Interactive Machine Learning with Applications in Health Informatics

by

Yue Wang

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in the University of Michigan
2018

Doctoral Committee:

Associate Professor Qiaozhu Mei, Chair
Associate Professor Kevyn Collins-Thompson
Assistant Professor Jia Deng
Assistant Professor Walter S. Lasecki
Associate Professor Kai Zheng, University of California, Irvine

Yue Wang

raywang@umich.edu

ORCID iD: 0000-0002-0278-2347

To Tong

To my parents

To my first teacher and grandmother, Xiangzhu Ma

Acknowledgments

First and foremost, I would like to express my sincerest gratitude to my advisor, Prof. Qiaozhu Mei. During my PhD study, Qiaozhu has been a tremendous source of wisdom, enthusiasm, inspiration, and support to me. I learned a great deal from Qiaozhu, from formulating research problems to steering through the maze and trenches of hypotheses; from the pursuit of principled methods to the passion in real-world applications; from nurturing collaboration relationship to mentoring junior students. The continuous freedom, trust, and encouragement that Qiaozhu has given me over the years is perhaps the most luxurious gift that a PhD student can get in the career – a gift that I relish and value so much in retrospection.

Besides my advisor, I would also like to thank other members of my committee: Prof. Kevyn Collins-Thompson, Prof. Jia Deng, Prof. Walter Lasecki, and Prof. Kai Zheng, for their broad perspectives, insightful comments, and deep questions. This dissertation would not have been completed without their careful review and valuable feedback. I am especially indebted to Prof. Kai Zheng for introducing the exciting field of health informatics to me, for connecting me to the real needs of health domain experts, for tirelessly dedicating his precious time in critiquing and revising my work, and for his continued support to my study and career.

The University of Michigan School of Information has been an intellectual home to me. Its people-centered mission and interdisciplinary atmosphere has shaped my research taste. I benefited from interacting with many faculty members at UMSI. I should especially thank Prof. Paul Resnick for his support to my career and his many witty questions in the ReQ-ReC project. If Paul hadn't challenged me to find a good explanation for the effectiveness of ReQ-ReC, I wouldn't have studied the theories of

machine learning and wouldn't have conceived the last chapter of this dissertation. Also I want to thank Prof. Ceren Budak, Prof. Daniel Romero, and Prof. David Jurgens for their constructive feedback to my work in various IAR seminars.

It is a great honor to work with my collaborators in health informatics. I want to thank Prof. Hua Xu for introducing to me the intriguing frontiers of medical natural language processing, Prof. Joyce Lee and Prof. V.G.Vinod Vydiswaran for their advice and domain expertise on patient-centered content analysis, Prof. Yunan Chen, Dr. Xinning Gui, and Dr. Yubo Kou for teaching me not just new research methodologies, but the value of a great team in a collaboration.

I want to thank my internship mentors as I am deeply influenced by their passion in solving large-scale real-world problems: Dr. Yi Chang and Dr. Dawei Yin at former Yahoo Labs; Dr. Paul Bennett, Dr. Ryen White, Dr. Eric Horvitz, Dr. Milad Shokouhi, Dr. Jin Young Kim, and Dr. Nick Craswell at Microsoft Research. I also want to thank my mentors Dr. Yan Xu and Dr. Eric Chang at Microsoft Research Asia; thank you for sparking my initial interest in text mining and machine learning, which led me onto the PhD journey.

I would like to thank my past and present comrades in the Foreseer research group for their fruitful discussions and collaborations all over the years: Yang Liu, Danny Tzu-Yu Wu, Jian Tang, Zhe Zhao, Xin Rong, Cheng Li, Sam Carton, Wei Ai, Shiyan Yan, Teng Ye, Tera Reynolds, Jiaqi Ma, Cristina Garbacea, Zhuofeng Wu, Huoran Li, Yunhao Jiao, Xuedong Li, and Sui Li. I also want to acknowledge the brilliant visiting students whom I was very fortunate to work with, for their tremendous help in my research: Ning Jiang, Qifei Dong, Jiatong Li, Gaole Meng, and Shengyi Qian.

