Label-And-Learn: Visualizing the Likelihood of Machine Learning Classifier's Success During Data Labeling

Yunjia Sun, Edward Lank, & Michael Terry
Cheriton School of Computer Science
University of Waterloo
Waterloo, ON, Canada
y277sun@uwaterloo.ca, lank@uwaterloo.ca, mterry@uwaterloo.ca

ABSTRACT
While machine learning is a powerful tool for the analysis and classification of complex real-world datasets, it is still challenging, particularly for developers with limited expertise, to incorporate this technology into their software systems. The first step in machine learning, data labeling, is traditionally thought of as a tedious, unavoidable task in building a machine learning classifier. However, in this paper, we argue that it can also serve as the first opportunity for developers to gain insight into their dataset. Through a Label-and-Learn interface, we explore visualization strategies that leverage the data labeling task to enhance developers' knowledge about their dataset, including the likely success of the classifier and the rationale behind the classifier's decisions. At the same time, we show that the visualizations also improve users' labeling experience by showing them the impact they have made on classifier performance. We assess the visualizations in Label-and-Learn and experimentally demonstrate their value to software developers who seek to assess the utility of machine learning during the data labeling process.

ACM Classification Keywords
H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous

Author Keywords
Machine learning; visualization; data labeling.

INTRODUCTION
Machine learning is a powerful tool for modeling and classifying data. From spam filtering to recommender systems, many of the proactive computational systems that we interact with have some aspect of data-driven machine learning. The wide application of machine learning has driven its commoditization through deployed services such as Google's Prediction Application Programming Interface and Amazon's Machine Learning Service. These services aim to allow software developers to quickly incorporate existing machine learning technologies into a system under development.

However, using machine learning to solve a new problem is still non-trivial. In particular, both selecting features and understanding classifier behavior require significant expertise. To aid machine learning developers, visualizations of clustering[20], receiver operating characteristic (ROC) curves [3], 2d projection of multi-dimensional data [14], and confusion matrices [22] attempt to communicate a compact summary of a particular view of the classifier. While these visualizations often work well for developers who are familiar with the behavior of machine learning classifiers, researchers have argued that they do not support less experienced users of machine learning, e.g. software developers seeking to apply machine learning algorithms to their systems, end users seeking to better personalize their intelligent application. For these less experienced users, researchers have designed systems that allow users to interactively train and assess system behavior [4, 1, 5], systems that compare the performances of different parameter configurations [19, 17], and mechanisms that communicate how and why a system classified an instance in a specific way [12, 10].

Regardless of experience level, there are a series of steps required to deploy a machine learning classifier or system to solve a real-world problem. Figure 1 depicts the process associated with incorporating supervised machine learning into an application. One task necessary to develop any machine learning system is the provision of labeled data, i.e. data that has been pre-classified typically by a person who examines and labels data to generate a sufficient dataset for the classifier to learn from. While unavoidable, data labeling is tedious and expensive, with a cost either in designer/user's time (if the developer labels data) or in money (if paying someone else to label data). On the other hand, we believe that labeling can serve as the first opportunity for developers to get insight into their dataset. Ideally, as labeling is happening, one would prefer to be able to answer questions such as:

1. Can my classifier learn the desired concept well enough, given the data and features available? Is it worth more investment in labeling?

IUI 2017, March 13 - 16, 2017, Limassol, Cyprus

ISBN 978-1-4503-4348-0/17/03…$15.00

DOI: http://dx.doi.org/10.1145/3025171.3025208

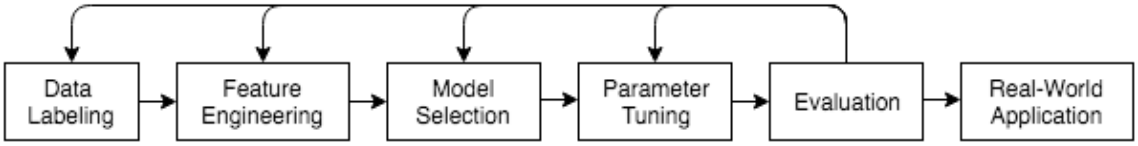

Figure 1: Machine Learning Pipeline

2. What does the classifier know about my dataset? What does it not know yet?
3. How will the classifier make decisions based on its knowledge?
4. How will this single datum affect the performance of the classifier? How useful will this new datum be?
5. How stable is my classifier? If the classifier incorporates new data during deployment, what data will likely affect the classification?

Systems exist to semi-automatically or automatically generate features, select models, and tune parameters, but labeling remains a task that is typically performed by human labelers. Given even a small amount of labeled data, a classifier can begin to try to make predictions, but, to be effective, a machine learning system needs a sufficient amount of labeled data. The feedback mechanisms and interactive machine learning systems described above are invaluable tools to help developers choose features, select models, and tune parameters, but these tools do not reveal how much more data labeling is required and whether additional data labeling will help classifier performance.

In this paper, we explore the design of tools to help assess machine learning classifier behavior during the data labeling stage. Specifically, we describe a *Label-and-Learn* interface which leverages a series of visualizations to communicate dynamic attributes of the classifier to labelers, with the goal of allowing labelers to determine the classifier's likely eventual success, current performance, and the worthiness of additional labeling. We evaluate our Label-and-Learn visualizations against a generic interface and show that users' mental model of classifier performance is significantly enhanced, and that users feel more engaged when performing the labeling task.

RELATED WORK

The goal of services such as Google's Prediction API or Amazon's Machine Learning Service is to allow developers of systems to incorporate machine learning without the need to acquire deep understanding of the algorithms or mathematical models used within these systems. However, while developers may not need to know the mathematical foundations of machine learning algorithms, to incorporate machine learning into deployed software applications developers must understand how machine learning algorithms will behave and they must have available some mechanisms to modify and/or improve the recognition generated by the algorithms. To date, most work in this area has been in the domain of interactive machine learning and in visualizations to communicate recognizer behavior. Referring to Figure 1, we classify much of this past work as supporting Feature Engineering, Model Selection, and Parameter Tuning.

Interactive machine learning systems include the Crayons system [4], a set of interactive machine learning systems developed by Amershi et al. (CueFlik, ReGroup, and CueT) [5, 1, 2], a set of systems by Patel et al. (Gestalt, Prospect, and Hindsight) [18, 19, 17], systems for interactive computer vision programming such as Eyepatch [13], and toolkits for gesture recognition [6]. These systems help software developers (and end-users) build machine learning systems by allowing users to label an initial set of data, generating classified result, and interactively allowing users to re-label data or refine features to improve classifier performance. To simplify the process for their users, these systems mask feature selection, model selection, and parameter tuning from the user. Users know nothing about the implementation of the system so they do not know whether it is applicable to other datasets. Overall, the primary goal of these systems is to allow users to examine the output and label more data so that the system can improve its classification result.

