This chapter describes the research literature relevant to the primary aspects of this thesis. The core aspects of this thesis are machine learning applications to natural language processing and classification techniques. Both these fields have received a lot of attention in the past years and there are a number of popular texts with relevant background material [Duda et al., 2001; Russell and Norvig, 2003; Manning and Schutze, 1999; Jurafsky and Martin, 2008]. As there is an enormous amount of literature available on both these aspects, these works can be described along several dimensions.

2.1. Review of Research Work in NLP

Natural Language Processing (NLP) is that field of computer science which consists of interfacing computer representations of information with natural languages used by humans. It examines the use of computers in understanding and manipulating the natural language text and speech. Over the past years, a lot of research has been done in the field of NLP. Some of the recent works have been discussed here. Kumarana et al. (2011) have developed a multilingual content creation tool for Wikipedia. Optimal Search for Minimum Error Rate Training has been discussed by Michel and Chris (2011). Associating Web Queries with Strongly-Typed Entities [Patrick et al., 2011], Linguistic Style Accommodation in Social Media [Cristian et al., 2011], Predicting the Importance of Newsfeed Posts and Social Network Friends [Tim et al., 2010], Wiki BABEL: A System for Multilingual Wikipedia Content [Kumaran et al., 2010], The utility of article and preposition error correction systems for English language learners: Feedback and Assessment [Martin et al., 2010]. The work presented in this Section has been previously published [Khan, Dar and Quadri, 2012].

2.1.1. Theoretical developments in NLP

Theoretical developments in NLP can be grouped into following classes: (i) statistical and corpus-based methods in NLP, (ii) use of WordNet for NLP research, (iii) use of finite-state methods in NLP.

2.1.1.1. Statistical Methods

The models and methods used in solving NLP problems are broadly classified into two types: deterministic and stochastic. A mathematical model is called deterministic if it does not involve
the concept of probability; otherwise it is said to be stochastic. A stochastic model can be probabilistic or statistical, if its representation is from the theories of probability or statistics, respectively [Edmundson, 1968]. Statistical methods are used in NLP for a number of purposes, e.g., speech recognition, part-of-speech tagging, for generating grammars and parsing, word sense disambiguation, and so on. There has been a lot of research in these areas. Geoffrey Zweig and Patrick Nguyen (2009) have proposed a segmental conditional random field framework for large vocabulary continuous speech recognition [Geoffrey and Patrick 2009]. Gerasimos Potamianos, Chalapathy Neti, Ashutosh Garg, Guillaume Gravier and Andrew W. Senior (2003) have reviewed Advances in the Automatic Recognition of Audio-Visual Speech and have presented the algorithms demonstrating that the visual modality improves automatic speech recognition over all conditions and data considered [Gerasimos et al., 2003]. Raymond J. Mooney has developed a number of machine learning methods for introducing semantic parsers by training on a corpus of sentences paired with their meaning representations in a specified formal language [Raymond, 2007]. Marine CARPUAT and Dekai WU (2007) have shown that statistical machine translation can be improved by using word sense disambiguation. They have shown that if the predictions of the word sense disambiguation system are incorporated within a statistical machine translation model then the translation quality is consistently improved [Marine and Dekai, 2007].

2.1.1.2. Use of WordNet for NLP research

Mihalcea & Moldovan (1999) have proposed the use of WordNet to make the outcome of statistical analysis of natural language texts better. WordNet or the electronic dictionary is developed at Princeton University. It is a large database that serves as an important NLP tool consisting of nouns, verbs, adjectives and adverbs. These are arranged in the form of synonym sets (synsets). Each set represents one underlying lexical concept. These sets are linked with each other by means of conceptual-semantic and lexical relations. There are different wordnets for about 50 different languages, but they are not complete like the original English WordNet [Gerard and Gerhard, 2009]. WordNet is now used in a number of NLP research and applications. One of the most important applications of WordNet in NLP is EuroWordNet developed in Europe. EuroWordNet is a multilingual database which consists of WordNets for the European languages. It has been structured in the same way as the WordNet for English. A
methodology for the automatic construction of a large-scale multilingual lexical database has been proposed where words of many languages are hierarchically organized in terms of their meanings and their semantic relations to other words. This database is capable of organizing over 800,000 words from over 200 languages, providing over 1.5 million links from words to word meanings. This universal wordnet has been derived from the Princeton WordNet. Lars Borin and Markus Forsberg have given a comparison between WordNet and SALDO. SALDO is a Swedish lexical resource which has been developed for language technology applications [Lars and Markus, 2009]. Japanese WordNet currently has 51,000 synsets with Japanese entries. Methods for enhancing or extending the Japanese Wordnet have been discussed. These include: increasing the cover, linking it to examples in corpora and linking it to other resources. In addition various plans have been outlined to make it more useful by adding Japanese definition sentences to each synset [Francis et al., 2009]. The use of WordNet in multimedia information retrieval has also been discussed and the use of external knowledge in a corpus with minimal textual information has been investigated. The original collection has been expanded with WordNet terms in order to enrich the information included in the corpus and the experiments have been carried out with original as well as expanded topics [Manuel et al., 2011]. A Standardized Format for Wordnet Interoperability [Claudia et al., 2009] has been given i.e., WordNet- LMF. The main aim of this format is to provide the WordNet with a format representation that will allow easier integration among resources sharing the same structure (i.e. other wordnets) and, more importantly, across resources with different theoretical and implementation approaches.

