

Active Learning with Irrelevant Examples

Dominic Mazzoni, Kiri L. Wagstaff, and Michael Burl

Jet Propulsion Laboratory, California Institute of Technology,
Pasadena CA 91109, USA, kiri.wagstaff@jpl.nasa.gov

Abstract. Active learning algorithms attempt to accelerate the learning process by requesting labels for the most informative items first. In real-world problems, however, there may exist unlabeled items that are irrelevant to the user’s classification goals. Queries about these points slow down learning because they provide no information about the problem of interest. We have observed that when irrelevant items are present, active learning can perform worse than random selection, requiring more time (queries) to achieve the same level of accuracy. Therefore, we propose a novel approach, **Relevance Bias**, in which the active learner combines its default selection heuristic with the output of a simultaneously trained relevance classifier to favor items that are likely to be both informative and relevant. In our experiments on a real-world problem and two benchmark datasets, the Relevance Bias approach significantly improved the learning rate of three different active learning approaches.

1 Introduction

For many classification problems, class labels must be generated by a domain expert and are therefore expensive to acquire. Active learning [1] attempts to reduce this burden by incrementally selecting the most useful items for labeling. Current active learning approaches assume that the expert can provide a valid label for any item in the data set. However, there are cases in which an item does not fall into any of the classes of interest and therefore cannot be labeled. Existing active learning methods have no mechanism for handling this case.

To illustrate this phenomenon and its prevalence, consider the problem of training a webpage classifier based on the “four universities” data set [2]. Each webpage is classified as “student”, “faculty”, “staff”, “department”, “course”, “project”, or “other”. The “other” class is, by definition, irrelevant to any of the possible classification tasks. Similarly, consider the problem of handwritten digit recognition, which the United States Post Office has been using to aid in sorting mail since 1965. While we have several excellent labeled data sets to work with today, the problem of irrelevant examples will again arise for any new data collection. When scanning and extracting examples, it is likely that there will be several non-digit items recorded, such as marks, smudges, or non-digit characters. More generally, this problem will arise anytime the same data set is used to train classifiers that have different subsets of relevant items, like “user webpage” vs. “course webpage” or “company website” vs. “university website”.

An obvious strategy for dealing with data sets that contain irrelevant items is to filter the data before training the classifier, such as previous work that omits the “other” class in the webpage data set. While this is a reasonable solution for a benchmark data set, it is unrealistic as a general solution, especially with very large data sets. In collecting additional data in the webpage domain, for example, we will inevitably encounter additional irrelevant items, since the “other” class is by far the most frequently occurring in the labeled data set. Critically, we observe that the process of filtering is as labor-intensive as labeling the entire data set. Since the purpose of active learning is to achieve high performance without requiring that every item be labeled, it is necessary for active learners to be able to work with the unfiltered data. a

This paper seeks to address the limitations of current active learning methods when irrelevant items are present via two major contributions. First, we propose an active learning framework where “irrelevant” is a valid response from the expert labeler (Section 2). In this framework, we demonstrate that several popular active learning methods are sensitive to the presence of irrelevant examples. In fact, this situation can cause them to perform worse (that is, learn more slowly) than a random selection strategy. Given this surprising result, we require a means to accommodate data sets with irrelevant examples without suffering a loss in performance. Our second contribution is a method, **Relevance Bias**, by which any active learning method can learn to avoid querying irrelevant items (Section 3). We present experimental results in Section 4 that demonstrate the improvements achieved by active learners with a relevance bias. We compare this approach to two alternative approaches in Section 5 and conclude in Section 6.

2 Active Learning

We focus on *pool-based* active learning, where the learner has access to a (fixed) pool of items for which it can request labels. We assume the existence of a pool $\mathcal{U} = \{x_i\}$ of unlabeled items. Each x_i is a d -dimensional vector in Euclidean space, and the items are assumed to be i.i.d. according to an unknown fixed distribution $P(x)$. For simplicity, we will discuss active learning in the context of binary classification. In traditional active learning, there exists a classification label $y_i \in \{\pm 1\}$ for each x_i that is available, upon request, from the expert labeler. The expert can be a human or an automated labeler that incurs some cost per query; we refer to the expert’s labeling of x as $f(x)$. Let \mathcal{L} be the set of items for which the learner has already requested labels. In each round, the active learner selects an unlabeled item x from \mathcal{U} and receives its label, $y = f(x)$. The learner then applies its classifier learning algorithm to $\mathcal{L} \cup \{(x, y)\}$ to train a new model.