In contrast, systems such as Gestalt[18], Prospect[19], and Hindsight[17] seek to reveal details of features, models, and parameters to software developers in a structured way to enhance comprehension. For example, Gestalt presents the design of machine learning systems as a step-by-step wizard, guiding the developers through feature engineering, model selection, and parameter tuning. Prospect focuses specifically on feature generation, leveraging simple visualizations to highlight areas of uncertainty in data so that users can enhance the features used by the system. Finally, Hindsight allows direct comparison of different machine learning models given a labeled dataset.

Alongside an ability to interactively create and contrast aspects of system behavior, researchers have also been seeking to design tools that communicate the rationale behind classifier behavior to the end-users of machine learning applications, so that they can establish trust with the application and know how to personalize it. Examples of this work include Lim and Dey's visualization toolkit[12] which depicts how rule-based classifiers, decision trees, naive Bayes, and hidden Markov model classifiers arrive at a result. In applying their toolkit to intelligent systems, they found that users desired streamlined explanations with detail available on demand. Similarly, Kulesza's EluciDebug[10] highlights features used to arrive at a prediction, thus helping users debug their classifiers by building a sound mental model.

There is significant additional research in developing machine learning systems. However, let us return to the questions we

asked in the introduction: How can we allow non-experts to predict whether machine learning will work effectively on a problem domain and how much labeling work is necessary before a classifier will stabilize. Ideally, one would like to answer these questions as early as possible in classifier design, preferably prior to extensive labeling work.

Labeling

We are not the first to note that a significant bottleneck of building a good classifier is the quality and quantity of labeled training data available; nor are we the first to note that labeling data is a time-consuming and costly process in the pipeline. Researchers have explored mechanisms to improve the quality of labels through *structured labeling* [9]. Researchers also work to reduce the number of labels needed by the classifier through *active learning* [21]. Finally researchers have also explored ways to maximize the utility of labeling by enhancing the user's ability to pick the most informative instance to label next [8]. Each of these systems seeks to increase the efficiency of labeling, but does not enhance labelers' understanding of the data or improve the labeling experience.

Beyond the efficiency of the labeling process, one significant question to ask is whether or not more labeling is worthwhile. How much accuracy will an additional label provide? How much uncertainty currently exists in the system? Will the classifier ever converge given additional labeled data? Our work focuses on helping users gain insight into their dataset and classifier, so that they can get motivated by their inner curiosity about the data, as well as estimate the classifier's likely performance at an early stage.

LABEL-AND-LEARN SYSTEM

Label-and-Learn is a system designed for developers who want to train a machine learning classifier to solve a specific problem. It can help them estimate the classifier's expected performance at the early construction phase of machine learning during labeling. In this section, we introduce the system. In particular, we describe the specific kind of task addressed, the machine learning model used, and the visualizations in the interface. Much of our design discussion in this section is grounded in the specific example used to design our classifier and in the specific task used in our experiment. This is hardly unique to our work; Amershi's systems focused on Group Creation [1], Network Alarm Triage [2], and Image Search [5]. The first system used Naive Bayes, while the last two used Nearest Neighbour classification. In fact, virtually all systems that seek to demystify the machine learning process also initially focus on a single problem domain, a single stage in the machine learning pipeline, or a single machine learning algorithm [1, 2, 4, 5, 17, 18, 19]. Later in this paper, we will explore the application of our system to other algorithms and recognition domains.

Task

To drive the design of our system, consider a basic machine learning task, Named-Entity Recognition [16]. In Named-Entity Recognition, given a string that matches one of the named-entities, the classifier should identify whether it actually refers to the named-entity or not (e.g. is '2016' a year or does it represent some other quantity such as a price, a count of items, etc.). We choose Named-Entity Recognition for the following reasons: (1) It is used in many real-world systems, (2) It usually requires a large amount of human labeling work that is tedious and error-prone, (3) The features are easy to identify and understand.

The Backend Classifier

For our backend classifier, we choose naive Bayes as our model because it has competitive performance in text categorization, because it is easy to implement, understand and visualize, and because it has been used as a test algorithm for past interactive machine learning research systems [1]. We add a small modification to the traditional naive Bayes and bag-of-words model. We record the prior probability, which is the ratio of positive instances in the training data, and three sets of dictionaries:

- the previous dictionary, containing the tokens before the matched string,
- the current dictionary, containing the strings of named entities, and
- the next dictionary, containing the tokens after the matched string.

To identify whether the string t_3 is actually the named-entity in the context $[t_1][t_2][t_3][t_4][t_5]$, the system uses the following formula to calculate its positive probability:

$$p(+|[t_1][t_2][t_3][t_4][t_5]) = \frac{p(positive)}{p(positive)+p(negative)} \quad (1)$$

where:

$$\begin{aligned} p(positive) &= p(+)\prod_{1\le i\le 5} p(t_i|+) \\ &= p(+)\prod_{1\le i\le 5} \frac{count(t_{i,+}\ \ in\ \ set_i)}{count(pos\ \ in\ \ trainingset)} \end{aligned} \quad (2)$$

Here, $p(+)$ is the ratio of positive instances in the training set, $count(pos\ \ in\ \ trainingset)$ is the number of positive instances in the training set, $count(t_{i,+}\ \ in\ \ set_i)$ is the number of t_i's positive occurrences in set_i. Negative probabilities use a similar representation. Specifically, $set_1 = set_2 =$previous dictionary, $set_3 =$ current dictionary, $set_4 = set_5 =$next dictionary. We use three different sets to emphasize the order of the words, as the terms that are more likely to appear before the named-entity and the terms that are more likely to appear after the named-entity might be different.

Our system also incorporates Active Learning to request labels for the most informative data. The informativeness (Information Gain) is calculated as the product of the instance's entropy (uncertainty) and similarity. The less certain the system is about the instance, and the more similar the instance is to the rest of dataset, the more informative the instance is. Specifically, the system selects the instance x^*_{ID} according to the criteria below[21]:

$$x^*_{ID} = \operatorname*{argmax}_x uncertainty(x) \times similarity(x)$$

$$uncertainty(x) = -p(+|x)\log(p(+|x)) - p(-|x)\log(p(-|x))$$

$$similarity(x) = \sum_{x' \in U} \text{sim}(x, x') = \sum_{1 \le i \le 5} count(t_{i,x} \ \ in \ \ set_i)$$

which is the sum of the number of occurrences for each term in the set of its position.