2.1.1.3. Use of finite state methods in NLP

The finite-state automation is the mathematical tool used to implement regular expressions – the standard notation for characterizing text sequences. Different applications of the Finite State methods in NLP have been discussed [Jurafsky and Martin, 2000; Kornai, 1999; Rocheand Shabes, 1997]. From past many years the finite state methods have been used in presenting various research studies on NLP. The FSMNLP workshops are the main forum of the Association for Computational Linguistics’ (ACL) Special Interest Group on Finite-State Methods (SIGFSM)[Anssiet al., 2011].
2.1.2. NLP Applications

There are a number of applications of NLP e.g. machine translation, natural language text processing and summarization, user interfaces, multilingual and cross language information retrieval (CLIR), speech recognition, and expert systems, and so on. In this paper we discuss automatic abstracting and information retrieval.

2.1.2.1. Automatic Abstracting

Automatic abstracting or text summarization is a technique used to generate abstracts or summaries of texts. Due to the increase in the amount of online information, it becomes very important to develop the systems that can automatically summarize one or more documents[Dragomir et al., 2002]. The main aim of summarization is to differentiate between the more informative or important parts of the document and the less ones [Dipanjan and Andre, 2007]. According to Radev et al. (2002) a summary can be defined as piece of text that can be produced from one or more texts in a way such that it conveys important information in the original text(s), and whose size is not more than half of the original text(s) and mostly significantly less than that". The summary can be of two types i.e. abstraction or extraction. Abstract summary is one in which the original documents” contents are paraphrased or generated, whereas in an extract summary, the content is preserved in its original form, i.e., sentences [Krysta et al, 2007]. Extracts are formed by using the same words, sentences of the input text, while abstracts are formed by regenerating the extracted content. Extraction is the process of identifying the important contents in the text while in abstraction the contents are regenerated in new terms. When the summaries are produced from a single document, it is called single document summarization. Multidocument summarization has been defined as a process of producing a single summary from a number of related documents. A lot of research has been done on automatic abstracting and text summarization. Zajic et al [David et al., 2008] have presented single-document and multi-document summarization techniques for email threads using sentence compression. They have shown two approaches to email thread summarization i.e. Collective Message Summarization (CMS) and Individual Message Summarization (IMS). NeATS[Chin and Eduard, 2002] is a multidocument summarization system in which relevant or interesting portions about some topic are extracted from a set of documents and presented in coherent order. NetSum [Krystael et al, 2007] is an approach to automatic summarization based on
neural networks. Its aim is to obtain those features from each sentence which helps to identify its importance in the document. A text summarization model has been developed which is based on maximum coverage problem and its variant [Hiroya and Manabu, 2009]. In this some decoding algorithms have been explored such as a greedy algorithm with performance guarantee, a randomized algorithm, and a branch-and-bound method. A number of studies have been carried out on text summarization. An efficient linear time algorithm for calculating lexical chains has been developed for preparing automatic summarization of documents [Silber and McCoy, 2000]. A method of automatic abstracting has been proposed that integrates the advantages of both linguistic and statistical analysis. Jin and Dong-Yan (2000) have proposed a methodology for generating automatic abstracts that provides an integration of the advantages of methods based on linguistic analysis and those based on statistics [Song and Zhao, 2000].

2.1.2.2. Information Retrieval

Information retrieval (IR) is concerned with searching and retrieving documents, information within documents, and metadata about documents. It is also called document retrieval or text retrieval. IR concerns with retrieving documents that are necessary for the users’ information. This process is carried out in two stages [Jun and Jianhan, 2009]. The first stage involves the calculation of the relevance between given user information need and the documents in the collection. In this stage probabilistic retrieval models that have been proposed and tested over decades are used for calculating the relevance to produce a “best guess” at a document’s relevance. In the second stage the documents are ranked and presented to the user. In this stage the probability ranking principle (PRP) [Cooper, 1971] is used. According to this principle the system should rank documents in order of decreasing probability of relevance. By using this principle the overall effectiveness of an IR system maximizes.

There has been a lot of research in the field of information retrieval. Some of the recent developments are included here. ChengXiangZhai (2008) has given a critical review of statistical language models for information retrieval. He has systematically and critically reviewed the work in applying statistical language models to information retrieval, summarized their contributions, and pointed out outstanding challenges [ChengXiang, 2008]. Nicholas J. Belkin has identified and discussed few challenges for information retrieval research which come under the range of association with users [Nicholas, 2008]. An efficient document ranking algorithm
has been proposed that generalizes the well-known probability ranking principle (PRP) by considering both the uncertainty of relevance predictions and correlations between retrieved documents [Jun and Jianhan, 2009]. Michael et al have discussed the various problems, directions and future challenges of content-based music information retrieval [Michael et al., 2008]. A unified framework has been proposed that combines the modeling of social annotations with the language modeling-based methods for information retrieval [Ding et al., 2008].

