Active Learning Survey

Article · June 2013

CITATION
1

READS
1,532

2 authors, including:

Ahmed Dawod
Mansoura University
1 PUBLICATION **1** CITATION

SEE PROFILE

Active Learning Survey

Prepared By: Ahmed Dawod

I. Abstract

Machine learning is the study of techniques and algorithms used to enable the computer to learn on his own. These algorithms are divided into Supervised and Unsupervised categories. A third category combines the previous two that is Active Learning. In this proposal, I discuss what Active Learning is and what are the current trends and algorithms used in it. Then I discuss my point of view and future work about these algorithms.

II. Introduction

In the introduction, we will start by explaining some major definitions in the field like Supervised, unsupervised and Machine learning in general.

1. Machine Learning

Machine learning, a branch of artificial intelligence, is about the construction and study of systems that can learn from data. For example, a machine learning system could be trained on email messages to learn to distinguish between spam and non-spam messages. After learning, it can then be used to classify new email messages into spam and non-spam folders.

The core of machine learning deals with representation and generalization. Representation of data instances and functions evaluated on these instances are part of all machine learning systems. Generalization is the property that the system will perform well on unseen data

instances; the conditions under which this can be guaranteed are a key object of study in the subfield of computational learning theory.

There is a wide variety of machine learning tasks and successful applications. Optical character recognition, in which printed characters are recognized automatically based on previous examples, is a classic example of machine learning.

In 1959, Arthur Samuel defined machine learning as a "Field of study that gives computers the ability to learn without being explicitly programmed".

Tom M. Mitchell provided a widely quoted, more formal definition: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E". This definition is notable for its defining machine learning in fundamentally operational rather than cognitive terms, thus following Alan Turing's proposal in Turing's paper "Computing Machinery and Intelligence" that the question "Can machines think?" be replaced with the question "Can machines do what we (as thinking entities) can do?"

Machine learning algorithms can be organized into a taxonomy based on the desired outcome of the algorithm or the type of input available during training the machine.

Supervised learning generates a function that maps inputs to desired outputs (also called labels, because they

are often provided by human experts labeling the training examples). For example, in a classification problem, the learner approximates a function mapping a vector into classes by looking at input-output examples of the function.

1- Unsupervised learning models a set of inputs, like clustering. See also data mining and knowledge discovery. Here, labels are not known during training.
2- Semi-supervised learning combines both labeled and unlabeled examples to generate an appropriate function or classifier. Transduction, or transductive inference, tries to predict new outputs on specific and fixed (test) cases from observed, specific (training) cases.
3- Reinforcement learning learns how to act given an observation of the world. Every action has some impact in the environment, and the environment provides feedback in the form of rewards that guides the learning algorithm.
4- Learning to learn learns its own inductive bias based on previous experience.
5- Developmental learning, elaborated for Robot learning, generates its own sequences (also called curriculum) of learning situations to cumulatively acquire repertoires of novel skills through autonomous self-exploration and social interaction with human teachers, and using guidance mechanisms such as active learning, maturation, motor synergies, and imitation.

Actually, we are going to concentrate on the first three types (Supervised, Unsupervised and Semi-Supervised Learning).

2. Supervised Learning

Supervised learning is the machine learning task of inferring a function from labeled training data. The training data consist of a set of training examples. In supervised learning, each example is a pair consisting of an input object (typically a vector) and a desired output value (called the supervisory signal). A supervised learning algorithm analyzes the training data and produces an inferred function, which can be used for mapping new examples. An optimal scenario will allow the algorithm to correctly determine the class labels for unseen instances. This requires the learning algorithm to generalize from the training data to unseen situations in a "reasonable" way (see inductive bias).

The parallel task in human and animal psychology is often referred to as concept learning.

3. Unsupervised Learning

In machine learning, unsupervised learning refers to the problem of trying to find hidden structure in unlabeled data. Since the examples given to the learner are unlabeled, there is no error or reward signal to evaluate a potential solution. This distinguishes unsupervised learning from supervised learning and reinforcement learning.