We provide users with two kinds of information to evaluate the current system:
(1) Before labeling the training set, users need to label 100 test data, which will be re-classified whenever the system incorporates a new labeled data and updates its model. Users can evaluate the classifier based on its accuracy on the test data.
(2) Before labeling each training datum, users can see the current classifier's prediction on it. Users can evaluate the classifier by whether the prediction matches their decision.

Visualizations

We used an iterative design process [15] and designed more than 20 different visualizations based upon common strucutres including trees, heat maps, floating bubbles, clusters, and word clouds. We introduced visualization subsets to participants in pilot studies and asked them to leverage different instances of the visualizations to identify changes and explain the reasons for changes in classifier performance. Our two primary visualizations (Figure 2-b,-f) were selected as most representative of convergence/non-convergence, and Figure 2-c, -d were selected as most representative of classifier behavior. As suggested in [10], we avoid overwhelming users with too many data points or visualization that are too eccentric. The visualizations selected cover the following aspects of the information one wants to know about machine learning: visualizations that use labeled or unlabeled data, visualizations that need or do not need a built classifier, visualizations that focus on features or on instances, visualizations that show all of the data or only the interesting data, visualizations that show data in their original forms or in abstract forms such as points.

Shown in Figure 2, our interface consists of six windows (clockwise from top middle): the Labeling Window(a), the Test Set Distribution(b), the Influential Terms(c), the Current Prediction(d), the Information Gain(e), and the Labeling Progress(f). The Labeling Window is the interface where users perform the labeling task.

The *Test Set Distribution*, shown in Figure 2-b, shows the predictions for the 100 test data the user first labeled. The blue points are those labeled as positive and the red points are those labeled as negative. The x-axis represents the instance's positiveness predicted by the system, with the right side representing positive and the left side representing negative. When a user wants to know why an instance is wrongly predicted, or why the system is uncertain about an instance, he can click on a point and examine the data the point represents.

The test set is fixed through the experiment session for the participants who are machine learning novices to solidify their understanding of the difference between training set and test set. In the mean time, participants can also evaluate the classifier's performance from other perspectives through different visualizations in Label-and-Learn. For real-life application of the system, the test set can be dynamically changed as more data are being labeled and leave-one-out cross validation can be used to evaluate the performance.

Shown in Figure 2-c, the *Influential Terms* view shows the 24 most influential terms in the previous dictionary, the current dictionary, and the next dictionary. The influential terms are those with strong positiveness or negativeness as well as a frequent occurrence. The bar for each feature shows its *positiveness* and *stability*. Blue and the right side means positive, while red and the left side means negative. Stability is the likelihood that an instance or a feature's confidence will be little affected by a new label with the same feature value, a function of information we have about a feature. A feature's stability is related to its occurrence in the training set. If a term is predicted as having a positive influence with high-stability, e.g. it appeared 900 times as positive and 100 times as negative, another 8 negative examples of this feature would still result in around 90% positive. If a term is predicted as having a positive influence with low-stability, e.g. it appeared 9 times as positive and once as negative, another 8 negative examples would significantly change the feature to be only 50% positive. In the visualization, the more condensed and saturated the region is, the less deeply the positiveness will be affected by new labels, i.e. it is more stable. Conversely, the more area covered, and the lighter the shading, the more the positiveness will be affected by new labels. If the color is grey everywhere, it means there is no information about the term at this moment, and the term has no influence on the classifier's decision. Its positiveness will be entirely determined by the first label so it has the highest unstability. This visualization helps users quickly locate the most influential features, and see whether these features match their knowledge about the task, what features are missing, and whether the features are stable.

For the current example that the user is labeling, *Current Prediction* (Figure 2-d) shows the *positiveness* and *stability* of the current instance's features, prior probability, and final result, so that user understands why the system makes the current prediction. This visualization uses the same color-cues and positioning as in Influential Terms: blue and the right side means positive while red and the left side means negative, and the color saturation reflects the feature's stability. This visualization is designed to help users understand why the classifier makes the current prediction and understand which features are positively affecting the classifier in making a correct decision.

Information Gain (Figure 2-e) visualizes all the data in the unlabeled pool with their information gain. Here we define information gain as the product of the datum's uncertainty and similarity. *Uncertainty* is the datum's entropy. The similarity between two data items is the overlap in terms between the two data items. Generalized over the entire set, the *Similarity* of a datum is the average of the similarity between it and each datum in the whole dataset. The y-axis represents the uncertainty of each data point, and the upper points are data with high uncertainty. The x-axis represents the similarity score of each data point, and the points on the right are the

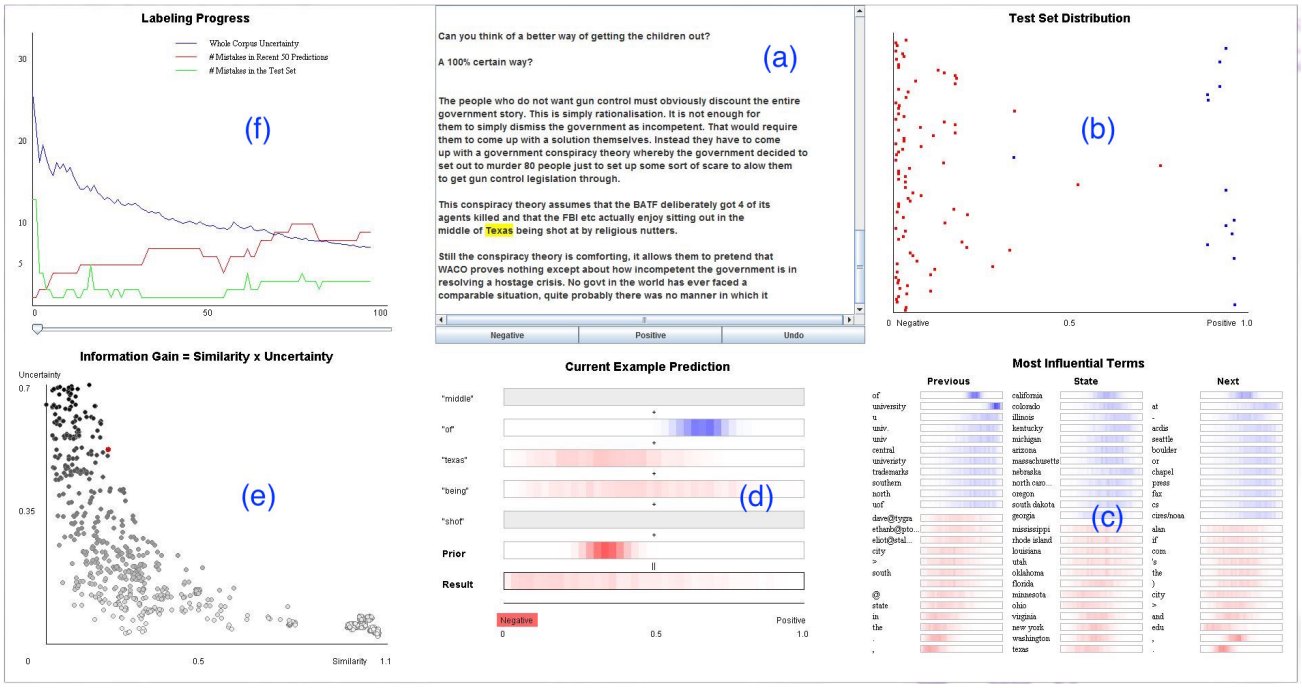

Figure 2: Visualizations in Label-and-Learn: (a) Labeling Window (b) Test Set Distribution (c) Influential Terms (d) Current Prediction (e) Information Gain (f) Labeling Progress

