Semi-supervised Clustering with User Feedback

David Cohn
Just Research
4616 Henry St.,
Pittsburgh, PA 15213
cohn@justresearch.com

Rich Caruana
Just Research
4616 Henry St.,
Pittsburgh, PA 15213
caruana@justresearch.com

Andrew McCallum
Just Research
4616 Henry St.,
Pittsburgh, PA 15213
mccallum@justresearch.com

Abstract

We present a new approach to clustering based on the observation that "it is easier to criticize than to construct." Our approach of *semi-supervised clustering* allows a user to iteratively provide feedback to a clustering algorithm. The feedback is incorporated in the form of constraints which the clustering algorithm attempts to satisfy on future iterations. These constraints allow the user to guide the clusterer towards clusterings of the data that the user finds more useful. We demonstrate semi-supervised clustering with a system that learns to cluster news stories from a Reuters data set.

1 Introduction

Consider the following problem: you are given 100,000 text documents (e.g., papers, newsgroup articles, or web pages) and asked to group them into classes or into a hierarchy such that related documents are grouped together. You are not told what classes or hierarchy to use or what documents are related; you have some criteria in mind, but may not be able to say exactly what it is. You are creating this taxonomy so that the documents can be browsed and accessed efficiently, either by yourself or by other people. We refer to this as the Yahoo! problem.

The Yahoo! problem is ubiquitous. It occurs in many fields and with many different types of "documents." The web has created a number of new examples of the Yahoo! problem, but many exist outside of the web. Librarians, astronomers, biologists — everyone tasked with creating a taxonomy from data faces the Yahoo! problem.

We propose the following iterative solution to the Yahoo! problem:

1. Give the 100,000 documents to an unsupervised clustering algorithm and have it cluster them.
2. Browse the resulting clusters and tell the system which clusters you like, and which clusters you don't like. Don't do this for all the clusters, just for some of the ones you browsed. Provide feedback to the system by saying:
   - "This document doesn't belong in here"
   - "Move this document to that cluster"
   - "These two documents shouldn't be (or should be) in the same cluster".

   Don't do this for all, or even many, of the documents; only for the few that look most out of place.

3. After your critique, re-cluster the documents, allowing the clustering algorithm to modify the the distance metric parameters to try to find a new clustering that satisfies the constraints you provided in the critique.
4. Repeat this until you are happy with the clustering.

This solution is distinct from both traditional supervised and unsupervised learning. Unsupervised clustering takes an unlabeled collection of data and, without intervention or additional knowledge, partitions it into sets of examples such that examples within clusters are more "similar" than examples between clusters. Much work in unsupervised clustering is dedicated to the problem of manually engineering similarity criteria that yield good partitioning of data for a given domain.

Supervised learning, on the other hand, assumes that the class structure or hierarchy already is known. It takes a set of examples with class labels, and returns a function that maps examples to class labels. The goal of supervised learning is to learn mappings that are accurate enough to be useful when classifying new examples, and perhaps to learn mappings that allow users to understand the relationships between the data and the labels, such as which features are important.

Semi-supervised clustering falls between the extremes of totally unsupervised clustering and totally supervised learning. The main goal of our approach to semi-supervised clustering is to allow a human to "steer" the clustering process so that examples can be partitioned into a useful set of clusters with minimum time and human effort. A secondary goal of semi-supervised clustering is to give the user a way to interact and play with the data so that they can understand it better.[1]

Our approach to semi-supervised clustering assumes that the human user has in their mind criteria that enable them to evaluate the quality of a clustering. It does not assume that the user is conscious of what they think defines a good clustering but that, as with art, they will "know it when they see it." Most importantly, semi-supervised clustering never expects a user to write a function that *defines* the clustering criterion. Instead, the user *interacts* with the clustering system, which attempts to learn a criterion that yields clusters the user is satisfied with.