Unsupervised learning is closely related to the problem of density estimation in statistics. However unsupervised learning also encompasses many other techniques that seek to summarize and explain key features of the data. Many methods employed in unsupervised learning are based on data mining methods used to preprocess data.

4. Semi-Supervised Learning

Semi-supervised learning is a class of machine learning techniques that make use of both labeled and unlabeled data for training - typically a small amount of labeled data with a large amount of unlabeled data. Semi-supervised learning falls between unsupervised learning (without any labeled training data) and supervised learning (with completely labeled training data). Many machine-learning researchers have found that unlabeled data, when used in conjunction with a small amount of labeled data, can produce considerable improvement in learning accuracy. The acquisition of labeled data for a learning problem often requires a skilled human agent (e.g. to transcribe an audio segment) or a physical experiment (e.g. determining the 3D structure of a protein or determining whether there is oil at a particular location). The cost associated with the labeling process thus may render a fully labeled training set infeasible, whereas acquisition of unlabeled data is relatively inexpensive. In such situations, semi-supervised learning can be of great practical value. Semi-supervised learning is also of theoretical interest in machine learning and as a model for human learning.

5. Active Learning

Now to our main concern in this survey. Active learning is a special case of semi-supervised machine learning in which a learning algorithm is able to interactively query the user (or some other information source) to obtain the desired outputs at new data points. In statistics, literature it is sometimes also called optimal experimental design.

6. Proactive Learning

Proactive learning is a generalization of active learning designed to relax unrealistic assumptions and thereby reach practical applications.

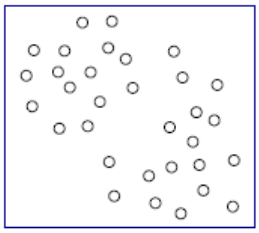
Active learning seeks to select the most informative unlabeled instances and ask an omniscient oracle for their labels, to retrain a learning algorithm maximizing accuracy. However, the oracle is assumed to be infallible (never wrong), indefatigable (always answers), individual (only one oracle), and insensitive to costs (always free or always charges the same). Proactive learning relaxes all four of these assumptions, relying on a decision-theoretic approach to jointly select the optimal oracle and instance, by casting the problem as a utility optimization problem subject to a budget constraint.

III. Active Learning

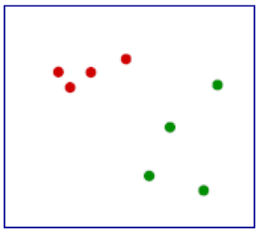
In the introduction part, we mentioned a brief definition for Active Learning.

There are situations in which unlabeled data is abundant but manually labeling it is expensive. In such a scenario, learning algorithms can actively query the user/teacher for labels. This type of iterative supervised learning is called active learning. Since the learner chooses the examples, the number of examples to learn a concept can often be much lower than the number required in normal supervised learning. With this approach, there is a risk that the algorithm be overwhelmed by uninformative examples.

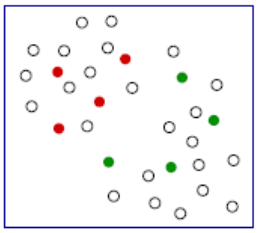
The concept of Active Learning actually comes from the "Active Learning" education technique that refers to several models of instruction that focus the responsibility of learning on learners.



Unlabeled points



Supervised learning



Semisupervised and active learning

- ## Active Learning Example (Drug Design):

I will quote here the example that was mentioned in “A tutorial on Active Learning” by Dasgupta and Langford.

The goal of the problem is to find compounds that bind to a particular target.

Therefore, we have a large collection of compounds collected from vendor catalogs, corporate collections and combinatorial chemistry.
We have the description of chemical compound (the unlabeled point) which we want to decide if it is active (binds to a target chemical) or inactive.
In this example, if we wanted to create a supervised training set it will cost very much money and time because each labeling process is done through a chemical experiment. Here comes the Active Learning part. We use as small set as possible of labeled data and we let the algorithm do the rest and query the user only for important unlabeled points that will help to make the classifier more accurate.

Now we will go into the algorithms used in Active Learning.