I have received help from many friends at University of Michigan. I would like to thank my friends at both UMSI and CSE who shared laughters and braved hardships with me along my PhD journey: Daniel Xiaodan Zhou, Tao Dong, Melody Ku, Carrie Wenjing Xu, Zhe Chen, Huan Feng, Youyang Hou, Xuan Zhao, Matthew Burgess,

Daphne Chang, Ark Fangzhou Zhang, Yingzhi Liang, Yan Chen, Shiqing He, Ryan Burton, Rohail Syed, Yuncheng Shen, Fengmin Hu, Linfeng Li, Xinyan Zhao, Hao Peng, Ed Platt, Danaja Maldeniya, Tawfiq Ammari, Ruihan Wang, and my five-year roommate, Xianzheng Dou. Also I want to thank my old friends Chang Sun and Di Chen, who encouraged me to apply for the University of Michigan and shared much fun with me in Ann Arbor all these years.

Finally, I would like to thank my wife and my parents for their unfailing love and support. A special thanks to my maternal grandmother and first teacher, Xiangzhu Ma, who showed me the joy of learning when I was a child. I would not have made this far without their continuous encouragement. This dissertation is dedicated to all of them.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Recent years have witnessed unprecedented growth of health data, including millions of biomedical research publications, electronic health records, patient discussions on health forums and social media, fitness tracker trajectories, and genome sequences. Information retrieval and machine learning techniques are powerful tools to unlock invaluable knowledge in these data, yet they need to be guided by human experts. Unlike training machine learning models in other domains, labeling and analyzing health data requires highly specialized expertise, and the time of medical experts is extremely limited. How can we mine big health data with little expert effort? In this dissertation, I develop state-of-the-art interactive machine learning algorithms that bring together human intelligence and machine intelligence in health data mining tasks. By making efficient use of human expert's domain knowledge, we can achieve high-quality solutions with minimal manual effort.

I first introduce a high-recall information retrieval framework that helps human users efficiently harvest not just one but as many relevant documents as possible from a searchable corpus. This is a common need in professional search scenarios such as medical search and literature review. Then I develop two interactive machine learning algorithms that leverage human expert's domain knowledge to combat the curse of "cold start" in active learning, with applications in clinical natural language processing. A consistent empirical observation is that the overall learning process can be reliably accelerated by a knowledge-driven "warm start", followed by machine-initiated active learning. As a theoretical contribution, I propose a general framework for interactive machine learning. Under this framework, a unified optimization objective explains many existing algorithms used in practice, and inspires the design of new algorithms.

CHAPTER 1

Introduction

Machine learning systems, especially deep learning systems in recent years, have achieved major breakthroughs in several research frontiers, including computer vision, speech recognition, machine translation, and board game playing. In restricted settings like trivia question-answering[1], the board game Go[2], and speech-to-text transcription for major languages [149], machine learning systems have been demonstrating impressive performance, on par with or even superior to human experts.

These recent successes have kindled enormous interest in applying machine learning techniques to solve a wide range of real-world problems. Private and public sectors like manufacturing, logistics and supply chain, marketing and customer relations, financial investments, health care, public transportation, law enforcement, and education are all pursuing machine learning approaches to augment and even revolutionize their traditional practices. Machine learning has been an increasingly popular tool for analyzing data and harvesting knowledge in scientific fields outside of computer science, including medicine, biology, astronomy, material science, communication studies, digital humanities, economics, and business. Our everyday life is surrounded by a variety of intelligent services and devices with machine learning capabilities, such as search engines, social network services, online retailing websites, intelligent personal assistants, fitness trackers, self-driving cars, and small autonomous aircrafts. Machine learning

[1] `https://en.wikipedia.org/wiki/Watson_(computer)`
[2] `https://en.wikipedia.org/wiki/AlphaGo`

is being integrated into the work and life of people from an extensive array of backgrounds, not limited to those trained in computer science.

If we can build a computer Go program that outsmarts the best human Go player, then can we build an intelligent health care program that is superior to the best human doctors? In other words, *can we replicate the recent successes of machine learning systems in critical domains, such as health care?* This is a natural question as machine learning is quickly expanding its application frontiers in recent years. We hear different opinions on this question. Prof. Geoffrey Hinton believes that artificial intelligence will replace radiologists in the next five to ten years[3]. However, at the same time, M.D. Anderson Cancer Center dissolved their contract with IBM Watson Health[4]. Physicians can never adopt AlphaGo's relentless try-and-error approach in treating human patients.