Semi-supervised clustering with user feedback is closely related to active learning (Cohn et al., 1996). In active learning, the learning system attempts to select which data points, if labeled, would be most informative. In semi-supervised clustering, the *human* selects the data points, and puts on them a wide array of possible *constraints* intead of labels. These two key differences point toward the important situations in which the semi-supervised approach is preferable. (1) In some clustering problems the desired similarity metric may be so different from the default that traditional active learning would make many inefficient queries. This problem also arises when there are many different plausible clusterings. Although less automated, a human browsing the data would do less work by selecting the feedback data points themself. (2) The intuitive array of possible constraints are easier to apply than labels, especially when the final clusters are not known in advance. (3) The very act of human browsing can lead to the discovery of what clusters are desired. Semi-supervised learning can thus be seen as a method of data exploration and pattern discovery, efficiently aided by cluster-based summarization.

One of the primary challenges of semi-supervised clustering is finding ways to make use of user feedback during clustering. The remainder of this paper describes one simple illustrative way in which this may be accomplished. Other challenges that will be addressed by future research in semi-supervised clustering are briefly touched on in the discussion section.

2 Clustering

Formally, clustering is the process of partitioning a data set into subsets such that all members of a given subset are "similar" by some distance measure D. We will denote the distance between two examples x_1

[1]Demiriz et al. (1999) independently introduced a semi-supervised clustering model similar to the one we describe here. The main distinction between our work and theirs is our use of iterative feedback to acquire labelings; Demiriz et al. assume that all available labels are given *a priori*.

and x_2 as $D(x_1, x_2)$. We can generalize this to refer to the distance between two cluster centers or between a cluster center and an example.

From a statistical standpoint, clustering may be viewed as a way of doing factor analysis on a mixture model. In this setting it is assumed that observed data result from sampling a mixture of unknown sources. Clustering attempts to infer the properties of the sources (the cluster centers or prototypes), and determine which source is most likely to be responsible for a given example.

The two most popular approaches to clustering are agglomerative clustering and prototype-based clustering. In agglomerative clustering, each datum is initially placed in its own cluster. The clusters that are most similar (according to D) are iteratively merged, until the desired number of clusters is reached, or some limit on data likelihood or distortion is exceeded (see Hofmann & Buhmann (1997), for an in-depth treatment of agglomerative clustering).

In prototype-based clustering, the final number of clusters is set a priori, and the corresponding prototypes are found using a form of EM. Each prototype is initialized to some position. Examples are assigned to prototypes according to their similarity to each prototype (the assignment may be 0-1 or fractional, depending on the algorithm). Prototypes are then adjusted to maximize the data likelihood, or, equivalently, minimize the data distortion. The assignment/adjustment process is repeated until no significant changes result (see Meilă and Heckerman (1998) for concise review of prototype-based clustering).

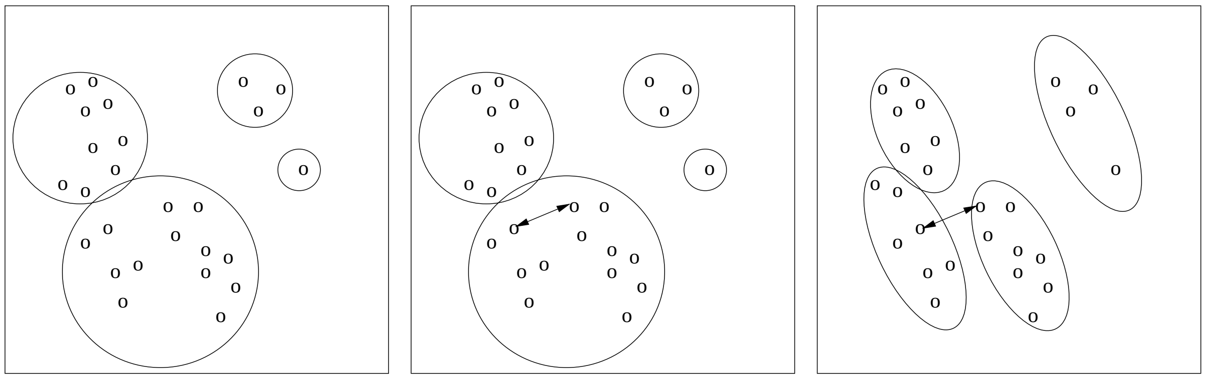

Figure 1: Illustration of semi-supervised clustering. Given an initial clustering, the user specifies two points that should not have been placed in the same cluster. The system warps its metric, allowing it to find a clustering that respects the constraint.