most representative data. The shades of the points also reflect their uncertainty. The system always chooses the top-right data point (highlighted in red) for the user to label next, as it has the largest product of similarity and uncertainty. This visualization shows users why this data is selected to be labeled, how the unlabeled data is distributed according to similarity, and the uncertainty in classifier's predictions. It is easy to see that if all the representative data points have low entropy, and only a few low-similarity data points have high entropy, then we can say the classifier is quite certain about most of the data and already has enough information to make predictions.

Labeling Progress provides a record of the labeler's progress with three trend lines (Figure 2-f). As the user labels the data, the system keeps track of the following information:

- The red line indicates the number of mismatches in the last 50 predictions vs user's labels.
- The green line indicates the number of mistakes the current classifier makes in predicting the test set.
- The blue line indicates the overall uncertainty the system has across all data, which is calculated as $\sum_{x \in U} H(x)$, where $H(x)$ is the entropy of data x.

This visualization can help users answer the following questions: Is the classifier good enough (Observe if the three lines are close to zero)? Is the classifier still learning (Observe if the red line and the blue line are going up but the green line is going down)? Is it going to be better with more labels (Observe if the three lines have a tendency to go down)? This visualization can also help users see what they have achieved so far and motivate them to continue labeling.

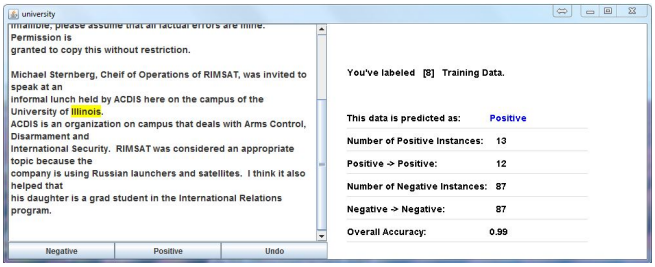

Figure 3: Traditional Interface

Traditional Interface

In designing our visualizations for our Label-and-Learn interface, we leveraged information that was readily available from the classifier during the labeling process. As a result, one question we had was whether providing an interface that supported visualization of the data was necessary, or if simply providing this data was sufficient. To test the utility of the visualizations (versus the information itself), we implemented a traditional version of the labeling interface with only text information (Figure 3). It shows the classifier's prediction on the current data and its performance on the test set. It also helps users keep track of the number of labels they have already provided.

EVALUATION

We evaluated our Label-and-Learn system via the following research questions:

RQ1: Can the visualizations help users better understand the current classifier's performance and better predict the classifier's success than the traditional interface?

RQ2: Do the visualizations give users a better labeling experience?

RQ3: What kind of information about the classifier do the users want to know? Which visualization is the most helpful?

The first question maps specifically onto the five questions we ask in the introduction regarding overall system performance, i.e. will the system produce a good classifier, has the classifier stabilized, and, if not, how much more labeling must be done? The remaining two research questions focus specifically on user experience aspects of labeling.

Participants

We recruited 20 participants. The participants were screened to ensure they understood simple charts and graphs (e.g. students with a high school math background, or people who can use Excel). As the study examines the usefulness of the visualizations, and our visualizations are non-traditional, those who cannot understand simple graphs and charts will introduce a confound to the experiment; they lack the essential knowledge to build a classifier. There are already visualization tools supporting machine learning experts, and our visualizations are designed to support machine learning novices, so we did not accept participants who had taken a machine learning course, or had written code about machine learning algorithms.

Machine Learning Tasks

Each participant used both versions of the system to build two classifiers by labeling. During the classifier building process, we hypothesize that participants will also build a mental model for each classifier. To avoid learning effects, the Named-Entity Recognition tasks they perform on the two versions of the systems are different. The two tasks are a university task and an address task.

University Task

In this task, participants trained the system to determine whether the highlighted state is part of a university's official name. For example, ***New York*** *University* is a positive instance, while *the* ***Florida*** *researchers* is a negative instance. For this task, a workable classifier can be built, because it has strong positive features such as "university" and "of".

Address Task

In this task, participants trained the system to determine whether the highlighted state is part of a full address, and refers to the state. For example, *49 Locust Avenue, Suite 104; New Canaan,* ***Connecticut*** *06840* is a positive instance, while *I went to* ***California*** *for a vacation* is a negative instance. For this task, a workable classifier cannot be built, because the dataset does not have strong positive features. Most of the positive indicators are specific city names and postcodes. Only when the training data covers all the city names and zip codes can a successful classifier be built.

The documents were randomly selected from the 20 News-Groups dataset[1]. Before the experiment, we processed the data and extracted the matched strings of state names. Each matched string is counted as one instance. Participants were only required to label the highlighted string as positive or negative (Figure 2), without identifying all the state names in the documents.

[1] http://qwone.com/ jason/20Newsgroups/

Procedure

The study used a 2×2 factorial design with two factors: system version and dataset. Each participant was randomly assigned a combination of system version and task: traditional-university and visualization-address, or traditional-address and visualization-university. The combinations assigned to the participants and the orders of the two systems were counterbalanced. The procedure for each participant is listed below:

1. Participants received a short tutorial on machine learning and the naive Bayes classifier to understand how the system learns and predicts.
2. Participants used the first interface to build a classifier for the first task by labeling 100 test data and 400 training data.
   (a) Participants received a short tutorial about the interface design and the meaning for each number and figure. They were also shown a quiz, explained below, to understand what information they should look for while labeling.
   (b) Participants labeled 100 data for testing.
   (c) Participants labeled data with intervals of 50, 50, 100, 200 instances, and completed the quiz (mentioned above and described below) after each interval (4 in total).
3. Participants used the second interface to build a classifier for the second task by labeling 100 test data and 400 training data (Same procedure as in 2).
4. Participants rated the workload for each system using the NASA-TLX[7].
5. Participants rated the utility of each visualization on a Likert-Scale, participated in an interview and provided feedback on the overall the system.