2.1.3. NLP Interfaces

A natural language interface accepts commands in natural language and sends data to the system which then provides the appropriate responses to the commands. A natural language interface translates the natural language statements into appropriate actions for the system. A large number of natural language interfaces have been developed [Stock, 2000]. A number of question answering systems are now being developed that aim to provide answers to natural language questions, as opposed to documents containing information related to the question. These systems use a variety of IE and IR operations to get the correct answer from the source texts. In information retrieval and NLP, question answering (QA) is the task of automatically answering a question posed in natural language. To find the answer to a question, a QA computer program may use either a pre-structured database or a collection of natural language documents. Unlike information retrieval systems (Internet search engines), QA systems do not retrieve documents, but instead provide short, relevant answers located in small fragments of text. That is why QA systems are significantly slower and require more hardware resources than information retrieval systems [Surdeanu et al., 2002]. QA track of TREC (Text Retrieval Conference) have shown some interesting results. Several steps were included in the technology used by the participants in the QA track. First, words like ‘who’, ‘when’ were identified to guess what was needed; and then a small portion of the document collection was retrieved using standard text retrieval technology. This was followed by a shallow parsing of the returned documents for identifying the entities required for an answer. If no appropriate answer type was found then best matching passage was retrieved. In TREC-8, the first QA track of TREC, the most accurate QA systems could answer more than 2/3 of the questions correctly [Voorhees, 1999]. In the second QA track (TREC-9), the best performing QA system, the Falcon system from Southern Methodist University, was able to answer 65% of the questions [Voorhees, 2000]. In the first two QA tracks
the questions were simple. In TREC 2001 QA track, which was the third running of a QA track in TREC, a number of conditions were included for increasing the practicality and complexity of the task [Ellen, 2001]. The TREC 2002 track repeated the main and list tasks from 2001, but with the major difference of requiring systems to return exact answers. The change to exact answers was motivated by the belief that a system’s ability to recognize the precise extent of the answer is crucial to improving question answering technology [Ellen, 2002]. These runnings of QA track have been carried out every year till date by adding different conditions to make the QA tracks more realistic.

2.1.4. NLP Software

A number of NLP software packages and tools have been developed, some of which are available for free, while others are available commercially. These tools have been broadly classified into different types some of which are mentioned here. General Information Tools (e.g. Sourcebank – a search engine for programming resources., The Natural Language Software Registry), Taggers and Morphological Analyzers (e.g. A Perl/Tk text tagger, AUTASYS – which is a completely automatic English Wordclass analysis system, TreeTagger – a language independent part-of-speech tagger, Morphy – which is a tool for German morphology and statistical part-of-speech tagging), Information Retrieval & Filtering Tools (e.g. Rubryx: Text Classification Program, seft – a Search Engine For Text, Isearch – software for indexing and searching text documents, ifile – A general mail filtering system, Bow: A Toolkit for Statistical Language Modeling, Text Retrieval, Classification and Clustering), Machine Learning Tools (e.g. Machine Learning Toolbox (MLT), The Machine Learning Programs Repository), FSA Tools (e.g. FSA Utilities: A Toolbox to Manipulate Finite-state Automata), HMM Tools (e.g. Hidden Markov Model (HMM) Toolbox, Discrete HMM Toolkit, A HMM mini-toolkit), Language Modeling Tools (e.g. Maximum Entropy Modeling Toolkit, Trigger Toolkit, Language modeling tools), Corpus Tools ( e.g. WebCorp, Multext: i.e. Multilingual Text Tools and Corpora, TACT- i.e. Text Analysis Computing Tools, Textual Corpora and Tools for their Exploration). Some more tools include DR-LINK (Document Retrieval using LINguistic Knowledge) system demonstrating the capabilities of NLP for Information Retrieval [Liddy et al, 2000], NLPWin: an NLP system from Microsoft that accepts sentences and delivers detailed syntactic analysis, together with a logical form representing an abstraction of the meaning.
[Elworthy, 2000]. Waldrop (2001) has described the features of three NLP software packages, viz. Jupiter: a product of the MIT research Lab that works in the field of weather forecast, Movieline: a product of Carnegie Mellon that talks about local movie schedules, and MindNet from Microsoft Research, a system for automatically extracting a massively hyperlinked web of concepts.

2.2. Review of Research Work in Machine Learning

Machine learning is a vast field and there has been a lot of research in this area. Here we discuss the literature relevant to our thesis. Machine learning studies algorithms capable of improving their performance automatically when provided with additional knowledge regarding the specified domain. As discussed earlier, successful use of machine learning techniques depends on availability of sufficient quantities of labeled data. However, obtaining a large labeled data set becomes very expensive, particularly for the complex real-world tasks where machine learning techniques are most useful. As stated, active learning provides a way to reduce the labeling costs by labeling only the most useful instances for learning. The learning algorithm selects only those instances for annotation that are required to learn an accurate classifier [Cohn et al., 1994]. Hence active learning algorithms provide much higher accuracy rates using small number of labeled examples and selecting the data from which it learns. An active learner can ask different queries in the form of unlabeled examples that are to be labeled by a human annotator. A lot of research has been carried out in this field, therefore we will describe these works along several dimensions.

2.2.1. Active Learning Scenarios

There are different circumstances in which the learner may ask queries. The learner may construct their own examples (membership query synthesis), request certain types of examples (pool-based sampling), or determine which of the unlabeled examples to query and which to discard (selective sampling). These different scenarios also determine the different sources from which the unlabeled instances are presented for annotation.
2.2.1.1. Membership Query Synthesis

In the membership query synthesis [Angluin, 1988], the learner may construct its own examples i.e. the learner may ask for labels for any unlabeled example in the input space. It also includes the queries that the learner generates anew, rather than the ones that are sampled from some underlying distribution. Query synthesis has been shown to be efficient for finite problem domains [Angluin, 2001]. It has also been extended to regression learning tasks, for example learning to predict the absolute coordinates of a robot hand [Cohn et al., 1996].