Why can't the same successes of IBM Watson and AlphaGo easily happen in critical domains like health care? Behind all celebrated machine learning systems is one thing in common: a massive amount of training data. Training data take the form of question-answer pairs, or labeled examples, for the machine to learn from[5]. In computer vision, ImageNet contains over 10 million hand-annotated images, and Google's internal data set is at least one order of magnitude larger [124]. State-of-the-art speech recognition systems are trained on hundreds of hours of transcribed utterances. Machine translation data for pairs of major languages often contain millions of translated sentence pairs. Major search engines, such as Google and Bing, regularly hire thousands of content editors to judge the relevance of millions of query-URL pairs, in order to train and evaluate their search algorithms. Fortunately, the above annotation tasks can be performed by the general crowd, which are accessible through platforms such

[3] `https://www.youtube.com/watch?v=2HMPRXstSvQ`

[4] `https://www.technologyreview.com/s/607965/a-reality-check-for-ibms-ai-ambitions/`

[5] The training data for reinforcement learning algorithms are state-action-reward trajectories generated by the environment – real or simulated – in which the machine will operate. Generating such data is inexpensive in the case of AlphaGo (as the rules of *Go* are completely known), but can be very expensive for real applications such as conversational agents and autonomous vehicles.

as Amazon Mechanical Turk[6] and Figure Eight (formerly CrowdFlower)[7], and can be programmatically managed in real time [86] and with humans in the loop [113]. Crowdsourcing approaches have fueled the collection of large-scale training data for a variety of machine learning tasks, such as computer vision [30, 73], speech and natural language processing [129, 19, 81], and robotics [49].

However, stepping out of research laboratories into real-world scenarios, machine learning systems hardly get sufficient high-quality training data. Unlike research benchmark data sets or highly focused industrial products, it is impossible for every real-world machine learning task to afford a large amount of labeled examples for at least the following two reasons.

- **Scarce domain expertise**: labeling and analyzing data in many domains can be extremely time-consuming and requires highly specialized expertise. For instance, labeling medical text data requires dedicated time and attention of experienced physicians and nurses. In a text de-identification task [132], labeling only 310 clinical notes took a group of MIT medical researchers 568 annotator-hours. Unfortunately, these medical experts are in short supply and often occupied by more urgent duties than data annotation. Although rich knowledge exists in the experts' head and medical knowledge bases, most machine learning algorithms can only learn from question-answer pairs. Similar difficulties happen in professional scenarios such as legal, commercial, governmental, and academic domain tasks. Practitioners in these domains have pressing needs for processing increasingly large data sets, but they rarely have enough time for labeling examples, and their expertise is scarce. To make things even worse, data in these domains are often protected by privacy and security regulations, making it very hard to crowdsource the labeling tasks to other professionals. Therefore these domains are witnessing little adoption of machine learning techniques.

[6] `https://www.mturk.com/`
[7] `https://www.figure-eight.com/`

- **Diverse *ad hoc* tasks**: in their working context, practitioners often need to define and solve *ad hoc* machine learning tasks, where no historical training data exist. For example, a physician wants to identify and analyze a cohort of patients with similar symptoms of a new patient; a researcher needs to perform a comprehensive literature survey on a new research topic; a paralegal is assigned to retrieve all relevant pieces of evidence from an email collection for a new case; a social scientist who studies a group of social media users aims to code their online posts into an ad hoc set of semantic categories. In all the cases above, the user aims to sort a large number of documents into predefined categories, which should be a perfect task for machine learning classifiers. However, the high cost of labeling a large amount of training examples from scratch may discourage the practitioner to adopt a machine learning approach, as these tasks are often one-off.

How can we proceed in these scenarios, where we cannot have enough labeled data to train powerful machine learning models? This intriguing and pressing question has been calling for answers from both researchers and practitioners in recent years [2]. Many research directions are trying to solve this problem, including semi-supervised learning [167], weakly supervised learning [166], transfer learning [99], one-shot/zero-shot learning [41, 131], active learning [117], Bayesian optimization [122], and lifelong machine learning [25].

My attempt in answering this question starts with a reflection on the current relationship between the human and the machine in recent machine learning practices. In supervised machine learning, one often uses the metaphor that the human is the teacher and the machine is the student. If we take a closer look at the current "teacher-student" relationship between the human and the machine, then we can find it far from what we would expect. On the human teacher side, her only job is to create millions of training examples, each like an miniature exam, as the sole material to teach the

student. The machine student, on the other hand, spends all her time trying to get high scores on millions of "practice exams" (training examples), in the hope that she can achieve a high score in the "final exam" (test examples). Under this learning strategy, the student might indeed obtain a high score in the "final exam", but it is very inefficient in terms of teaching efforts, or number of training examples. The teacher actually serves as a cheap labor who tirelessly provides question-answer pairs to the student. The situation is very similar to "rote learning", where learning largely relies on brutal force.