The experiment slot was 120 minutes, but most of the participants finished within 90 minutes. As real-life data labeling task can take days or weeks, we try to simulate that situation by introducing certain boredom to the participants to demonstrate the advantage of our system. Labeling only a small amount of data cannot understand the dataset's property enough either.

We asked participants to complete the quiz four times in each trial because the classifier changes; with additional labeled data, the classifier makes different decisions each time on the same data. Participants' understanding of the data also changes as they know more about the dataset. The interval between each quiz increases because both the classifier's model and the participant's mental model of the classifier change frequently early in the experiment and become stable and less likely to change after seeing more data.

Measures and Quiz

We evaluated the utility of our system by collecting the following data during the experiment.

First, as noted in the experimental procedure, after labeling 50, 100, 200, 400 instances, participants completed a short quiz with the following questions to assess their mental model:

Prediction of the success: Predict if the dataset and the algorithm is good enough such that the classifier will be successful with 1000 training data, and decide whether it is time to stop labeling because the performance is unlikely to improve. By successful, we mean that more than 80% of positive instances are predicted correctly, and more than 80% of negative instances are predicted correctly.

Stability of the features: Select the most unstable feature from the given choices. This question tests if participants can identify weaknesses of the classifier. The features in the choices include: extremely positive/negative features, features without significant positiveness/negativeness, features that appeared many times, features that appeared less frequently, and features that never appeared. Not all the features in the given choices appeared in the Influential Terms visualization.

Classifier's decisions and reasons: Predict the classifier's decisions on three instances and provide reasons by selecting the influencing factors – either features or prior probability. The questions include instances that are always predicted as negative, instances that are always predicted as positive, instances whose predictions change after more labels, and instances that have the same number of positive factors and negative factors.

Second, after finishing the labeling task and the mental model quizzes, participants rated the cognitive workload for the two systems using the NASA-TLX questionnaire. Participants also rated the utility for each visualization on a Likert-Scale. Finally, to complete the study, participants provided feedback in a semi-structured interview exploring their preference between the visualization interface and the traditional interface, and the visualization features they found useful.

Hypotheses

Formally, with respect to quantitative measures, we evaluate the following hypotheses:

- Users have more correct judgements about the success of the classifier and the time to stop labeling (they receive a higher score in the related quiz questions) when using the visualizations compared to using the text data.
- Users are able to identify more features that are unstable (they receive higher score in the related quiz questions) when using the visualizations compared to using the text data.
- Users are able to predict the classifer's decisions more accurately and reasonably (they receive higher score in the related quiz questions) when using the visualizations compared to using the text data.
- Users' cognitive workload was lower (lower score in the NASA-TLX) when using the visualizations compared to using the text data.

Result

As each measure is influenced by two factors – the inclusion of the visualizations and the dataset – and the experiment is a mixed design, we perform non-parametric multi-factor analysis using Wobbrock et al.'s aligned rank transform approach [23]. To perform this analysis, this method first "aligns" the data for each effect (main or interaction) before applying averaged ranks. Once transformed, a common ANOVA procedures is used to measure main effects. The benefit of this approach is that the analysis leaverages the F-test, which is a common analytical function in statistical toolboxes [23].

Success of the Classifier

In each quiz, participants decided whether the dataset and the algorithm are good enough such that the classifier will be successful with 1000 training data. They also decided whether it was time to stop labeling because the performance was unlikely to improve. Participants receive 1 score for correctly answering each question in each quiz (8 points in total). The mean and standard deviation are shown in Figure 4-a. Participants scored higher when using the visualization system. They also scored higher when doing the *university* task. We found significant effect from the system version ($F_{1,38} = 4.20, p < 0.05, \eta^2 = 0.10$). We also found significant effect from the dataset ($F_{1,38} = 15.99, p < 10^{-3}, \eta^2 = 0.30$), which may be because participants were optimistic about the *address* dataset's success at the beginning, while a workable classifier cannot be built for the task. No significant effect was found from the interaction between the system version and the dataset ($F_{1,36} = 1.72, p = 0.20, \eta^2 = 0.04$).

Stability of the Features

In each quiz, participants selected the most unstable feature from the given choices. Higher score means more correct answers. The bar graph (Figure 4-b) shows that participants score higher when using the visualizations. We found a highly significant effect from the system version ($F_{1,38} = 125.94, p < 10^{-12}, \eta^2 = 0.77$), as participants can simply look up each feature's presence and stability from *Most Influential Terms*. Highly significant effect was also found from the dataset ($F_{1,38} = 64.71, p < 10^{-9}, \eta^2 = 0.63$) and the interaction between dataset and system ($F_{1,36} = 35.36, p < 10^{-6}, \eta^2 = 0.46$). When participants were labeling for the *university* dataset, they are relating the text with the actual universities because they know of the existence of the universities, thus paying more attention to the data even without the visualizations.

Classifier's Decision and Reasons

In each quiz, participants predicted the classifier's decisions on three instances, and also provided the reasons by selecting the influencing factors from the five features and prior probability. Participants receive 4 points for each correct prediction and 1 point for judging the influence for each factor (6 points in total). There are 10 (points) $\times$ 3 (questions) $\times$ 4 (quizzes) = 120 points for each trial.

Figure 4-c shows the mean and standard deviation for each condition; the visualization system helped users predict the classifier's decision more accurately and reasonably. We found a highly significant influence from the system ($F_{1,38} = 115.41, p < 10^{-12}, \eta^2 = 0.75$). We also found a significant influence from the dataset ($F_{1,38} = 39.88, p < 10^{-6}, \eta^2 = 0.51$) and the interaction of the dataset and system ($F_{1,36} =$

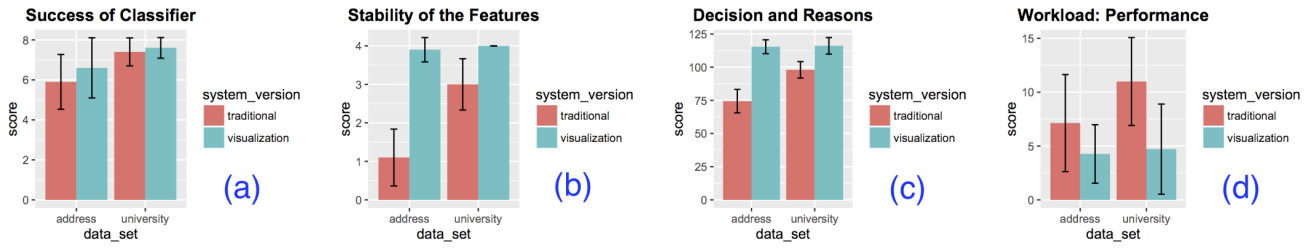

Figure 4: Quantitative Result

38.02, $p < 10^{-6}, \eta^2 = 0.51$). As in the *university* dataset, features "university" and "of" are highly positive, and make it easy to make a judgement. However, in the *address* dataset, there is no such obvious feature, and positive terms are typically city names and zip codes. It is hard to keep track of all data without the help of visualizations.