In many situations query synthesis has been used efficiently however it has some disadvantages too. One of the drawbacks is that the labeling of such random instances cannot be easy if human annotator does the annotations. For example, Baum and Lang (1992) used membership query learning along with human annotators oracles for training a neural network to classify handwritten characters. They had to face an unexpected problem: most of the query images that the learner generated contained no meaningful and recognizable symbols. They only consisted of artificial characters that were meaningless. Therefore, membership query synthesis for natural language processing tasks creates meaningless streams of text or speech that are nothing more than garbage. This method usually generates meaningless examples which are hard to label as the learner is able to request a label from any possible instance from the input space and ignores the underlying sample distribution. The stream-based and pool-based scenarios have been developed to solve the above mentioned limitations. Systems using membership query syntheses have been implemented practically [King et al., 2004]. In these systems an application of the membership query synthesis has been described in which a robot scientist has been shown executing a series of experiments in order to discover pathways of metabolism in yeast. In this application, a mixture of chemical solutions can be regarded as an instance and a label can be whether or not the mutant thrived in the growth medium. All experiments have been carried out autonomously using active machine learning, and physically carried out using a robot. This method reduced the experimental costs by three-fold as compared to when the least expensive experiment is run, and resulted in a 100-fold decrease in cost compared to randomly generated experiments.
2.2.1.2. Stream-based Selection/Selective sampling

Selective sampling [Cohn et al., 1994] is another active learning scenario which can be regarded as an alternative to membership query synthesis. In this scenario the instances are presented to the learner from an infinite source of unlabeled data. The learner performs the sampling of an unlabeled instance from the actual distribution as its free (or inexpensive), and then decides whether it should pay the cost of labeling it or not. This scenario is also known as stream-based or sequential active learning, because of the fact that an unlabeled instance is drawn one at a time from the data stream, and the learner has to decide whether to query or discard it. The main point on which pool-based and stream-based active learning differ is that the whole stream cannot be observed during each round of active learning, and hence limiting the protocol as the learner is able to examine each example in a stream only once during the life span of the learner and it is suitable for many applications such as speech recognition. For uniform distribution of input, this technique behaves similar to membership query learning. However, for non-uniform distribution or unknown distribution, it is certain that queries will still be meaningful, as they come from a real underlying distribution.

There are several ways by which the decision of whether to label an instance or not can be framed. One way of determining this is to evaluate the samples using some “informativeness measure” or “query strategy” and taking a random decision, so that more informative instances are more likely to be queried [Dagan and Engelson, 1995]. In another way a region of uncertainty is found [Cohn et al., 1994], i.e. finding that explicit part of the instance space which is ambiguous to the learner, and then only querying the instances which fall within this region. One way of doing this is determining a minimum threshold of an informativeness measure which defines the region and query those instances whose evaluation is above this threshold. Another more principled approach is to define the region that is still unknown to the overall model class, i.e., to the set of hypotheses consistent with the current labeled training set called the version space [Mitchell, 1982]. In other words, if any two models of the same model class (but different parameter settings) agree on all the labeled data, but disagree on some unlabeled sample, then that sample lies within the region of uncertainty. The complete and explicit calculation of this region is very expensive computationally and it must be maintained after each new query. This is the reason why approximations are used in practice [Cohn et al., 1994; Dasgupta et al., 2008].
The stream-based scenario has been used in many practical problems, including part-of-speech tagging [Dagan and Engelson, 1995], sensor scheduling [Krishnamurthy, 2002], and learning ranking functions for information retrieval [Yu, 2005]. Fujii et al. (1998) employ selective sampling in active learning for word sense disambiguation, e.g., determining if the word “bank” means land alongside a river or a financial institution in a given context (only they study Japanese words in their work). The approach not only reduces annotation effort, but also limits the size of the database used in nearest-neighbor learning, which in turn expedites the classification algorithm.

2.2.1.3. Pool-based Selection

Pool-based scenario [Lewis and Gale, 1994] of active learning is based on the assumption that a small set of labeled data L and a large pool of unlabeled data U are available. During the process of active learning, an unlabeled instance is selected by the querying function Q from the unlabeled pool. The pool is assumed to be static i.e. non-changing also called closed. The querying of instances takes place according to informativeness measure in a greedy fashion. Then the annotation of the queried instance is done and the instance is then added to the set of labeled data for the purpose of training. In pool-based active learning techniques a querying function is used for scoring each instance \( x \in U \) according to their informativeness. These techniques then use this score for ranking the unlabeled elements, and finally selects the highest ranked instances.

The real world problems of machine learning for which the pool-based active learning techniques have been studied include text classification [Lewis and Gale, 1994; McCallum and Nigam, 1998b; Tong and Koller, 2001; Hoi et al., 2006a], information extraction [Thompson et al., 1999; Settles and Craven, 2008], image classification and retrieval [Tong and Chang, 2001; Zhang and Chen, 2002], video classification and retrieval [Yan et al., 2003; Hauptmann et al., 2006], speech recognition [Turet al., 2005], and cancer diagnosis [Liu, 2004] to name a few. There is a difference between stream-based and pool-based active learning. In the stream based learning the data is scanned sequentially and the query decisions are made individually. In pool based learning the entire collection is evaluated and ranked before selecting the best query.
2.2.2. Querying Strategies

The main aspect of all active learning strategies is the design of an appropriate querying function, which uses the current state of the learner and properties of the available data to select unlabeled examples for annotation. The querying function evaluates the informativeness of unlabeled instances, which can either be generated de novo or sampled from a given distribution. There have been many proposed ways of designing a good querying function. Some of them are surveyed below.