The performance of an active learning algorithm can be quantified with learning curves that show accuracy as a function of the number of rounds. It is also useful to compress the information in a learning curve into a single scalar value, such as area under the curve (AUC). The AUC of algorithm \mathcal{A} after n rounds is

the sum of the accuracy it obtained at each round from 1 to n , normalized by n .

$$\text{AUC}_n(\mathcal{A}) = \frac{1}{n} \sum_{t=1}^n \text{Acc}_t(\mathcal{A}) \quad (1)$$

If n is clear, we omit the subscript. In general, a higher AUC indicates faster, more efficient learning.

In this section, we describe several active learning methods and then discuss how the active learning problem changes when irrelevant items exist.

2.1 Active Learning Algorithms

Although active learning is not restricted to any single inductive learning technique, much of the recent work in this area has focused on active learning for support vector machines (SVMs) [3] due to their strong performance on a variety of problems. An SVM is a binary classifier that constructs a hyperplane in d dimensions to separate the two classes. In particular, it seeks the hyperplane that will maximize the *margin*, or distance between each class and the hyperplane. For classes that are not linearly separable, the SVM implicitly maps each point into a higher-dimensional space via a *kernel function*, which often improves separability. All active learning methods seek to select the item x which, when labeled (as y), provides the greatest accuracy improvement achieved by the model constructed from $\mathcal{L} \cup (x, y)$.

In this work, we consider four data selection methods, allowing us to compare three active learning methods with passive (random) selection:

1. **Simple Margin (“Simple”)** [4]: **Simple** assumes that items nearest the current hyperplane will be most useful in updating the classifier’s decision boundary. It ranks each example by its distance from the hyperplane (computed using the absolute value of the SVM output) and then chooses the item with the smallest distance.
2. **MaxMin Margin** [4]: Rather than guessing that the item closest to the hyperplane will yield the most information, this method empirically tests which item will be most effective, in terms of maximizing the separation between the two classes. Let m be the size of this separation (the margin) for a given SVM. Then, for each $x \in \mathcal{U}$, this method trains two SVMs: one on $\mathcal{L} \cup \{(x, +1)\}$ (yielding m^+) and one on $\mathcal{L} \cup \{(x, -1)\}$ (yielding m^-). Finally, **MaxMin** selects x such that $\min(m^-, m^+)$ is maximized. That is, it seeks the x that yields the largest margin, regardless of which label x receives.
3. **Diverse** [5]: This algorithm attempts to avoid choosing too many similar queries by increasing the diversity of the chosen examples. The diversity of $\mathcal{L} \cup x$ can be maximized by selecting x whose maximum normalized kernel distance to all of the other labeled examples is minimized. The complete method uses a weighted sum of diversity and hyperplane distance, controlled by a parameter λ , where $\lambda = 0$ is the equivalent of focusing solely on diversity and $\lambda = 1$ is the same as **Simple**. We determined that $\lambda = 0.5$ worked well for our experiments.

4. **Random:** This method selects $x \in \mathcal{U}$ randomly. It is equivalent to passive learning, since the algorithm has no control over the order of the items it sees. It serves as a baseline for comparison with active learning algorithms.

Probabilistic Active Learning. Because each of the active learning algorithms described above proceeds heuristically, it will not always be advantageous to select the top-ranked query. Therefore, for each algorithm, we instead used a variant that sorts all of the examples in the pool according to the active learning algorithm’s heuristic, and then chooses an item at random from the top $p\%$ of the pool instead of choosing the top-ranked example. In our experiments, with $p = 10\%$, we determined that these probabilistic active learners outperformed the “strict” algorithms by 1-6% on all three data sets and never resulted in decreased performance. Therefore, in the remainder of the paper we will report results using probabilistic active learning.

2.2 Active Learning with Irrelevant Items

Each of the existing active learning methods uses a different heuristic to select the next item for labeling, but they were all designed with the assumption that the label exists. Although active learners estimate the expected gain in information associated with obtaining a label for item x , they do not consider relevance as a factor. As we will show, this can have very negative results on learning speed.