As is shown in Figure 4-d, participants were more confident in their performance when using the visualizations, as in the NASA-TLX, better performance means lower workload. ANOVA shows highly significant influence from system version ($F_{1,38} = 18.902, p < 10^{-4}, \eta^2 = 0.33$), because participants can always find the correct answers with the visualization system, but can only retrieve from the impression when using the traditional system. We also found significant influence from the dataset ($F_{1,38} = 4.23, p < 0.05, \eta^2 = 0.10$) and interaction of system version and dataset ($F_{1,36} = 4.40, p < 0.05, \eta^2 = 0.11$). This may be because the *university* dataset has more positive instances, so the model is more complicated than the *address* dataset, and it is harder to perform analysis without the visualizations.

In the interview, we also asked the participants their preference between the two system versions. All participants preferred the visualization version, as it provides more information about the classifier and helps answer quiz questions. We also asked their preference between the two system versions if they were just asked to label without answering the quizzes. 16 participants preferred the visualization version, saying they want to see what is happening in the system and more visual information rather than plain numbers and texts, 2 participants reported same level of preference, and 2 others preferred the traditional version. The participants who preferred the traditional interface noted in interviews that they were not particularly interested in either task; it remains to be seen whether, if participants had a more interesting task, an even larger proportion would want to understand more about classifier evolution. Note that, despite four participants not preferring the visualizations, the 16/20 majority result suggests that these visualizations can help classifier builders understand their dataset by labeling as well as help labelers become more engaged in their labeling task.

Utility of Individual Visualizations

After the participants finished the labeling tasks, they rated the utility of each visualization on a Likert-Scale. Overall, participants found the visualizations useful in helping them understand the classifier. As shown in Figure 5, with the exception of Information Gain, all visualizations received mean scores above 5 on a 7-point Liker Scale. Information Gain received a mean score of 2.8; however, despite its low mean score some participants still rated it highly for utility, e.g., as P2 noted in interviews: *"Information Gain is very useful. I can see how the graph changes and where most data lies."* It may have been the case that the interpretation of information gain was less clear for some of our participants.

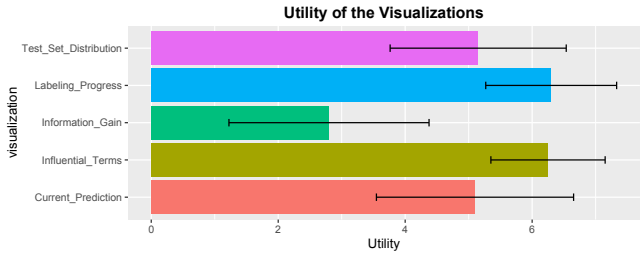

Figure 5: Utility of the Visualizations

DISCUSSION AND LIMITATIONS

Discussion

Experimental data from our study indicate that our hypotheses were all supported, and, in each case, we can reject the null hypothesis. This result is perhaps unsurprising: We do know that visualizations can serve as a valuable tool to communicate information to individuals, and the use of visualizations allows the depiction of data in context. As a result, while it was important to ensure that non-machine-learning experts could glean data from our visualizations in a way that was significantly better than without visualizations, we feel that the more important implications of this work extend beyond this discrete comparison.

As we noted in the introduction, supervised machine learning depends on data and, in particular, on sufficient labeled data to learn models. Feature engineering, automatic model selection, feedback and feedforward mechanisms for parameter tuning – there exist many different techniques for automating parts of the machine learning pipeline, but, at heart, to do any learning at all, the system needs labeled data, usually provided by labelers, to generate a model from data and then use that model to classify new data. A machine learning classifier can attempt to predict at any point in time, even with no labeled data. What labeled data does is allow the classifier to assign

values and ranges to features, to model distributions, and to refine estimates for parameters. We hypothesized that watching this process converge during labeling, or even during some time-consuming unsupervised machine learning problems that do not need labeling, would be beneficial because, if information was well-presented, developers or users attempting to train a classifier would know at an earlier point whether machine learning would effectively learn a model on their dataset and how much more work was needed before the classifier's stability and accuracy converged to an acceptable level for deployment. As a result, both from the perspective of the ability to answer the five questions posed in the introduction and from the perspective of mitigating the tedium of labeling data, it appears that the use of a *Label-and-Learn* paradigm is beneficial in the machine learning pipeline.

Limitations

It is not hard to notice that some information (e.g. information gain) is missing from the control condition. It is a limitation in the study. However, it is impossible to convey every piece information presented in the visualizations in textual format; visualizations, by their nature, are designed to compactly represent data and a textual interface that contains too much information would introduce contradictory confounds of information overload.

On the other hand, there are three questions that we want to answer from our experiment:

Q1: Is a visualization interface helpful for understanding classifier performance while labeling?
Q2: Are the visualizations at all motivating?
Q3: Which visualizations are most helpful for participants?

Our experiment is designed to answer each of these questions in turn, and the design is influenced by our adherence to the scientific method:

- We believe visualizations are useful (our hypothesis).
- Therefore, how could we design an experiment that would argue that visualizations are useless? (Our null hypothesis).
- Our answer: take a really simple task where someone could infer the answer without visualizations and see whether the visualizations help.

Figure 2 shows the labeling for the university task. At this point, the most influential terms in the classifier (Figure 2-c) are two previous terms "University" and "of". As data progresses, next terms evolve to include "State" and "University" (e.g. California State University). It should be trivial for any labeler to figure this out and understand the stability of features by inspection without an interface to help with understanding terms and behavior. Our goal initially was first to identify a task where the visualization really was not necessary at all – a user could probably infer the answer – and then see whether it was still helpful. As rationale for decision making gets more complex, it is likely that visualizations and, by extension, our interface might be even more helpful.

While the quizzes provide data for question 1, questions two and three are addressed through NASA-TLX evaluations and interviews. Data collected from these sources allows us to assess participant interest in understanding the classifier and to assess the utility of each visualization.