2.2.2.1. Uncertainty Sampling

Uncertainty sampling [Lewis and Gale, 1994] is the simplest and most widely used query framework where the learner selects instances for which its prediction is most uncertain i.e. about which it is least confident how to label. This approach is often straightforward for probabilistic learning models. For example, when using a probabilistic model for binary classification, an uncertainty sampling strategy simply queries the instance whose posterior probability of being positive is nearest 0.5 [Lewis and Gale, 1994; Lewis and Catlett, 1994]. For many learning algorithms, a widely used method of uncertainty sampling is to select instances for which their predicted label is least confident, either from a probabilistic viewpoint or through a margin-based analogue [Lewis and Gale, 1994; Tong and Koller, 2001; Schohn and Cohn, 2000; Culotta and McCallum, 2005; Roth and Small, 2006b; Settles and Craven, 2008].

A more general uncertainty sampling strategy uses entropy [Shannon, 1948] as an uncertainty measure:

$$\Phi^{\text{ENT}}(x) = -\Sigma P(y_i|x) \log P(y_i|x),$$

where $\Phi$ represents a query strategy, which is a function used to evaluate the informativeness of a query, $x$ represents the best query instance which maximizes this function, and $y_i$ ranges over all possible labeling. The entropy-based approach can be generalized easily to probabilistic multi-label classifiers and probabilistic models for more complex structured instances, such as sequences [Settles and Craven, 2008] and trees [Hwa, 2004]. An alternative to entropy in these more complex settings involves querying the instance whose best labeling is the least confident:

$$\Phi^{\text{LC}}(x) = 1 - P(y^*|x),$$

where $y^* = \arg\max P(y|x)$ is the most likely class labeling. This sort of strategy has been shown to work well, for example, with conditional random fields or CRFs [Lafferty et al., 2001] for
active learning in information extraction tasks [Culotta and McCallum, 2005; Settles and Craven, 2008]. Uncertainty sampling strategies may also be employed with non-probabilistic models. One of the first works to explore uncertainty sampling used a decision tree classifier [Lewis and Catlett, 1994] by modifying it to have probabilistic output. Similar approaches have been applied to active learning with nearest-neighbor (“memory-based” or “instance-based”) classifiers [Fujii et al., 1998; Lindenbaum et al., 2004], by allowing each neighbor to vote on the class label of x, with the proportion of these votes representing the posterior label probability. Tong and Koller (2000) also experiment with an uncertainty sampling strategy for support vector machines, or SVMs [Cortes and Vapnik, 1995], that involves querying the instance closest to the linear decision boundary. This last approach is analogous to uncertainty sampling with a probabilistic binary linear classifier, such as logistic regression or naive Bayes [Kosmopoulos et al., 2008].

2.2.2.2. Query-By-Committee
The query-by-committee (QBC) framework [Seung et al., 1992; Freund et al., 1997; Fine et al., 2002] is similar to uncertainty sampling, but is distinguished by using an ensemble of experts to select instances for annotation. In QBC, a committee of learned models is trained using the labeled data and a querying function is derived through a voting mechanism. The QBC approach involves maintaining a committee C of models which are all trained on the current labeled set L, but represent competing hypotheses. Each committee member is then allowed to vote on the labelings of query candidates. The most informative query is considered to be the instance about which they most disagree. The basic principle of QBC approach is to minimize the version space. Version space is the set of hypotheses that are consistent with the current labeled training data L. If machine learning is considered as the search for the best model within the version space, then the aim of active learning is to limit the size of this space as much as possible with as few labeled instances as possible in order to make the search more precise. QBC does exactly this by querying in controversial regions of the version space.

Two things are necessary in a QBC framework, one is to construct a committee of models that approximate different regions of the version space and the other is to have some measure of disagreement among them. Seung et al. (1992) accomplish the first task simply by sampling a committee of two random hypotheses that are consistent with L. For generative model classes, this can be done more generally by randomly sampling models from some posterior distribution.
P(θ|L). For example, McCallum and Nigam (1998b) do this for naive Bayes by using the Dirichlet distribution over model parameters, whereas Dagan and Engelson (1995) sample HMMs by using the Normal distribution. For other model classes, such as discriminative or non-probabilistic models, Abe and Mamitsuka (1998) have proposed query-by-boosting and query-by-bagging, which employ the well-known ensemble learning methods boosting [Freund and Schapire, 1997] and bagging [Breiman, 1996] to construct committees. Melville and Mooney (2004) propose another ensemble-based method which encourages diversity among committee members. For measuring the degree of disagreement, two main approaches have been proposed: vote entropy [Dagan and Engelson, 1995] and average KL-divergence [McCallum and Nigam, 1998b]. There is no consensus on the appropriate committee size to use, which may in fact vary by model class or application. However, even small committee sizes (e.g., two or three) have been shown to work well in practice [Seung et al., 1992; McCallum and Nigam, 1998b; Settles and Craven, 2008]. Aside from the QBC framework, several other query strategies attempt to minimize the version space as well. For example, Cohn et al. (1994) describe a related selective sampling algorithm for neural networks using a combination of the “most specific” and “most general” models, which lie at two extremes the version space given the current labeled examples in the training set L. Tong and Koller (2000) propose a pool-based query strategy that tries to minimize the version space for support vector machine classifiers directly. The membership query algorithms of Angluin (1988) and King et al. (2004) can also be interpreted as synthesizing de novo instances that limit the size of the version space. However, Haussler (1994) shows that the size of the version space can grow exponentially with the size of L. This means that, in general, the version space of an arbitrary model class cannot be explicitly represented in practice. The QBC framework, rather, uses a committee which is a subset-approximation of the full version space.