Consider the problem of training a classifier to distinguish cloudy from clear pixels in satellite images. The Multi-angle Imaging SpectroRadiometer (MISR) instrument in Earth orbit captures several such images each day. A cloudy/clear pixel classifier provides an important service by enabling users to automatically identify regions of interest: atmospheric scientists can study clouds directly, and scientists interested in surface features can exclude the cloudy regions. However, in our experiments with this data set (described in more detail in Section 4.1), we discovered a large number of pixels that could not be classified as either type. These pixels, which were contaminated by a large dust plume, provide no useful information about how to distinguish cloudy pixels from clear pixels; they are irrelevant to the classification goal. Existing active learning methods cannot operate at all with irrelevant items, since they require that each queried item be labeled with one of the relevant classes. “None of the above” is not a valid label response.

Applying active learners when irrelevant items are present requires a modified learning framework. We model the new expert labeler as a function h that maps items to *three* possible values, $y_i \in \{-1, 0, +1\}$. A value of 0 indicates that the label is irrelevant to the learning task at hand. Again, on each round, the active learner applies a selection function to choose an unlabeled item x from \mathcal{U} . The learner proceeds normally unless $h(x) = 0$, in which case it acquires no new information and must wait until the following round to make a new request. For some problems, waiting until the next round can be extremely expensive. For example, we have investigated using active learning to select initial conditions

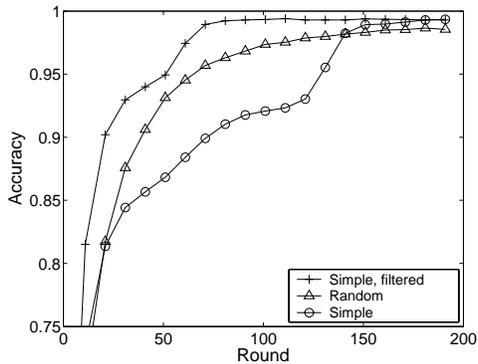


Fig. 1. Accuracy achieved by active learning with random item selection vs. the Simple active learner on the MISR cloudy/clear pixel classification task. Results are averaged over 100 trials. To make all of the figures easier to read, we average each 10 successive rounds into a single point.

for an asteroid collision simulator [6]. Determining the “label” (outcome) associated with one set of initial conditions requires running a numerical simulation algorithm for days. In such a case, a significant amount of time is lost due to an irrelevant query.

Given this framework, we are able to proceed experimentally with the MISR data set. We found that the irrelevant pixels had a very adverse effect on active learning. As shown in Figure 1, the Simple active learner performs worse than random selection. This behavior is surprising, as it contradicts a large body of previous work demonstrating that Simple outperforms random selection on a variety of problems. Indeed, when restricted to a filtered version of the data set that excludes the dusty pixels (“Simple, filtered”), the algorithm does outperform random selection. Without filtering, the regular Simple algorithm devotes an inordinate number of queries (more than would be expected by random chance) to the irrelevant class. Because no information is gained from a query to a dusty pixel, Simple’s learning is severely delayed. The increase in performance after about round 120 is due to the fact that the irrelevant items in \mathcal{U} have been exhausted and Simple is forced to select relevant items. Its default selection bias, we observe, is a weakness in this situation.

3 Solution: Active Learning Relevance Bias

We propose an active learner that, in addition to selecting the most informative items for labeling, also learns to separate relevant from irrelevant items. This approach can be adopted by any existing active learning method. The new active learner collects irrelevant items x in a set \mathcal{R} of rejected queries, then trains an additional classifier to distinguish between the set of examples in \mathcal{L} (the labeled set, both positive and negative examples) and the examples in \mathcal{R} . The active learner uses this relevance classifier, C^* , to influence its decision about which

<p>SELECT-LB(active learner \mathcal{A}, relevance classifier C^*)</p> <ol style="list-style-type: none"> 1. Rank all items x in the unlabeled pool \mathcal{U} using \mathcal{A}. 2. Normalize all $\mathcal{A}(x)$ scores into the range $[0, 1]$. 3. Calculate $P(\text{relevant} x)$ using C^*. 4. Select x such that $\mathcal{A}(x) \times P(\text{relevant} x)$ is maximized, and request $y = h(x)$, the label for x. 5. If $y = 0$ (irrelevant), add x to \mathcal{R}; otherwise, add x to \mathcal{L}. Re-train C^*.
--

Fig. 2. Pseudo-code for adding a Relevance Bias to active learner \mathcal{A} .

example should be chosen next. Classifier C^* is unlikely to be 100% accurate, especially in early rounds, so relying on it to strictly filter the pool \mathcal{U} may not yield the best results. Instead, we use C^* to influence the active learner’s *rankings* of items in the pool, creating a **Relevance Bias (RB)**. Figure 2 outlines pseudo-code that replaces a normal active learner’s item selection method. In step 4, the learner combines its ranking of the items with the probability that they are relevant to yield a final decision about which item to query. In our experiments, using C^* as a modifier rather than a filter increased performance by up to 30%.