As the interfaces provide the classifier's prediction on the datum currently being labeled, we were concerned about the bias it might bring to the labeler's correctness. It turned out to be unnecessary. As observed in the experiments, at the beginning of the labeling, the participants complained about the mistakes made by the immature classifier, so they quickly learned not to trust its prediction.

Considering the background differences of each participant, we limited our experiment only to the labeling process. This enables us to conduct a statistics-based study to confirm the effectiveness of the interface where the participants worked in a controlled experiment setup. However, it is true that future work to study the utility of the visualizations on the full classifier construction cycle would be a valuable next step in this research.

GENERALIZABILITY

One final question that is important to explore is on generalizability. Our system was designed around specific tasks (named-entity recognition) using a specific model (naive Bayes). Does it generalize to other domains?

Conceptually, our Label-and-Learn interface, Figure 2, was designed to include both generic visualizations, which could be applied to virtually any labeling task, and algorithm-specific visualizations which would have to be adapted depending on the information that could be gleaned from the classifier to develop an understanding of its performance. For example, the Test Set Distribution (Figure 2-b) and the Labeling Progress Visualization (Figure 2-f) make use of a labeled test set to measure classifier performance at the present point in time and the change in classifier performance as labeling progresses. Visualizations that show performance on a test set are useful for any machine learning classifier, as the goal is to achieve acceptable performance and the best way to measure performance is to evaluate a classifier on a test set. The three visualizations along the bottom of Figure 2 are more tuned to the specific problem domain we explore. The Influential Terms and the Current Prediction visualizations (Figure 2-c and -d) can be applied to any machine learning systems that use explicit feature values from a dataset to classify data, e.g. decision trees, support vector machines (SVM), and rule-based classifiers; these visualizations are, however, ill-suited to machine learning tools where the match between features in the model and prediction are less explicit or are hidden, e.g. hidden Markov models (HMMs) or neural networks. For machine learning models such as HMMs and neural networks, there exist alternative visualizations to explain classifier reasoning (see Figure 6). Finally, looking at our initial Label-and-Learn interface, the information gain (Figure 2-e) assumes an explicit connection between labels, features, and how labeling affects the decisions of other instances that share common features with the current labeled instance. When this information cannot be extracted from the data and classifier, one option would be to visualize uncertainty as information gain, or one could instead choose other active learning schemes from [21] and visualize these schemes as a proxy for information gain. In summary, two of

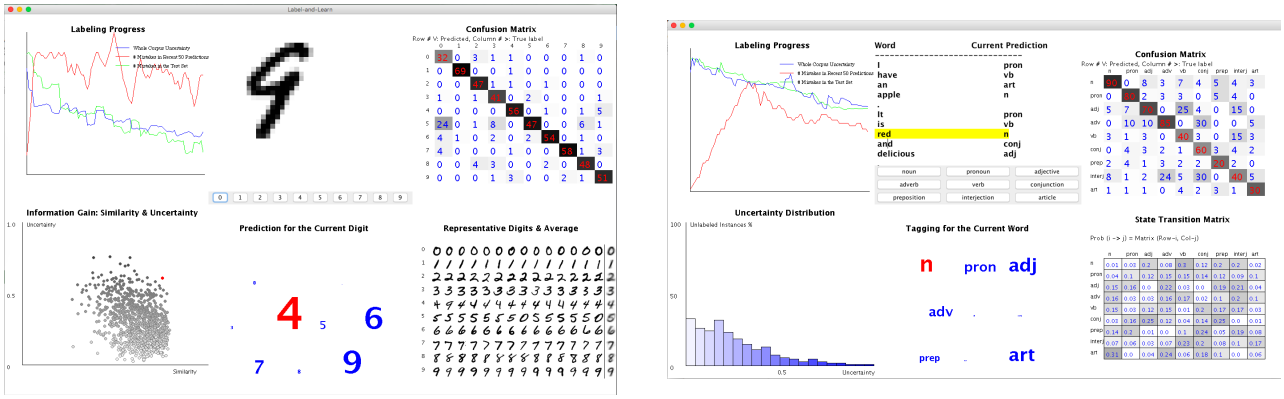

Figure 6: Two additional Label-and-Learn interfaces. To the left, a neural network interface applied to the recognition of off-line hand-drawn digits. To the right, a hidden Markov Model interface applied to part-of-speech tagging.

our five visualizations are broadly applicable; the other three make assumptions about the link between features, model, and classification that, in some cases, require revisiting if using different machine learning models.

To fully address issues of generic+algorithm specific visualizations, we have explored additional machine learning algorithms and tasks using our system. Figure 6 shows two additional Label-and-Learn interfaces that we have designed for additional machine learning algorithms and tasks. First, on the left, we see a Neural Network labeling interface designed to recognize scanned, hand-written digits from '0' to '9' [11]. The ultimate purpose for each of the five visualizations still remains the same: showing the labeling progress, the performance on the test set, information gain, prediction of the current data, and the reasoning behind the classifier. The two visualizations on the left are identical to the Named-Entity Recognition task shown in Figure 2. Rather than a two-category classification problem with meaningful words as features, the neural network classifier is being applied to a ten-category classification problem with each pixel as a feature. To adapt the change, the test set performance visualization on the top right corner, has been modified to a confusion matrix to visualize the ten-category classification result. The one below the labeling window still shows the current example prediction, but is modified as a word cloud to enhance the appearance of the most likely result. The visualization on the bottom right corner shows the classifier's current knowledge by displaying the most representative data classified as each category, and the pixel average of each category. Users can assume that the Neural Network classifier is well-trained when all the data cluster at the top-left-to-bottom-right diagonal of the confusion matrix, only one digit stands out in the word cloud, the most representative digits all belong to the category, and the pixel average shows a clear image.

On the right is an interface for hidden Markov Models (HMMs) applied to part-of-speech tagging. A state transition probability matrix is used to show the classifier's reasoning. The Information Gain visualization is also modified due to the absence of *Similarity*, as quantifying how "different" two inputs were did not make sense (Is the difference between a verb and a noun greater or less than the difference between a noun and a preposition?). Instead, we use a histogram to display the distribution of *Uncertainty* or entropy in the system. Inputting data as subsampled frequency spectrum for prosodic features might look similar to this interface. While these two interfaces have not been evaluated experimentally in the same way that our binary classifier has been evaluated, these two interfaces do demonstrate that the Label-and-Learn system can be applied to different machine learning tasks that use different algorithms, have more categories, and have larger feature space.