3 Semi-supervised clustering

The goodness of any clustering depends on how well the metric D matches the user's (perhaps unknown) internal model of the target domain. We propose allowing the user to impose their model on the metric *via* the clustering algorithm, by having the user provide the algorithm with feedback, and allowing it to alter the metric so as to accommodate that feedback. Not only is it easier to critique than to construct, but the user's criticism can take many forms — specifying that a particular example does/doesn't belong in a particular cluster, that two examples do/don't belong in the same cluster, or that a particular cluster is good (and should be preserved) or bad (and should be split up).

Feedback may be incorporated into the metric as constraints to be respected by the clustering algorithm. Consider two examples x_1 and x_2 that are constrained by the user feedback to be in separate clusters. When the clustering algorithm attempts a partitioning which places x_1 and x_2 in the same cluster, the metric may be altered to increase the distance between x_1 and x_2 until one or the other of them falls in a different cluster (Figure 1). Other constraints may be implemented similarly, shrinking the distance between some example and a cluster prototype, or increasing the distance between a cluster prototype and all the examples assigned to it.

3.1 Semi-supervised document clustering

We adopt a statistical approach to document clustering resulting from the naive Bayes model of document generation (McCallum & Nigam, 1998). Given a vocabulary V, a document is assumed to be a "bag of words" generated from a multinomial distribution θ. The probability of document d is

$$p(d) = \prod_{w_j \in V} p(w_j|\theta)^{N(w_j,d)},$$

where $p(w_j|\theta)$ is the parameterized probability of word w_i being generated, and $N(w_j,d)$ is the number of times w_j appears in the document. Each document d forms an estimate of a multinomial distribution θ_d; likewise, each cluster c forms an estimate θ_c composed from the θ_d of its constituent documents.[2]

We search for optimal clusters, applying EM as follows: for each cluster c and document d, we compute $p(d|c)$ and apply Bayes rule to compute $p(c|d)$. Each cluster is given partial ownership of a document proportional to $p(c|d)$. The cluster parameters θ_c are recomputed as the weighted sum of their component documents, and the process is repeated. The algorithm is guaranteed to converge to a locally optimal clustering.

In this probablistic setting, the natural measure of dissimilarity between two documents d_1 and d_2 is the probability that they were generated by the same multinomial; this is proportional to the *KL divergence to the mean* of their multinomial distributions: $D_M(d_1||d_2) = |d_1|D(d_1||M_{12}) + |d_2|D(d_2||M_{12})$, where M_{12} is a distribution composed of the mean word probabilities of d_1 and d_2, $|d|$ is the length of document d, and $D(d||M)$ is the standard KL divergence of d to M. The advantage of KL divergence to the mean of the multinomial distributions is that it is symmetric, unlike standard KL divergence.

To implement our constraints, we augment the standard KL divergence $D(d_1||d_2)$ with a weighting function

$$D'(d_1||d_2) = \prod_{w_j \in V} \gamma_j \cdot p(w_j|\theta_{d_1}) \log\left(\frac{p(w_j|\theta_{d_2})}{p(w_j|\theta_{d_1})}\right)$$

where γ_j may be interpreted as indicating the importance of w_j for distinguishing d_1 and d_2. Then, given a constraint that d_1 and d_2 must be in separate clusters, we can warp the metric by computing

$$\begin{aligned}\frac{\partial D'_M(d_1||d_2)}{\partial \gamma_j} \quad = \quad & |d_1|p(w_j|\theta_{d_1}) \log\left(\frac{p(w_j|\theta_{d_1 d_2})}{p(w_j|\theta_{d_1})}\right) + \\ & |d_2|p(w_j|\theta_{d_2}) \log\left(\frac{p(w_j|\theta_{d_1 d_2})}{p(w_j|\theta_{d_2})}\right)\end{aligned}$$

and hillclimbing over γ to increase the effective distance between the two. This gradient tells us the direction to move the γ's in order to increase (or decrease) the separation between two documents. (In the current experiments we constrain the γ's to be positive, but it might be interesting to relax this and allow some γ's to go negative.) These γ's are then used by EM in the analogous distance measure between a document and a prototype. Thus we inject a learned distance metric into the clustering algorithm by changing the E-step.