2.2.2.3. Unreliability Sampling

Another recently developed strategy for designing a querying function is unreliability sampling [Becker, 2008]. The basic premise of this framework is that instances should be selected which have parameters which have not observed sufficient data for confident estimation. An early instantiation of this method was active learning for syntactic parsing, where unlabeled instances which cause the current parsing model to fail are used to request labels from the expert.
[Thompson et al., 1999]. Following the same basic principles, this paradigm has been extended for improvements in active learning for syntactic parsing [Becker and Osborne, 2005] and active learning for machine translation [Haffari et al., 2009]. Recent work on confidence-weighted active learning [Dredze and Crammer, 2008] applies a similar philosophy by selecting examples with parameters possessing high variance during estimation. As opposed to uncertainty sampling, which selects examples for which the prediction has low confidence, unreliability sampling selects those instance for which an accurate measure of certainty cannot be computed.

2.2.2.4. Expected Model Change
A much more recently formalized approach for designing a querying function is to select instances which exhibit the greatest expected model change [Settles and Craven, 2008] i.e. that would impart the greatest change to the current model if we knew its label. As opposed to selecting instances for which the learner is least confident, the expected model change selects instance for which there is an expectation of significant change in between the current hypothesis and the resulting induced hypothesis if the instance was labeled. This strategy was noted earlier in the context of selecting instances for learning an SVM [Bordes et al., 2005], but without an accurate estimate of model change, they relied on a margin-based uncertainty method. The intuition behind this framework is that those instances will be preferred that are likely to most influence the model (i.e., have greatest impact on its parameters), regardless of the resulting query label. This approach has been shown to work well in empirical studies, but can be computationally expensive if both the feature space and set of labelings are very large.

2.2.2.5. Estimated Error Reduction
A traditionally less popular strategy gaining increasing attention is the use of querying functions which attempt to directly minimize the generalization error. Under this framework, each instance is scored with respect to the expected reduction in future error if labeled and added to the training data. This method is theoretically appealing as it attempts to directly minimize error, the true task at hand. Although shown to be empirically effective, the drawback to querying by expected error reduction is the computation required to estimate expected error and compute an updated model for each possible labeling for each unlabeled instance. However, this approach has been shown very successful when methods such as sub sampling the unlabeled pool with a
naive Bayes classifier [Roy and McCallum, 2001], exact incremental updates with Gaussian random fields [Zhu et al., 2003], and approximate training methods with logistic regression [Guo and Greiner, 2007].

Unfortunately, estimated error reduction may also be the most prohibitively expensive query selection framework. Not only does it require estimating the expected future error over $U$ for each query, but 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 dramatic increase in computational cost. For some model classes such as Gaussian random fields [Zhu et al., 2003], the incremental training procedure is efficient and exact thus making this approach fairly practical. For a many other model classes, this is not the case.

A statistically well motivated querying function strategy is selecting instances which minimize variance [Cohn et al., 1996]. Given the observation that expected generalization error can be decomposed into bias and variance components [Geman et al., 1992], the variance minimization strategy is to select instances for which once labeled and added to the training data will result in the greatest reduction in variance and thus generalization error. As this approach is only feasible for definitions of variance which are smooth and differentiable, it has only been applied to problems such as regression and neural networks [Cohn et al., 1996]. Related and more appropriate for the standard active learning settings is selection based upon the Fischer information associated with a prediction [Zhang and Oles, 2000; Hoi et al., 2006; Settles and Craven, 2008], which also require approximation techniques to calculate efficiently.

### 2.2.2.6. Density-Weighting Methods

One unfortunate property of many active learning querying functions is that they are relatively noise intolerant, motivating the study of techniques which weigh instances by how representative they are of the input distribution of the data, referred to as density-weighted querying functions. Pre-clustering the data and selecting examples which represent each cluster has been demonstrated a very successful for querying representative instances [Nguyen and Smeulders, 2004; Donmez et al., 2007; Xue et al., 2007]. These methods are particularly beneficial when learning from only a few instances, which is done early in the active learning process. Density-weighting formulations have also been studied for query-by-committee [McCallum and Nigam, 1998b] and in the context of sequence prediction [Settles and Craven, 2008]. The main idea is
that informative instances should not only be those which are uncertain, but also those which are “representative” of the input distribution (i.e., inhabit dense regions of the input space). Fujii et al. (1998) explored a query strategy for nearest-neighbor methods that selects queries that are unlike the labeled instances already in L, and most similar to the unlabeled instances in U.