Human learning strategies, in contrast, are much more efficient. In the process of human learning, the teacher and the student interact and collaborate with one another. Not only the teacher can ask the student questions, the student can also raise questions back to the teacher. A good teacher does not just provide answers, but also explains why an answer should be as such. To teach a concept, the teacher will break down a whole instance into smaller parts, show its key attributes, and present typical examples as well as nonexamples. A good student is not just good at answering questions, but also proactive and curious in the learning process. She knows where her understanding is solid and where it is still vague, and is able to formulate good questions to explore unknown areas to resolve uncertainty. As such, an active student can grasp the essence of a concept without having to solve a large number of problems. In the ideal case, a student comes to the class knowing *how to learn*. The teacher starts by showing the definition (i.e., key attributes) of a concept and a few typical examples. The student then sets out gathering relevant material to study, comes back asking clarification questions, and quickly masters the concept after a few rounds of interaction.

What makes human learning so efficient? The reflection above reveals two distinctive features of human learning. First, instead of solely observing input-output pairs, a human teacher decomposes an entire example into named attributes or subconcepts, and directly teaches the students which attributes are essential and which should be

ignored. Explicitly passing on knowledge this way saves a good amount of student effort, as otherwise the student has to observe a large amount of examples to figure out which attributes are important. Second, instead of passively waiting for the teacher, a good human learner knows what she knows and actively seek for material to learn what she does not know yet. Using such a meta-learning strategy, the student maximizes the information gained from every question, thus increases the chance of learning in each interaction. The two aspects – knowledge and meta-learning strategy – reinforce one another. The more knowledge a student has, the better she knows her weakness on the subject, the more targeted and sensible her questions will be, and the faster her knowledge accrues. Such a learning strategy is the major inspiration behind this dissertation.

1.1 The Goal and Contribution of this Dissertation

As discussed above, learning is naturally an interactive and continuous process. This dissertation aims to design algorithms and study principles that embody this basic idea, which we call ***interactive machine learning***. Compared to the conventional supervised machine learning, an interactive machine learning algorithm has one or more of the following characteristics:

- It understands *diverse input modalities*, such as keywords, key attributes, contextual cues, logical rules, relative preferences, knowledge base entries, related data and models, and even natural language statements about the task. This allows the human teacher to flexibly express her knowledge to the learner, which maximizes the chance of learning. This is especially useful at the early stage of learning, where the teacher needs to endow the learner with as much prior knowledge as possible, so as to reduce the teaching effort later on. To realize this subgoal, the learner needs new input channels and internal representation

methods to consolidate and learn from different learning signals.

- It communicates in *diverse output modalities*, such as predicted labels, confidence scores, and decision explanations for individual examples; decision boundary, clustering structure, summary statistics, and visualizations for a set of examples; and even generated examples (synthesized or retrieved) pertaining to the task. This allows the machine to clearly expose its current understanding of the task and express its current doubts in various forms, which informs the human where to target her teaching effort. To realize this subgoal, the learner needs new output channels and presentation methods to communicate different learning outcomes.

- It is *proactive in the learning process*, not passively waiting for training materials to arrive. A typical strategy is active learning [117], where the machine learner chooses examples and asks the human teacher to label. With diverse input and output modalities, the algorithm can seek for and understand supervision signals in more flexible forms than labeled examples. In a broader sense, the machine is intelligent not only after the learning is done, but during the learning process. It is a meta-learner that knows *how to learn*, and may even adjust its meta-learning strategies to improve the learning outcome, a capability known as *learning to learn*.

The goal of this dissertation is to design, evaluate, and understand interactive machine learning algorithms that help human experts accomplish real-world data mining tasks with minimal teaching effort. On the application side, it aims to propose novel learning algorithms that delivers high-performance models using intuitive modes of interaction. On the theoretical side, it aims to discover the common principle underlying a variety of interactive machine learning algorithms, which can then inform the design of new ones.