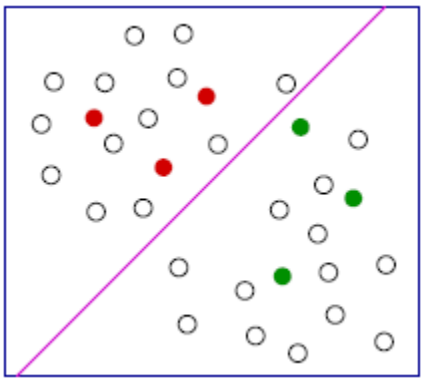
- Active Learning Algorithms:

During my research, I found some good algorithms and tools used in Active Learning and we will discuss them here.

However, before we look into the algorithms we will look at a general heuristic for Active Learning mentioned by Dasgupta and Langford, which is called “Biased Sampling”.

Start with a pool of unlabeled data
Pick a few points at random and get their labels
Repeat
    Fit a classifier to the labels seen so far
    Query the unlabeled point that is closest to the boundary (or most uncertain, or most likely to decrease overall uncertainty...)

This simple approach divides the space like this



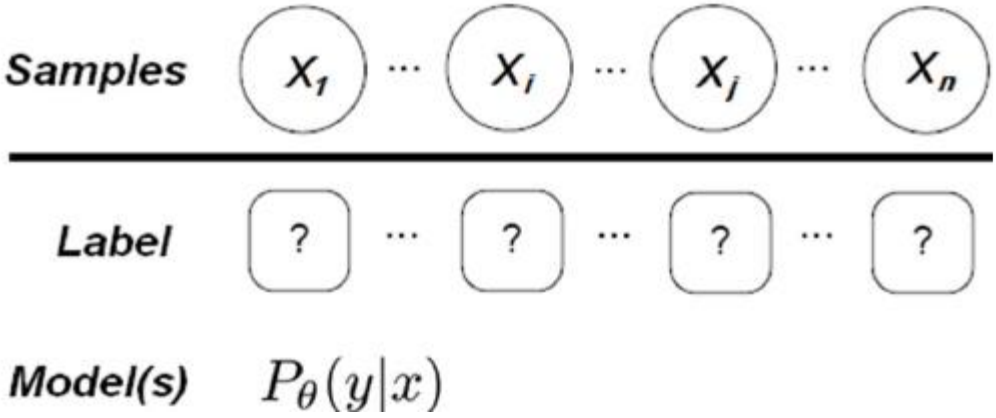
Now to the algorithms. Of course, there are many algorithms out there for Active Learning like A^2, QBC, and DHM, DC… Etc.
However, as this is a proposal for a continuing work in Active Learning and Machine Learning in General I picked three algorithms to discuss Uncertainty Sampling, Version Space Reduction, Query By Committee and Expected Error Reduction.

1- Uncertainty Sampling

To know when we can use uncertainty sampling we should state the three scenarios of active learning:

- Query Synthesis
  Learner constructs examples for labeling.
- Selective Sampling
  Unlabeled data come as a stream then for each arrived point, learner decides to query or discard
- Pool-based Active Learning
  Given a pool of unlabeled data learner chooses from the pool for labeling.

In Uncertainty Sampling we work with the third approach (Pool-based).



The simple idea behind uncertainty sampling is to query the sample x that the learner is most uncertain about.
However, how can we measure uncertainty?
Actually, Burr Settles quotes some methods to measure Uncertainty:

1- Maximum Entropy:
Dagan and Engelson suggested this method and it is represented as:

$$\emptyset_{ENT}(x) = -\sum_{y} P_\theta(y|x) \log_2 P_\theta(y|x)$$

2- Smallest Margin:
Suggested by Scheffer.