2.2.3. Structured Outputs

Several important learning problems involve predicting structured outputs on instances, such as sequences and trees. In these problems multiple local predictions must be combined to form a coherent structure. These models have garnered significant interest in the NLP and other application communities as they can effectively incorporate information from multiple sources regarding many interdependent prediction tasks. As structured output labels are generally more expensive to obtain, there has been a corresponding interest in reducing labeling requirements in these settings. In the context of active learning, there has been some recent work regarding learning in structured output spaces including work on active learning for HMMs [Dagan and Engelson, 1995; Scheffer and Wrobel, 2001; Anderson and Moore, 2005], CRFs [Culotta and McCallum, 2005; Settles and Craven, 2008] and structured Perceptron [Roth and Small, 2006b]. More application targeted includes active learning for probabilistic context free grammars (PCFGs) [Baldridge and Osborne, 2004; Hwa, 2004]. Also, closely related works for settings more complex than binary classification include active learning for multiclass classification [Yan et al., 2003; Brinker, 2004] and active learning for ranking data [Brinker, 2004; Donmez and Carbonell, 2008].

Active learning, most notably pool-based selection, has been applied to many NLP applications including text/spam classification [Lewis and Gale, 1994; Liere and Tadepalli, 1997; McCallum and Nigam, 1998a; Schohn and Cohn, 2000; Tong and Koller, 2001; Hoi et al., 2006a; Schein and Ungar, 2007; Dredze and Crammer, 2008; Zhu et al., 2008a], chunking [Ngai and Yarowsky, 2000], part of speech tagging [Dagan and Engelson, 1995], named entity recognition [Scheffer and Wrobel, 2001; Shen et al., 2004; Becker et al., 2005; Jones, 2005; Kim et al., 2006; Vlachos, 2006; Tomanek et al., 2007; Laws and Schutze, 2008], information extraction [Thompson et al., 1999; Scheffer et al., 2001; Finn and Kushmerick, 2003; Jones et al., 2003; Culotta and McCallum, 2005; Culotta et al., 2006; Roth and Small, 2008; Settles and Craven, 2008], prepositional phrase attachment [Hwa, 2004; Becker, 2008], syntactic parsing [Thompson et al.,
word sense disambiguation [Chen et al., 2006; Chan and Ng 2007; Zhu and Hovy, 2007], semantic role labeling (Roth and Small, 2006b) and machine translation [Haffari et al., 2009; Haffari and Sarkar, 2009].

A framework and objective functions have been introduced for active learning in three fundamental HMM problems: model learning, state estimation, and path estimation. In addition, a new set of algorithms has been described for efficiently finding optimal greedy queries using these objective functions. The algorithms are fast, i.e., linear in the number of time steps to select the optimal query and we present empirical results showing that these algorithms can significantly reduce the need for labelled training data [Anderson and Moore, 2005].

Many classification problems with structured outputs can be regarded as a set of interrelated sub-problems where constraints dictate valid variable assignments. The standard approaches to these problems include either independent learning of individual classifiers for each of the sub-problems or joint learning of the entire set of classifiers with the constraints enforced during learning. An intermediate approach has been proposed where these classifiers are learnt in a sequence using previously learned classifiers to guide learning of the next classifier by enforcing constraints between their outputs. A theoretical motivation has been provided to explain why this learning protocol is expected to outperform both alternatives when individual problems have different `complexity'. This analysis motivates an algorithm for choosing a preferred order of classifier learning. This technique has been evaluated on artificial experiments and on the entity and relation identification problem where the proposed method outperforms both joint and independent learning. [Bunescu, 2008].

The success of interactive machine learning systems depends both on the machine and on the human performance. An understanding of machine capabilities and limitations should inform interaction design, while the abilities, preferences, and limitations of human operators should inform the choice of inputs, outputs, and performance requirements of machine learning algorithms. A relevant example from the past work is Arnauld system [Krzysztof and Daniel, 2005] for active preference elicitation. A lot of previous work in that area solicited user feedback in the form numerical ratings over possible outcomes. However, unless the rating scale is well grounded, people tend to be inconsistent and unreliable providing this type of feedback. What works much more robustly is pairwise comparison queries, where the person only has to state which of two possible outcomes he or she prefers [Krzysztof and Daniel, 2005]. Adopting this
input interaction, however, requires developing a new learning algorithm. In turn, to account for the limitations of the algorithm, the example critiquing interaction [Pearl and Chen, 2009] has been implemented to allow people to manually direct the learning once the active learning process no longer resulted in rapid improvements in the model quality. Work has been done on incorporating richer user feedback into interactive machine learning systems. Typically, machine learning algorithms only solicit labels from the users but several projects e.g. [Gregory et al., 2007] have shown that incorporating richer feedback—that captures the user's rationale—leads to faster and more generalizable learning. So far, this feedback has been limited to feature relevance. Is this the best or the only type of rich feedback that can be elicited from users? A preliminary study has been conducted in the context of preference elicitation for an e-commerce application to understand what types of feedback people naturally provide, and what the value of these different types of feedback might have for the speed and quality of learning. Specifically, users were asked to answer a set of pairwise comparison questions regarding digital cameras and their choices has been recorded as well as free form explanations of their choices.

End-user interactive concept learning is a technique for interacting with large unstructured datasets, requiring insights from both human-computer interaction and machine learning. This note re-examines an assumption implicit in prior interactive machine learning research i.e. interaction should focus on the question “what class is this object?” Amershi, S. et al (2010) have broadened interaction to include examination of multiple potential models while training a machine learning system. They evaluated this approach and found that people naturally adopted revision in the interactive machine learning process and that this improved the quality of their resulting models for difficult concepts.