Other constraints described in the previous section may be similarly implemented by hillclimbing over the example-to-cluster and cluster-to-cluster distance. Note that the linear warping we describe will not guarantee that all constraints can be satisfied - some clusterings desired by the user may be non-convex and unrealizable in the space of models supported by naive Bayes. In this case, the hillclimbing will converge to a weighting that provides a local minimum of constraint violations. Local or nonlinear warpings of the distance metric, such as the one described by Friedman (1994) and Yianilos (1996) may be of use in these situations.

[2] The estimates for word probabilities are derived from the relative word frequencies in the documents. Following McCallum & Nigam (1998), we smooth with a LaPlacean prior to avoid zero word probabilities.

4 Experiments

In this section, we illustrate the semi-supervised approach on a small document clustering problem. We used a set of 25 documents each from five Reuters topic areas: business, health, politics, sports and tech. Starting from five randomly initialized prototypes, the EM-based clustering algorithm described in the previous sections found clusters that maximized data likelihood.

Each time clustering converged, we added a constraint. We simulated a human user by identifying two documents from the same cluster whose sources were different Reuters topics, and constraining them to be in different clusters.[3] For each unsatisfied constraint, we reweighted the divergence by a fixed number of hillclimbing steps. The cluster prototypes were then re-initialized, and EM training repeated.

4.1 Clustering performance

Figure 2a compares the performance of supervised, unsupervised and semi-supervised learning. For the supervised learner, classification accuracy is plotted. For unsupervised and semi-supervised learners, cluster purity is plotted. Cluster purity is the fraction of examples that would be classified correctly if all examples were assigned the majority label in each cluster.

After only a few constraints have been added, cluster purity increases sharply over that of unsupervised clustering. It is not clear, however, how to fairly compare the performance of semi-supervised clustering with that of fully supervised clustering: constraints do not exactly correspond to labeled examples, and it is uncertain what constitutes a proper test set. In supervised learning, documents used for training are traditionally excluded from the test set, since their labels are already known. But the semi-supervised model clusters (and is tested on) the entire corpus, so it is also reasonable to gauge it against a supervised learner tested the same way. In the figure we show the cluster purity of supervised learning on the training set as well as its generalization to an independent test set.

The semi-supervised learner reaches its asymptotic performance after about 10 constraints have been added; the supervised learners require between 3 and 6 times more labeled examples to reach that level of performance.[4] It is interesting to note that the performance of the semi-supervised learner actually begins to decrease after roughly 20 constraints have been added. The Reuters data set contains many documents which appear under more than one topic (an identical article on Microsoft, for example, appears under both 'business' and 'tech'). We hypothesize that, in an attempt to separate these unseparable documents, the learner is pushing its term weightings to unhealthy extremes. Future experiments will investigate this hypothesis and how best to overcome it.

Preliminary experiments on a different data set consisting of 20,000 USENET articles suggest that semi-supervised clustering is just as effective with large data sets. More importantly, these experiments show that semi-supervised clustering is able to cluster the same data according to different orthogonal criteria. This dataset contains articles on four subjects: aviation simulators, real aviation, auto simulators and real autos. Semi-supervised clustering can cluster the simulators and real groups together (e.g., aviation simulators and real aviation) or the auto and aviation groups together (e.g., aviation simulators and auto simulators) depending on the feedback provided by the user. In both cases it does so at about 80% accuracy with 10 constraints. When the distance metric is not adjusted, the same constraints give an average of only 64% accuracy. (Purely unsupervised clustering achieves only about 50% accuracy.)