$$\emptyset_M(x) = P_\theta({y_1}^*|x) - P_\theta({y_2}^*|x)$$

3- Least Confident:
Suggested by Culotta and McCallum

$$\emptyset_{LC}(x) = 1 - P_\theta(y^*|x)$$

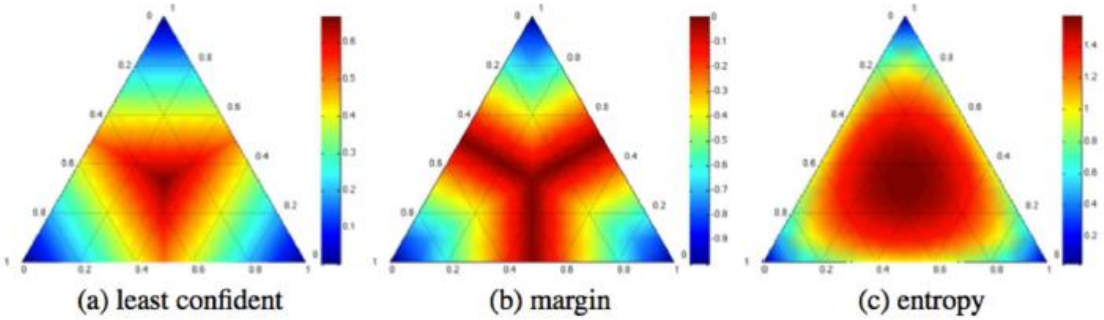


illustration of preferred (dark red) posterior distributions in a 3-label classification task

A final notice: all these methods are equivalent when dealing with binary tasks not multi-class tasks.

2- Query By Committee (QBC):

Query by Committee (QBC) is a very effective active learning approach that has been successfully applied to different classification problems (McCallum & Nigam, 1998; Dagan & Engelson, 1995; Liere & Tadepalli, 1997). A generalized outline of the QBC approach is presented in Algorithm 1. Given a pool of unlabeled examples, QBC iteratively selects examples to be labeled for training. In each iteration, it generates a committee of classifiers based on the current training set. Then it evaluates the potential utility of each example in the unlabeled set, and selects a subset of examples with the highest expected utility. The labels for these examples are acquired and they are transferred to the training set. Typically, the utility of an example is determined by some measure of disagreement in the committee about its predicted label. This process is repeated until the number of available requests for labels is exhausted.

Freund et al. (1997) showed that under certain assumptions, Query by Committee can achieve an exponential decrease in the number of examples required to attain a particular level of accuracy, as compared to random sampling. However, these theoretical results assume that Gibbs algorithm is used to generate the committee of hypotheses used for sample selection.

The Gibbs algorithm for most interesting problems is computationally intractable. To tackle this issue, Abe and Mamitsuka (1998) proposed two variants of QBC, Query by Bagging and Query by Boosting, where Bagging and AdaBoost are used to construct the

committees for sample selection. In their approach, they evaluate the utility of candidate examples based on the margin of the example; where the margin is defined as the difference between the number of votes in the current committee for the most popular class label, and that for the second most popular label. Examples with smaller margins are considered to have higher utility.

Algorithm 1 Generalized Query by Committee:

Given:
T - set of training examples
U - set of unlabeled training examples
BaseLearn - base learning algorithm
k - number of selective sampling iterations
m - size of each sample

1. Repeat k times
    2. Generate a committee of classifiers,
        C* = EnsembleMethod(BaseLearn, T)
    3. for all $x_j \in U$, compute Utility(C^*, x_j), based on the current committee
    4. Select a subset S of m examples that maximizes utility
    5. Label examples in S
    6. Remove examples in S from U and add to T
7. Return EnsembleMethod(BaseLearn, T)

Therefore, it is that simple, Keep a committee of classifiers and Query the instance that the committee members disagree.

3- Expected Error Reduction

In this approach, we try to minimize the risk R(x) of a query candidate using the function:

$$R(x) = \sum_{u \in U} E_y\left[H_{\theta+<x,y>}(Y|u)\right]$$

Where ∑ is the sum over unlabeled instances, Ey is the expectation over possible labeling of x and H is the uncertainty of u after retraining with x.