M. Kristan et al (2009) have proposed a Gaussian-kernel-based online kernel density estimation which can be used for applications of online probability density estimation and online learning. This approach generates a Gaussian mixture model of the observed data and allows online adaptation from positive examples as well as from the negative examples. The adaptation from the negative examples is realized by a novel concept of unlearning in mixture models. Low complexity of the mixtures is maintained through a novel compression algorithm. In contrast to other approaches, this approach does not require fine-tuning parameters for a specific application, they have not assumed specific forms of the target distributions and temporal constraints have not been assumed on the observed data. The strength of the proposed approach
Very recently there has been work on actively selecting examples with the intention of labeling properties regarding features. The earliest example of this work is the tandem learning algorithm where the expert iteratively queries the expert for instance labels and then feature labels. This idea of labeling both instances and features simultaneously has been further pursued in the active dual supervision model [Sindhwani et al., 2009]. Even more recently, the generalized expectation criteria has been incorporated into the active learning framework to present instances to the domain expert for the explicit purpose of incorporating domain knowledge by labeling features [Druck et al., 2009]. The learning from measurements model [Liang et al., 2009] also works along this vein by deriving a framework based on Bayesian experimental design to select instances for which the largest expected information gain will be achieved if the feature is labeled.

In most of the active learning research, queries are selected in serial, i.e., one at a time. However, sometimes the training time required to induce a model is slow or expensive, as with large ensemble methods and many structured prediction tasks. Consider also that sometimes a distributed, parallel labeling environment may be available, e.g., multiple annotators working on different machines at the same time on a network. In both of these cases, selecting queries in serial may be inefficient. By contrast, batch-mode active learning allows the learner to query instances in groups, which is better suited to parallel labeling environments or models with slow training procedures.

Myopically querying the “N-best” queries according to a given instance-level query strategy often does not work well, since it fails to consider the overlap in information content among the “best” instances. To address this, a few batch-mode active learning algorithms have been proposed. Brinker (2003) considers an approach for SVMs that explicitly incorporates diversity among instances in the batch. Xu et al. (2007) propose a similar approach for SVM active learning, which also incorporates a density measure. Specifically, they query cluster centroids for instances that lie close to the decision boundary. Hoi et al. (2006a,b) extend the Fisher information framework to the batch-mode setting for binary logistic regression. Most of these approaches use greedy heuristics to ensure that instances in the batch are both diverse and informative, although Hoi et al. (2006b) exploit the properties of submodular functions to find
near-optimal batches. Alternatively, Guo and Schuurmans (2008) treat batch construction for logistic regression as a discriminative optimization problem, and attempt to construct the most informative batch directly. For the most part, these approaches show improvements over random batch sampling, which in turn is generally better than simple “N-best” batch construction.

In some learning problems, the cost of acquiring labeled data can vary from one instance to the next. If our goal in active learning is to minimize the overall cost of training an accurate model, then reducing the number of labeled instances does not necessarily guarantee a reduction in overall labeling cost. One proposed approach for reducing annotation effort in active learning involves using the current trained model to assist in the labeling of query instances by pre-labeling them in structured learning tasks like parsing [Baldridge and Osborne, 2004] or information extraction [Culotta and McCallum, 2005]. However, such methods do not actually represent or reason about labeling costs. Instead, they attempt to reduce cost indirectly by minimizing the number of annotation actions required for a query that has already been selected.

Another group of cost-sensitive active learning approaches explicitly accounts for varying label costs in active learning. Kapoor et al. (2007) propose one approach that takes into account both labeling costs and estimated misclassification costs. In this setting, each candidate query is evaluated by summing the labeling cost for the instance and the expected future misclassification costs that would be incurred if the instance were added to the training set. Instead of using real costs, however, their experiments make the simplifying assumption that the cost of labeling a voice mail message is a linear function of its length (e.g., ten cents per second). King et al. (2004) use a similar active learning approach in an attempt to reduce actual labeling costs. They describe a “robot scientist” which can execute a series of autonomous biological experiments to discover metabolic pathways, with the objective of minimizing the cost of materials used (i.e., the cost of an experiment plus the expected total cost of future experiments until the correct hypothesis is found).

As previously stated, the primary research issue for active learning is the design of an appropriate querying function. However, it is possible that different querying functions work better for different regions of the active learning cycle. For example, a querying function using density-weighted selection is very helpful for initial queries, but uncertainty sampling is more effective once the classifier is relatively stable [Donmez et al., 2007]. Baram et al. (2004) examine scenarios where several querying functions are employed by being cast in the multi-
armed bandit framework, where querying functions are selected which explicitly follow an exploration and exploitation cycles. In addition to selecting appropriate querying functions for different operating regions, as the overall goal of active learning is to reduce total annotation, it is also useful to know when maximal performance is achieved such that unnecessary actions will be avoided, referred to as a stopping criterion [Schohn and Cohn, 2000; Campbell et al., 2000; Tomanek et al., 2007; Vlachos, 2008; Dimitrakakis and Savu-Krohn, 2008; Laws and Schutze, 2008; Zhu et al., 2008a,b]. The critical aspect of deriving a stopping criterion is a method for autonomously determining the performance of the current learner hypothesis (i.e. without development or testing data). Other works have used a self-estimated measure of active learning performance to determine different operating regions which require different querying functions to be most effective [Baram et al., 2004; Donmez et al., 2007; Roth and Small, 2008].