[3]A fully-operational semi-supervised clustering system would benefit from a graphical user interface that permits efficient browsing of the current clusters and supports easy specification of user constraints. See the discussion of Scatter/Gather in later in this paper.

[4]To assure ourselves that metric-warping alone wasn't responsible for the performance disparity, we also incorporated metric warping into the supervised clusterer, shrinking the divergence between a document and its assigned cluster. The addition resulted in no significant performance improvement.

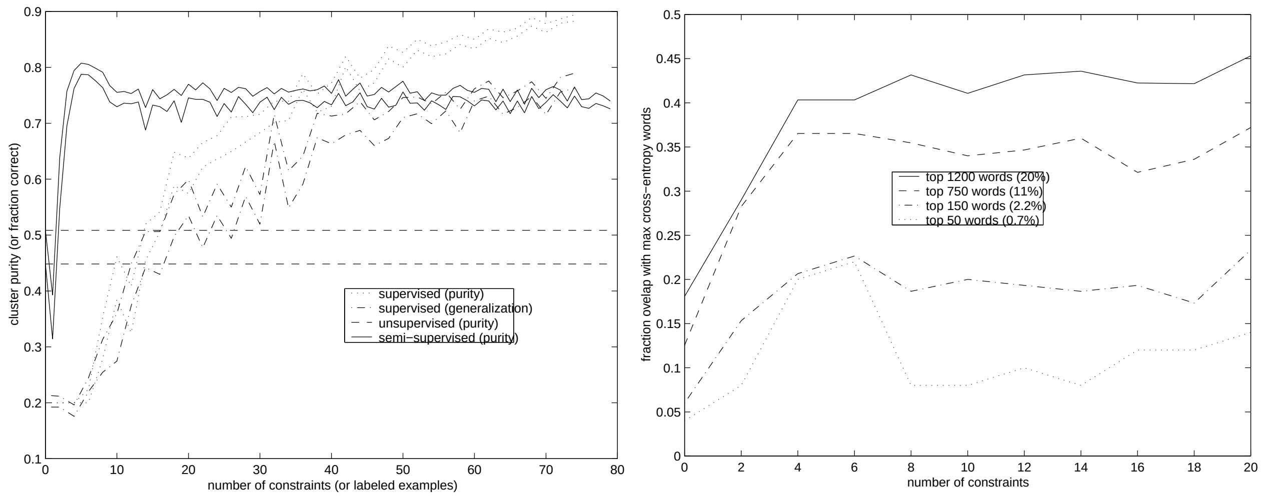

Figure 2: (a) Learning curves for supervised, unsupervised and semi-supervised clustering. For supervised clustering, cluster purity (measured on the train set) and generalization (measured on an independent test set) are plotted against the number of labeled examples; for semi-supervised clustering, purity is plotted against the number of constraints. Averages over 10 runs each, with the upper and lower lines indicating error bars at one standard deviation. See text for details. (b) Fraction overlap of the top n weighted words with top n words ranked by information gain on fully-supervised data. As the number of constraints increases, there is increasing correlation with words that strongly affect class conditional probabilities. Note that this overlap is achieved with far fewer constraints than the number of labels in the fully-supervised data.

4.2 Learning term weightings

Adjusting γ_j warps the metric by adjusting the resolving power of word w_j, essentially identifying which words are most useful for distinguishing documents. If γ_j is large, small disparities in the frequency of w_j become important and will tend to separate documents; if γ_j is small, large disparities in frequency will be ignored.

Empirically, this behavior is borne out on the Reuters experiments - words that subjectively appear highly relevant for distinguishing topics, such as ‘Iraq’, ‘economy’, ‘weapons’ and ‘council’ are given large weightings. We computed the information gain of w_j using all document labels (Mitchell 1997), and compared it with γ_j. Figure 2b shows the overlap between the top-weighted $n\%$ words in the vocabulary with the same words ranked by information gain. After about a dozen constraints, semi-supervised clustering learns word weightings with moderate overlap to the word weightings learned by supervised learning from all 125 document labels.