Roy and McCallum (2001) first proposed the expected error reduction framework for text classification using naïve Bayes. Zhu et al. (2003) combined this framework with a semi-supervised learning approach (Section 7.1), resulting in a dramatic improvement over random or uncertainty sampling. Guo and Greiner (2007) employ an "optimistic" variant that biases the expectation toward the most likely label for computational convenience, using uncertainty sampling as a fallback strategy when the oracle provides an unexpected labeling. This framework has the dual advantage of being near optimal and not being dependent on the model class. All that is required is an appropriate objective function and a way to estimate posterior label probabilities. For example, strategies in this framework have been successfully used with a variety of models including naïve Bayes (Roy and McCallum, 2001), Gaussian random fields (Zhu et al., 2003), logistic regression (Guo and Greiner, 2007), and support vector machines (Moskovitch et al., 2007). In theory, the general approach can be employed not only to minimize loss functions, but also to optimize any generic performance measure of interest, such as

maximizing precision, recall, F1-measure, or area under the ROC curve.

In most cases, unfortunately, expected error reduction is also the most computationally expensive query framework. Not only does it require estimating the expected future error over U for each query, but also a new model must be incrementally re-trained for each possible query labeling, which in turn iterates over the entire pool. This leads to a drastic increase in computational cost. For nonparametric model classes such as Gaussian random fields (Zhu et al., 2003), the incremental training procedure is efficient and exact, making this approach fairly practical1. For a many other model classes, this is not the case. For example, a binary logistic regression model would require O(ULG) time complexity simply to choose the next query, where U is the size of the unlabeled pool U, L is the size of the current training set L, and G is the number of gradient computations required by the by optimization procedure until convergence. A classification task with three or more labels using a MaxEnt model (Berger et al., 1996) would require $O(M^2ULG)$ time complexity, where M is the number of class labels. For a sequence labeling task using CRFs, the complexity explodes to $O(TM^{T+2}ULG)$, where T is the length of an input sequence. Because of this, the applications of the expected error reduction framework have mostly only considered simple binary classification tasks. Moreover, because the approach is often still impractical, researchers must resort to Monte Carlo sampling from the pool (Roy and McCallum, 2001) to reduce the U term in the previous analysis, or use approximate

training techniques (Guo and Greiner, 2007) to reduce the G term.

Now, we surveyed three of the most important algorithms in Active Learning. Of course, there are others as I said before like A^2, Cost Sensitive AL, Batch Mode AL, Multi Task AL, Variance Reduction, Expected Model Change … etc.

Some of these algorithms uses what is called SVM (Support Vector Machine) so we need to define it.

- *SVM:*

  In machine learning, support vector machines (SVMs, also support vector networks) are supervised learning models with associated learning algorithms that analyze data and recognize patterns, used for classification and regression analysis. The basic SVM takes a set of input data and predicts, for each given input, which of two possible classes forms the output, making it a non-probabilistic binary linear classifier. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples into one category or the other. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on.

In addition to performing linear classification, SVMs can efficiently perform non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