Initially, this solution sounds similar to a multi-class active learning approach, in which each item is assigned to the positive, negative, or irrelevant class. As we will show in Section 5, however, the multi-class approach has much lower performance than the Relevance Bias method. The approach we propose, which mathematically combines item ranking with the probability of relevance, consistently yields superior results.

In our experiments, we trained an SVM for C^* using the same parameters (kernel function and regularization parameter) as were used for \mathcal{A} . However, any learning method that outputs a probability can be used. When C^* is an SVM, $P(\text{relevant}|x)$ can be approximated in several ways, such as clipping values outside the range $[-1, 1]$ and mapping them to the range $[0, 1]$, or using Platt’s technique of mapping the SVM outputs to a sigmoid probability model [8]. We used the former method in this work.

4 Experimental Results

To evaluate the effectiveness of our proposed approach, we conducted experiments on a real-world data set and two benchmark problems. In this section, we first introduce the data sets we used and then present experimental results for all three active learners. Key data set characteristics, including SVM parameters, are shown in Table 1.

4.1 Data Sets

MISR. This data set consists of the pixels in an image that was collected by the MISR instrument over the Sahara Desert on February 6, 2004. With the help of an atmospheric scientist, we identified three regions in the image corresponding

Table 1. Summary of the data sets, including the SVM parameters used, the choice of positive, negative, and irrelevant classes, and number of items in each class. All experiments used an RBF kernel.

Data set	Number of features	Training set (# items)			SVM params	
		Positive	Negative	Irrelevant	γ	C
MISR	156	cloudy (100)	clear (100)	dusty (100)	1.0	1.0
DNA	180	EI (50)	IE (50)	neither (50)	2^{-6}	8.0
MNIST	784	‘1’ (100)	‘7’ (100)	others (800)	1.0	1.0

to clouds, clear land, and dust. For each pixel, we extracted 156 features: the bidirectional reflectance factor for the pixel and a subset of the pixels within a 5x5 neighborhood, from four different spectral bands and from cameras viewing the scenes from three different angles. We selected 300 pixels randomly from the image to form the training set, 100 each of the cloudy, clear, and dusty classes.

DNA. The second data set is the `dna` data set from the StatLog repository [9]. Our task is to use the DNA sequence on either side of a splice junction to distinguish between two classes, corresponding to exon/intron boundaries (EI sites, or “donors”) and intron/exon boundaries (IE sites, or “acceptors”). Some splice junctions fall into neither category and therefore present us with an irrelevant class. Each feature vector consists of 180 features (60 DNA base pairs, each encoded using three binary features). Each experiment was run on a pool of 150 examples chosen randomly from the training set of 2000 examples and tested on all relevant examples from the test set. We trained our SVMs using the same parameters as those used in the one-vs-one experiments by Hsu and Lin [10].

MNIST. The MNIST data set consists of scanned images of handwritten digits 0 through 9. Each image is composed of 28x28 pixels, which are the features for that image. The goal is to learn to distinguish between different digits, allowing for the wide variation that occurs naturally in human handwriting. We used a subset of the full data set that contained 1000 items, 100 from each class (digit). For these experiments, we focus on learning to distinguish between digits 1 and 7, which is one of the more difficult cases. The 800 items representing the eight other digits are all irrelevant items, since they are neither 1’s nor 7’s.