5 Discussion

This paper only scratches the surface of semi-supervised clustering with user feedback. There are still many issues to be addressed; we touch on a few of these here.

5.1 Constraints vs. Labels

When applying supervised learning to classification problems, it is assumed that the users know the target classes, and have labeled examples from each target class. In many interesting problems, this is an unrealistic assumption. A semi-supervised system allows users to give label-like information to the learner without having to know labels. Although user feedback in semi-supervised clustering serves a similar role as class labels serve in supervised learning, comparing supervised learning with semi-supervised clustering is an apples–to–oranges comparison. Semi-supervised clustering usually will be applied to problems where labels are not readily available. Evaluating clustering systems is difficult and usually subjective. We compare the performance of semi-supervised clustering to supevised learning using a labelled dataset principally to avoid this subjectivity.

The performance disparity between supervised and semi-supervised clustering is surprising. While we have argued that it is easier to provide constraints than labels, constraints also provide less information than labels. Constraints don't require the user to know the correct label (or even what labels exist!) — only the relationship among pairs or sets of labels. There are only 125 possible labels in the small Reuters dataset, but thousands of possible separation constraints. Yet empirically, even with very few constraints, the semi-supervised learner is able to perform surprisingly well.

One explanation is in the connection to active learning. As a means of of user feedback, the addition of a constraint indicates a problem, and effectively acts as counterexamples for the present clustering. Counterexamples are a powerful tool for doing active learning which, in some situations, is much more efficient than learning from randomly labeled examples (Angluin, 1987). The user, by iteratively directing the clusterer's attention towards points that are incorrectly clustered, gives a semi-supervised clustering system many advantages of an active learning system.

5.2 Types of User Feedback

As we indicate above, there are many different types of feedback that users might provide to a semi-supervised clsutering system. One type of feedback is the constraints on individual data points and clusters we used above. But many other forms of feedback might prove useful, as well. For example, a user might tell the system that the current clustering is too coarse or too fine. Or the user might point to a cluster and indicate that the cluster is bad without saying how it is bad. Similarly, a user might indicate that a cluster is good, suggesting that future re-clusterings of the data should attempt to maintain this cluster. Users might also give feedback that is not cluster specific, such as telling the system that the entire clustering looks bad, and that the next clustering should be very different.

Some types of user feedback may require adaptive clustering that cannot easily be handled by the γ weighting scheme we used above. For example, we are working on an approach to finding good — but qualitatively different — clusterings of the same data by exploiting EM's *weakness* for getting trapped in local minima. Different local minima may capture qualitatively different ways of clustering the data, one of which may better match the user's internal preference function than the deepest minima the system can find. In the long run we hope to develop a general framework for representing user feedback about clusters.

5.3 Other Applications

We believe there are many applications of feedback-driven semi-supervised clustering. Imagine a Yahoo! hierarchy for web pages that allows the user to tailor the hierarchy to better match their own interests by providing feedback while browsing. Similarly, consider an automatic email system where a user allows the system to cluster email into related mailboxes instead of manually specify the mailboxes. Semi-supervised feedback would allow the user to tailor mailbox clusters to fit their (possibly changing) needs. As a different example, consider a user clustering proteins into homology groups (groups of proteins with similar structure). Large proteins have complex structure and could be clustered many different ways. A feedback-driven semi-supervised clustering system would allow the user to explore many different ways the proteins might be clustered, and to find clusterings most suitable to their purposes.

5.4 Related work

As indicated above, our model is similar to the model of Demiriz et al. They report how a fixed set of labeled examples may be used to bias a clustering algorithm; we investigate how a user, interacting with the system, may efficiently guide the learner to a desired clustering.

Our technique of incorporating user feedback is a cousin to relevance feedback, a technique for information retrieval (Buckley & Salton, 1995). Given a query and initial set of retrieved documents, relevance feedback asks the user to tag documents as being more or less relevant to the query being pursued. As the process is iterated, the retrieval system builds an increasingly accurate model of what the user is searching for.