- Active Learning Applications

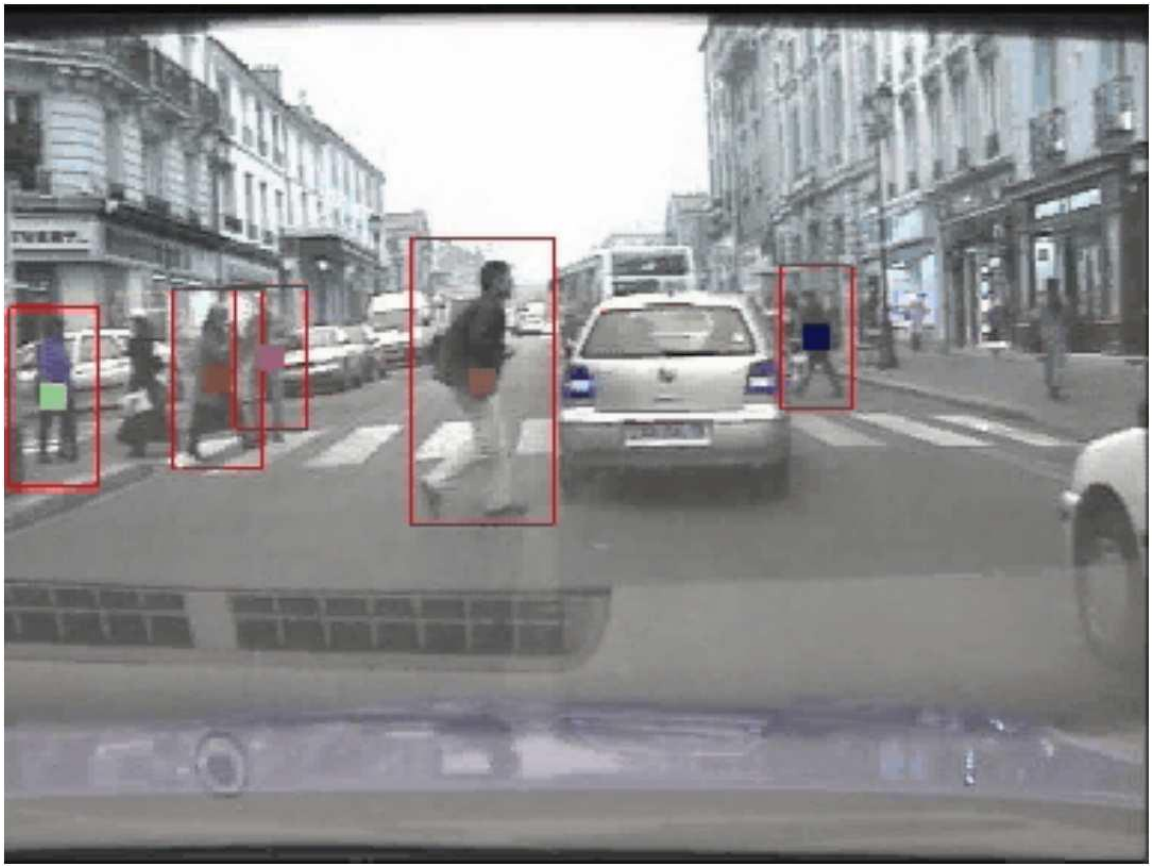
To know how is Active Learning very important part of Machine Learning we will discuss its different applications and implementations.

The main field that is implementing Active Learning technology is Classification. Companies like Microsoft and Google are using Active Learning researchers to enhance their classification of text files, search results… Etc.

I will list some research references for Microsoft and Google in this field.

In addition, we can classify the applications to:

- Text classification
- Image classification
- Pedestrian detection
- Network intrusion detection
- Visual object recognition
- Pedestrian Detection



At last, I will talk about some very practical work in this field:

- *Dualist*

  DUALIST is an interactive machine learning system for quickly building classifiers for text processing tasks. It does so by asking "questions" of a human "teacher" in the form of both data instances (e.g., text documents) and features (e.g., words or phrases). It uses active learning and semi-supervised learning to build text-based classifiers at interactive speed.
  Burr Settles developed dualist based on his papers.
  The goals of this project are threefold:

1. A practical tool to facilitate annotation/learning in text analysis projects.
2. A framework to facilitate research in interactive and multi-modal active learning. This includes enabling actual user experiments with the GUI (as opposed to simulated experiments, which are pervasive in the literature but sometimes inconclusive for use in practice) and exploring HCI issues, as well as supporting new dual supervision algorithms which are fast enough to be interactive, accurate enough to be useful, and might make more appropriate modeling assumptions than multinomial naive Bayes (the current underlying model).
3. A starting point for more sophisticated interactive learning scenarios that combine multiple "beyond supervised learning" strategies. See the proceedings of the recent ICML 2011 workshop on this topic.

This work is supported in part by DARPA (under contract numbers FA8750-08-1-0009 and AF8750-09-C-0179), the National Science Foundation (IIS-0968487), and Google. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsors.

- *Vowpal Wabbit*

The Vowpal Wabbit (VW) project is a fast out-of-core learning system sponsored by Microsoft Research and (previously) Yahoo! Research. Support is available through the mailing list.

There are two ways to have a fast learning algorithm: (a) start with a slow algorithm and speed it up, or (b) build an intrinsically fast learning algorithm. This project is about approach (b), and it has reached a state where it may be useful to others as a platform for research and experimentation.