4.2 Active Learning Results

To test our claim that adding a Relevance Bias improves the performance of active learners, we performed an empirical comparison for all three active learning methods against Random selection and the RB-enhanced versions of each method. Figure 3 shows the behavior of Simple, RB-Simple, and Random on each data set. The quantified AUCs observed for all three active learners are shown in Table 2. As before, Random outperforms Simple on the MISR data set. RB-Simple improves on Simple’s performance, yielding an AUC that exceeds that of Random. Simple is less affected by the irrelevant items in the DNA

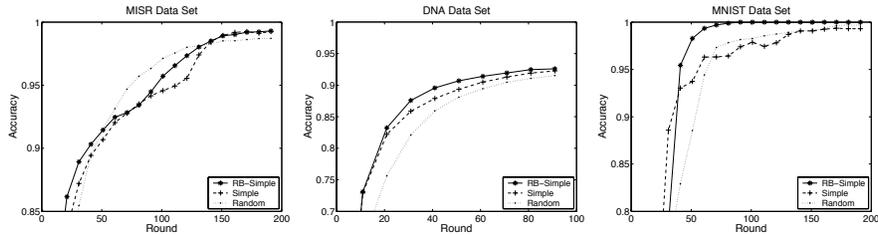


Fig. 3. Simple active learning with and without a Relevance Bias (100 trials).

data set, and it outperforms Random. However, adding the Relevance Bias results in even better performance. Simple marginally outperforms Random on the MNIST data set. Although the Relevance Bias changes the shape of the learning curve, the overall AUC does not change. The MaxMin algorithm is even more sensitive to the presence of irrelevant items. While its performance on the DNA data set is not much lower than that of Random, it suffers major decreases in performance for MISR and MNIST. The Diverse algorithm obtains results more similar to Simple’s. Overall, we see that the RB-enhanced version of each active learner is equivalent to or better than the original method.

Why Does Regular Active Learning Fail? We have observed that the presence of irrelevant items can negatively impact an active learner’s performance. We claim that this is due to the active learner’s repeated selection of items that are irrelevant, which delays learning. Otherwise powerful active learning heuristics such as Simple are particularly vulnerable to this effect when the irrelevant items fall near the decision boundary. Simple’s bias is well motivated, but it needs to be applied selectively.

To more clearly illustrate this behavior, Figure 4 (left) shows the (cumulative) number of irrelevant items that each algorithm selected. We can see that Simple has a definite bias towards selecting the irrelevant items (more than would be expected by random chance). In contrast, RB-Simple chooses far fewer irrelevant items (usually fewer than Random), and correspondingly demonstrates improved performance.

Table 2. AUC for three active learners compared to Random and RB-enhanced versions, over three data sets (100 trials each). In each case, the best result is in bold. Note that AUC for DNA is calculated at 100 rounds (instead of 200), due to its smaller training set size.

Data set	Random	Simple		MaxMin		Diverse	
		Regular	RB	Regular	RB	Regular	RB
MISR	91.2	90.0	92.0	63.0	72.4	91.0	91.2
DNA	79.3	81.7	82.4	78.1	78.8	81.7	82.8
MNIST	87.1	88.0	88.0	54.0	65.0	89.4	90.5

