understanding convolutional networks. In *European conference on computer vision*. Springer, 818–833.

[59] Muhammad Zubair, Jinsul Kim, and Changwoo Yoon. 2016. An automated ECG beat classification system using convolutional neural networks. In *2016 6th international conference on IT convergence and security (ICITCS)*. IEEE, 1–5.