There are several optimization algorithms available with the baseline being sparse gradient descent (GD) on a loss function (several are available); the code should be easily usable. Its only external dependence is on the boost library, which is often installed by default.

There are several features that (in combination) can be powerful:

1. Input Format. The input format for the learning algorithm is substantially more flexible than might be expected. Examples can have features consisting of free form text, which is interpreted in a bag-of-words way. There can even be multiple sets of free form text in different namespaces.
2. Speed. The learning algorithm is pretty fast---similar to the few other online algorithm

implementations out there. As one datapoint, it can be effectively applied on learning problems with a sparse terafeature (i.e. 1012 sparse features). As another example, it is about a factor of three faster than Leon Bottou's svmsgd on the RCV1 example in wall clock execution time.

3. Scalability. This is not the same as fast. Instead, the important characteristic here is that the memory footprint of the program is bounded independent of data. This means the training set is not loaded into main memory before learning starts. In addition, the size of the set of features is bounded independent of the amount of training data using the hashing trick.

To make sure how powerful VW can be we will see some examples of its functioning:

- *Predicting closed questions on Stack Overflow:*

  The Stack Overflow challenge is about predicting a status of a given question on SO. There are five possible statuses, so it is a multi-class classification problem.

  We would prefer a tool able to perform multiclass classification by itself. It can be done by hand by constructing five datasets, each with binary labels (one class against all others), and then combining predictions, but it might be a bit tricky to get right. Fortunately, Vowpal Wabbit supports multiclass classification.

In case you are wondering, Vowpal Wabbit is a fast linear learner. We like the “fast” part and “linear” is OK for dealing with lots of words, as in this contest. In any case, with more than three million data points it would not be that easy to train a kernel SVM, a neural net or what have you.

In addition, the result? Using VW actually made the prediction accurately.

Not just this example but also many others can be found in the VW link in the references.

IV. Conclusion:

Active Learning is a very exciting and “active” field of study in Machine Learning, which is not surprising because this trend combines advantages from both supervised and unsupervised learning. It reduces the cost of labeling data and increases accuracy.

V. Future Work:

I plan to study most of notable works in Active Learning and try to come up with a new or improved algorithm to make it faster and more accurate than the existing ones.

VI. References:

1- Machine Learning in Medical Imaging, IEEE Signal Processing Magazine, vol. 27, no. 4, July 2010, pp. 25-38
2- Foundations of Machine Learning, The MIT Press ISBN 9780262018258
3- Computer Science Handbook, Second Edition (Section VII: Intelligent Systems). Boca Raton, FL: Chapman & Hall/CRC Press LLC. ISBN 1-58488-360-X
4- http://en.wikipedia.org/wiki/Semi-supervised_learning
5- Settles, Burr (2009), "Active Learning Literature Survey", Computer Sciences Technical Report 1648. University of Wisconsin–Madison, retrieved 2010-09-14
6- Olsson, Fredrik. A literature survey of active machine learning in the context of natural language processing
7- http://www.causality.inf.ethz.ch/activelearning.php?page=tutorial
8- Donmez, P., Carbonell, J.G.: Proactive Learning: Cost-Sensitive Active Learning with Multiple Imperfect Oracles, in Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM '08), Napa Valley 2008
9- http://en.wikipedia.org/wiki/Active_learning
10- Cortes, Corinna; and Vapnik, Vladimir N.; "Support-Vector Networks", Machine Learning, 20, 1995
11- http://hunch.net/~active_learning/
12- Active Learning: Synthesis Lectures on Artificial Intelligence and Machine Learning. (Burr Settles)
13- http://code.google.com/p/dualist/
14- https://github.com/JohnLangford/vowpal_wabbit/wiki
15- http://fastml.com/blog/categories/vw/
16- http://research.microsoft.com/apps/pubs/default.aspx?id=141158
17- http://research.microsoft.com/apps/pubs/default.aspx?id=164140
18- http://research.microsoft.com/apps/pubs/default.aspx?id=70547
19- http://dl.acm.org/citation.cfm?id=1172940
20- http://cse.unl.edu/~sscott/research/r-area.shtml#active