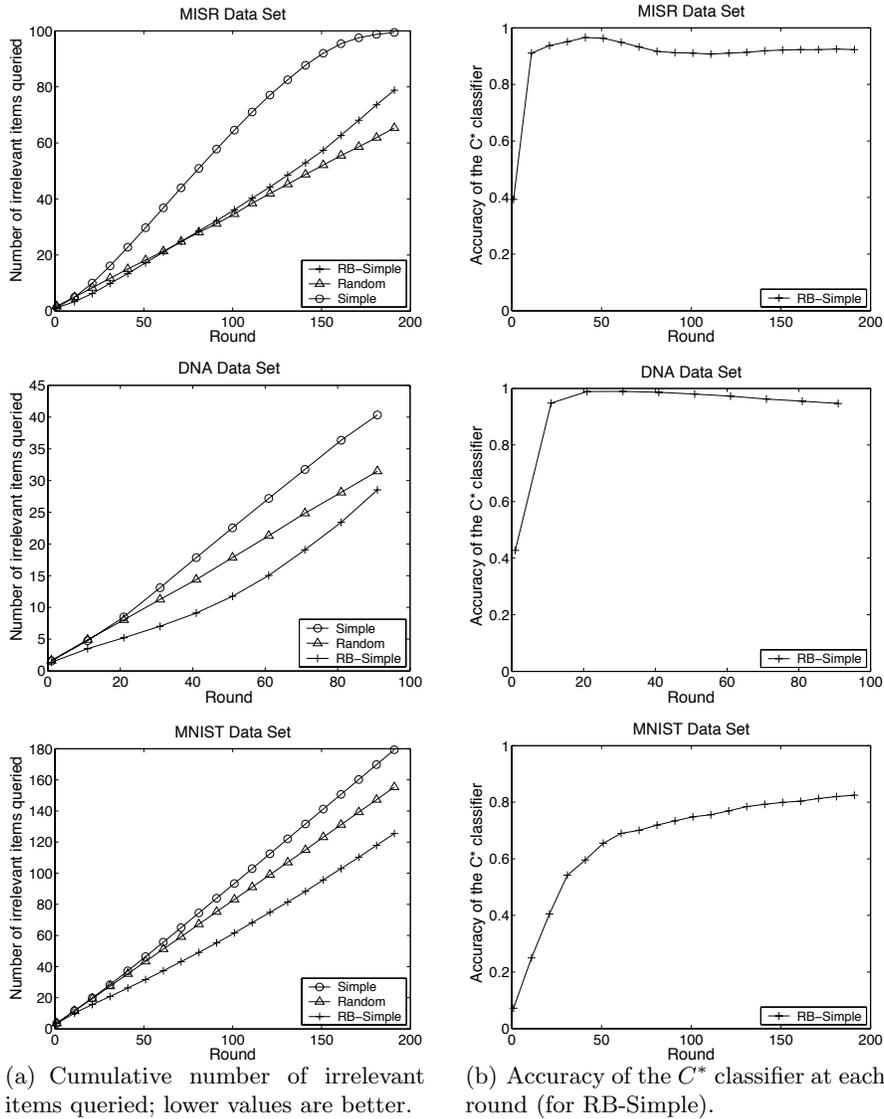


Fig. 4. Comparison between Simple, Random, and RB-Simple in terms of number of irrelevant examples queried. In each case, Simple queries more irrelevant examples than would be expected by random chance. The right column shows the evolving accuracy of RB-Simple’s C^* (relevance) classifier. All results are averaged over 100 trials.

Table 3. Assessment of two alternatives to Relevance Bias: assigning fake labels, or using a multi-class active learner (average AUC over 100 trials). The best value for each data set is in bold.

Data set	Random Simple		Simple; irrelevant items assigned:			Multi-class	RB-Simple
			Positive	Negative	Randomly		
MISR	90.7	90.1	90.9	90.8	90.7	86.8	91.6
DNA	78.5	82.0	80.7	77.6	78.3	73.5	85.5
MNIST	86.2	87.9	84.0	90.1	87.0	73.2	91.9

Because the C^* classifier for RB-Simple is also engaged in a learning task, we can plot its learning curve (see Figure 4 (right)); evaluation is over a separate set of items not included in \mathcal{U}). After each round, C^* adds another item to its training set, either a member of \mathcal{L} (relevant, labeled) or \mathcal{R} (irrelevant, rejected). We observe that, for the MISR and DNA data sets, C^* quickly achieves a high level of performance. However, C^* learns much more slowly on the MNIST problem, confirming our previous hypothesis that this is a harder classification task.

5 Alternatives to Relevance Bias

We have suggested one solution to the problem of trying to learn in the presence of irrelevant examples: adding a Relevance Bias to the active learner. There are two other approaches that could be used with no changes to existing active learning algorithms. Here we introduce both options and explain why they do not perform as well as the Relevance Bias approach.

5.1 Label the Irrelevant Examples

Given a situation in which some items cannot be labeled, we can simply assign them fake labels and proceed with regular active learning. Naturally, these labels will be noisy, but if, for example, the impact on generalization performance were negligible, then we would be able to quickly dispatch the problem of irrelevant examples using existing techniques.

However, we find that the noisy labels do introduce significant additional errors. We created three new versions of each data set, assigning the irrelevant items either a) to the positive class, b) to the negative class, or c) randomly to one of the two classes. Table 3 shows the AUC values obtained by Simple with each of these modified label sets. For comparison, we include the AUC values for Random, Simple, and RB-Simple; these algorithms used the original label sets (with irrelevant items present). Surprisingly, the fake labels can sometimes boost Simple’s performance higher than if it received no information at all from the irrelevant items (e.g., MNIST with irrelevant items assigned to the negative class does better than Simple). However, in each case we see superior performance when using the Relevance Bias to accommodate irrelevant examples.