The question of how a user (or teacher) may best select examples to help a learner identify a target concept is the focus of much work in computational learning theory. See Goldman and Kearns (1995) for a detailed treatment of the problem.

The Scatter/Gather algorithm (Cutting et al., 1992) is an interactive clustering algorithm, designed for information retrieval. The system provides an initial clustering of data. When the user selects a subset of the clusters for further examination, the system gathers their components and regroups them to form new clusters. Scatter/Gather aims at pursuing and finding structure in a small part of a corpus. This makes it an interesting complement to our approach: Scatter/Gather may provide an effective means for browsing and focusing on clusters of interest, and semi-supervised learning may be an effective means of improving the quality of those clusters.

Note that we do not compare our performance to that of other purely unsupervised clustering systems such as AutoClass (Stutz & Cheeseman, 1994), COBWEB (Fisher, 1987) or Iterative Optimization (Fisher, 1996). The contribution of our work is not to introduce a new clustering *algorithm*, but an approach that allows user feedback to guide the clustering. While we have illustrated our approach on a relatively simple system, we believe it is equally applicable to more sophisticated algorithms, and expect that it will provide similar improvements over the unsupervised variants.

References

Angluin, D. (1987) Learning Regular Sets from Queries and Counterexamples, *Information and Computation*, 75(2):87–106.

Buckley, C. & Salton, G. (1995) Optimization of Relevance Feedback Weights. *SIGIR 1995.*

Cheeseman, P., Stutz, J., Self, M., Taylor, W., Goebel, J., Volk, K. & Walker, H. (1989) Automatic Classification of Spectra from the Infrared Astronomical Satellite, NASA Reference Publication 1217.

Cohn, D. Ghahramani, Z. and Jordan, M. (1996) Active learning with statistical models, *Journal of Artificial Intelligence Research* 4: 129-145.

Cutting, D., Karger, D., Pedersen, J. & Tukey, J. (1992) Scatter/Gather: A Cluster-based Approach to Browsing Large Document Collections, *SIGIR 1992.*

Demiriz, A., Bennett, K. and Embrechts, M. (1999) Semi-Supervised Clustering Using Genetic Algorithms. In Dagli et al, eds., *ANNIE 1999.*

El-Yaniv, R., Fine, & Tishby, N. (1998) Agnostic Classification of Markovian Sequences. In M. Jordan et al, eds., *Neural Information Processing Systems 10*, MIT Press.

Fisher, D. (1996) Iterative Optimization and Simplification of Hierarchical Clusterings, *Journal of Artificial Intelligence Research*, 4:147–179.

Fisher, D. (1987) Knowledge Acquisition via Incremental Conceptual Clustering, *Machine Learning* 2:139–172.

Friedman, J. (1994) Flexible Metric Nearest Neighbor Classification. Stanford University, Department of Statistics Tech Report.

Goldman, S. and Kearns, M. (1995) On the Complexity of Teaching. *Journal of Computer and System Sciences* 50(1):0–31.

Hofmann, T. & Buhmann, J. (1997) Pairwise Data Clustering by Deterministic Annealing, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(1).

McCallum, A., & Nigam, K. (1998) A Comparison of Event Models for Naive Bayes Text Classification, *AAAI-98 Workshop on "Learning for Text Categorization"*.

Meilă, M. & Heckerman D. (1998) An Experimental Comparison of Several Clustering and Initialization Methods, in G. Cooper & S. Moral, eds., *Uncertainty in Artificial Intelligence*, Morgan Kaufmann.

Mitchell, T. (1997) Machine Learning, McGraw-Hill.

Stutz, J., & Cheeseman, P. (1994) AutoClass - a Bayesian Classification System. In *Maximum Entropy and Bayesian Methods*, J. Skilling et al., eds., Kluwer.

Yalinos, P. (1995) Metric learning via normal mixtures, NEC Research Institute Technical Report.