CONCLUSION

This paper presents our Label-and-Learn system, which is designed to help people without machine learning expertise understand the classifier's behavior while they perform the labeling task. The result of our user study shows the visualizations improve user experience during labeling and quantitatively help labelers understand the reasoning and performance of the current classifier. More importantly, the visualization allows users to predict the likelihood of success for the classifier so that they can avoid futile labeling. As a result, participants using the system reported that the previously tedious task of labeling was transformed such that participants felt more motivated to label and felt they were developing greater insight into the data. The success of the Label-and-Learn system is a useful addition to the suite of interactive machine learning and visualization tools being designed to improve the user's understanding and experience in machine learning tasks. It also promises to make data labeling a more integrated task in the design of machine learning classifiers.

ACKNOWLEDGEMENTS

Funding for this research was provided by the Natural Science and Engineering Research Council of Canada (NSERC Discovery Grant Program), Google Research via an Expedition Grant and the Faculty Fellowship Program, and the Ontario Ministry of Education Research Fund Program in Research Excellence (ORF-RE).

REFERENCES

1. S. Amershi, J. Fogarty, and D. Weld. 2012. Regroup: Interactive Machine Learning for On-demand Group

Creation in Social Networks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 21–30. DOI:http://dx.doi.org/10.1145/2207676.2207680

2. S. Amershi, B. Lee, A. Kapoor, R. Mahajan, and B. Christian. 2011. CueT: Human-guided Fast and Accurate Network Alarm Triage. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 157–166. DOI: http://dx.doi.org/10.1145/1978942.1978966
3. A.P. Bradley. 1997. The Use of the Area Under the ROC Curve in the Evaluation of Machine Learning Algorithms. *Pattern Recogn.* 30, 7 (July 1997), 1145–1159. DOI: http://dx.doi.org/10.1016/S0031-3203(96)00142-2
4. J.A. Fails and D.R. Olsen, Jr. 2003. Interactive Machine Learning. In *Proceedings of the 8th International Conference on Intelligent User Interfaces (IUI '03)*. ACM, New York, NY, USA, 39–45. DOI: http://dx.doi.org/10.1145/604045.604056
5. J. Fogarty, D. Tan, A. Kapoor, and S. Winder. 2008. CueFlik: Interactive Concept Learning in Image Search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*. ACM, New York, NY, USA, 29–38. DOI: http://dx.doi.org/10.1145/1357054.1357061
6. N. Gillian and J. A. Paradiso. 2014. The Gesture Recognition Toolkit. *J. Mach. Learn. Res.* 15, 1 (Jan. 2014), 3483–3487. http://dl.acm.org/citation.cfm?id=2627435.2697076
7. S.G. Hart and L.E. Staveland. 1988. Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. *Human mental workload* 1, 3 (1988), 139–183.
8. F. Heimerl, S. Koch, H. Bosch, and T. Ertl. 2012. Visual Classifier Training for Text Document Retrieval. *IEEE Trans. Vis. Comput. Graph.* 18, 12 (2012), 2839–2848. DOI:http://dx.doi.org/10.1109/TVCG.2012.277
9. T. Kulesza, S. Amershi, R. Caruana, D. Fisher, and D. Charles. 2014. Structured Labeling for Facilitating Concept Evolution in Machine Learning. In *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 3075–3084. DOI: http://dx.doi.org/10.1145/2556288.2557238
10. T. Kulesza, M. Burnett, W. Wong, and S. 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, New York, NY, USA, 126–137. DOI: http://dx.doi.org/10.1145/2678025.2701399
11. Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. 1998. Gradient-Based Learning Applied to Document Recognition. In *Proceedings of the IEEE*.
12. B.Y. Lim and A.K. Dey. 2010. Toolkit to Support Intelligibility in Context-aware Applications. In *Proceedings of the 12th ACM International Conference on Ubiquitous Computing (UbiComp '10)*. ACM, New York, NY, USA, 13–22. DOI: http://dx.doi.org/10.1145/1864349.1864353
13. D. Maynes-Aminzade, T. Winograd, and T. Igarashi. 2007. Eyepatch: Prototyping Camera-based Interaction Through Examples. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology (UIST '07)*. ACM, New York, NY, USA, 33–42. DOI:http://dx.doi.org/10.1145/1294211.1294219
14. M.A. Migut, M. Worring, and C.J. Veenman. 2015. Visualizing Multi-dimensional Decision Boundaries in 2D. *Data Min. Knowl. Discov.* 29, 1 (Jan. 2015), 273–295. DOI:http://dx.doi.org/10.1007/s10618-013-0342-x
15. T. Munzner. 2014. *Visualization Analysis and Design*. A K Peters. http://www.cs.ubc.ca/~tmm/vadbook/
16. D. Nadeau and S. Sekine. 2007. A survey of named entity recognition and classification. *Linguisticae Investigationes* 30, 1 (January 2007), 3–26. http://www.ingentaconnect.com/content/jbp/li/2007/00000030/00000001/art00002 Publisher: John Benjamins Publishing Company.
17. K. Patel. 2012. *Lowering the Barrier to Applying Machine Learning*. Ph.D. Dissertation. University of Washington.
18. K. Patel, N. Bancroft, S.M. Drucker, J. Fogarty, A.J. Ko, and J. Landay. 2010. Gestalt: Integrated Support for Implementation and Analysis in Machine Learning. In *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology (UIST '10)*. ACM, New York, NY, USA, 37–46. DOI: http://dx.doi.org/10.1145/1866029.1866038
19. K. Patel, S.M. Drucker, J. Fogarty, A. Kapoor, and D.S. Tan. 2011. Using Multiple Models to Understand Data. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Two (IJCAI'11)*. AAAI Press, 1723–1728. DOI:http://dx.doi.org/10.5591/978-1-57735-516-8/IJCAI11-289
20. C. Seifert, V. Sabol, and M. Granitzer. 2010. *Networked Digital Technologies: Second International Conference, NDT 2010, Prague, Czech Republic, July 7-9, 2010. Proceedings, Part I*. Springer Berlin Heidelberg, Berlin, Heidelberg, Chapter Classifier Hypothesis Generation Using Visual Analysis Methods, 98–111. DOI: http://dx.doi.org/10.1007/978-3-642-14292-5_11
21. B. Settles. 2012. *Active Learning*. Morgan and Claypool.
22. J. Talbot, B. Lee, A. Kapoor, and D.S. Tan. 2009. EnsembleMatrix: Interactive Visualization to Support Machine Learning with Multiple Classifiers. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. ACM, New York, NY, USA, 1283–1292. DOI: http://dx.doi.org/10.1145/1518701.1518895

23. J.O. Wobbrock, L. Findlater, D. Gergle, and J.J. Higgins. 2011. The Aligned Rank Transform for Nonparametric Factorial Analyses Using Only Anova Procedures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 143–146. DOI: http://dx.doi.org/10.1145/1978942.1978963