5.2 Multi-class Active Learning

Another intuitive approach would be to use a multi-class active learner and provide it with three classes: the positive, negative, and irrelevant items. If this approach is successful, then we do not need a new framework for handling irrelevant items; current methods will suffice.

We tested this approach for each of our data sets. In each case, we trained a multi-class active learner [11] to discriminate between these three classes. As shown in Table 3, the multi-class active learner also performs worse (learns more slowly) than Random, because it aims to simultaneously maximize performance over all three classes and must therefore devote even more queries to the third (irrelevant) class. In contrast, the Relevance Bias approach can provide performance gains even when C^* is not yet fully accurate, because the majority of its effort is devoted to modeling the two critical classes. This intuition is confirmed experimentally: we find that RB-Simple strongly outperforms the multi-class learner, by 5–19%.

6 Conclusions and Future Work

Active learning enables the application of machine learning methods to problems where it is difficult or expensive to acquire expert labels. A key barrier to the use of current active learning methods is that the expert labeler will not always be able to assign a label to every example. Real-world data sets may contain items that are irrelevant to the user’s classification goals. When filtering is not a realistic option, it is essential that active learning methods be able to cope with the presence of irrelevant items. However, existing active learning algorithms can perform worse than passive learning in this situation.

We have proposed a new active learning framework that allows for any item to be assigned to an “irrelevant” class. We presented a novel method, Relevance Bias, by which any active learning algorithm can be modified to avoid irrelevant examples by training a second classifier to distinguish between the relevant and irrelevant items. This method consistently improves the performance of active learners when irrelevant items are present. We have also shown that some potential alternatives, such as assigning fake labels to the irrelevant items or using a multi-class active learner, do not perform as well as the Relevance Bias approach.

An important extension to this work will be to add a mechanism that compensates for the fact that C^* cannot perfectly distinguish relevant from irrelevant items (especially early on). For example, the algorithm could estimate C^* ’s accuracy via leave-one-out cross-validation and adjust the strength of the relevance bias over time. Thus, initial rounds will be more exploratory, while later ones can rely more strongly on C^* ’s recommendations.

Acknowledgments

The authors wish to thank Dennis DeCoste for suggesting the initial idea that led to this work and for valuable suggestions and feedback along the way. We also

thank David Diner, Roger Davies, and Michael Garay for motivating this work by working with us on MISR cloud classification, and Rebecca Castaño and Robert Granat for other valuable feedback. The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

References

1. Cohn, D., Atlas, L., Ladner, R.: Improving generalization with active learning. *Machine Learning* **15** (1994) 201–221
2. Craven, M., DiPasquo, D., Freitag, D., McCallum, A., Mitchell, T., Nigam, K., Slattery, S.: Learning to extract symbolic knowledge from the World Wide Web. In: *Proceedings of the Fifteenth National Conference on Artificial Intelligence*. (1998) 509–516
3. Cortes, C., Vapnik, V.: Support-vector network. *Machine Learning* **20** (1995) 273–297
4. Tong, S., Koller, D.: Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research* **2** (2002) 45–66
5. Brinker, K.: Incorporating diversity in active learning with support vector machines. In: *Proceedings of the Twentieth International Conference on Machine Learning*, Washington, D. C. (2003) 59–66
6. Burl, M.C., DeCoste, D., Enke, B., Mazzoni, D., Merline, W.J., Scharenbroich, L.: Automated knowledge discovery from simulators. In: *Proceedings of the Sixth SIAM International Conference on Data Mining*. (2006)
7. Greiner, R., Grove, A., Roth, D.: Learning cost-sensitive active classifiers. *Artificial Intelligence* **139** (2002) 137–174
8. Platt, J.C.: Probabilities for SV Machines. In Smola, A.J., Bartlett, P., Schölkopf, B., Schuurmans, D., eds.: *Advances in Large Margin Classifiers*, MIT Press (1999) 61–74
9. Michie, D., Spiegelhalter, D., Taylor, C.: *Machine Learning, Neural and Statistical Classification*. Prentice Hall, Englewood Cliffs, N.J. (1994) Data available at <http://www.liacc.up.pt/ML/statlog/>.
10. Hsu, C., Lin, C.: A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks* **13** (2002) 415–425
11. Tong, S.: *Active Learning: Theory and Applications*. PhD thesis, Stanford University (2001)