The dissertation has a special focus on applications in the health domain. The spe-

cial nature of health domain creates a unique challenging scenario for machine learning methods, because (1) data in the health domain are often unstructured, heterogeneous, and large; (2) training data are in very limited supply as it is expensive and time-consuming for domain experts to label examples; (3) high-performance models are often required to ensure high-quality care; (4) the domain is known to have curated a wealth of knowledge, including systematic knowledge bases, ever-increasing literature, and physicians' rules of thumb. Therefore the goal of interactive machine learning algorithms for the health domain is to allow medical domain experts to efficiently train machine learning algorithms, with minimum supervision effort and maximum reuse of medical domain knowledge.

1.1.1 Summary of Contributions

This dissertation makes the following contributions to the fields of information retrieval and machine learning.

1. **A general framework for interactive high-recall retrieval**. This is a versatile framework that integrates the strengths of relevance feedback in information retrieval (IR) and active learning (AL) for classification (Chapter 3). From an IR perspective, it provides an effective algorithmic solution to high-recall retrieval, an important and hard problem in professional IR. From an AL perspective, it extends AL algorithms to scenarios where the data collection is only accessible via a search interface, which is often the case for very large data collections.

2. **Methods for warm-start active learning**. Active learning works best when the base learner already has decent performance. This is often not the case at the very beginning of the learning process, when few or no labeled examples are available, a problem known as "cold start". This dissertation proposes a suite of algorithms to "warm-start" the active learning process by leveraging

domain knowledge through diverse input channels (Chapter 4 and 5). Empirical experiments show that a warm start at the early stage and active learning in the later stage can reliably accelerate the overall learning process.

3. **A unified objective for interactive machine learning algorithms**. Many supervised learning algorithms can be explained by the structural risk minimization principle, yet there lacks a common principle that unifies the myriad of interactive machine learning algorithms. This dissertation proposes a unified optimization objective that explains a variety of interactive machine learning algorithms (Chapter 6). The unified objective not only enhances our understanding of existing interactive learning algorithms, but also informs the improvement of existing algorithms and the design of new ones.

1.2 Dissertation Outline

In Chapter 3, I describe a novel interactive high-recall retrieval framework, which we call ReQ-ReC. The goal is to help professional searchers efficiently find as many relevant documents as possible. High-recall retrieval can be useful in many cases, such as systematic literature review, patient cohort retrieval, patent search, e-discovery, and market research.

In Chapter 4, I adopt an instantiation of ReQ-ReC to solve medical word sense disambiguation (WSD) tasks. By inviting domain experts to search for typical examples of each word sense, the WSD model quickly gain performance at the beginning of the learning process, which reduces the overall demand of labeled examples to achieve a high WSD accuracy.

In Chapter 5, I designed a novel algorithm that directly learns from expert's prior knowledge (distinctive features) in medical WSD tasks. It gives the WSD model a strong performance at the very beginning of learning, effectively helping the model

reach high accuracy with significantly fewer labels than baseline interactive learning methods, including classical active learning methods and the ReQ-ReC instantiation in Chapter 4.

In Chapter 6, I propose a common principle underlying many interactive machine learning algorithms. It is a unified framework that depicts interactive machine learning as a two-player game, in which the data selection algorithm has a clear objective. The framework is general enough to explain many active learning algorithms as special cases. The chapter then discusses novel instantiations of the framework, including different choices of an objective term and a new query synthesis algorithm for text classification. Preliminary results show that the framework is effective in a high label noise case where uncertainty-based active learning underperforms random sampling.

I summarize the dissertation in Chapter 7, with discussions on its limitations and many research directions that naturally follow from it.

CHAPTER 2

A Review of Interactive Machine Learning

In this chapter, I conduct a review of interactive machine learning. It starts with a general discussion about possible interaction modalities between the human and the machine. To develop better interactive learning algorithms, it is important to first recognize the difference between "languages" spoken by the human and the machine, and then design interactions on top of their common language. Then it surveys a variety of algorithms for interactive machine learning, sorted by type of questions they can ask. As machine learning is increasingly used by real human users, it has attracted the attention from the human-computer interaction (HCI) community. The last part of the survey shows recent studies on machine learning from an HCI perspective.

This dissertation only considers training supervised learning models, not reinforcement learning agents. Note that reinforcement learning agents also learn by interacting with the environment, and sometimes human teachers. This is out of the scope of this dissertation.

2.1 Interactive Learning Algorithms

What types of interaction can happen between a machine learning algorithm and a human teacher? To answer this question, it is helpful to first consider what language is spoken by both the human and the machine. Their languages have overlap but are not completely the same. Figure 2.1 is a Venn diagram that visualizes this relationship.

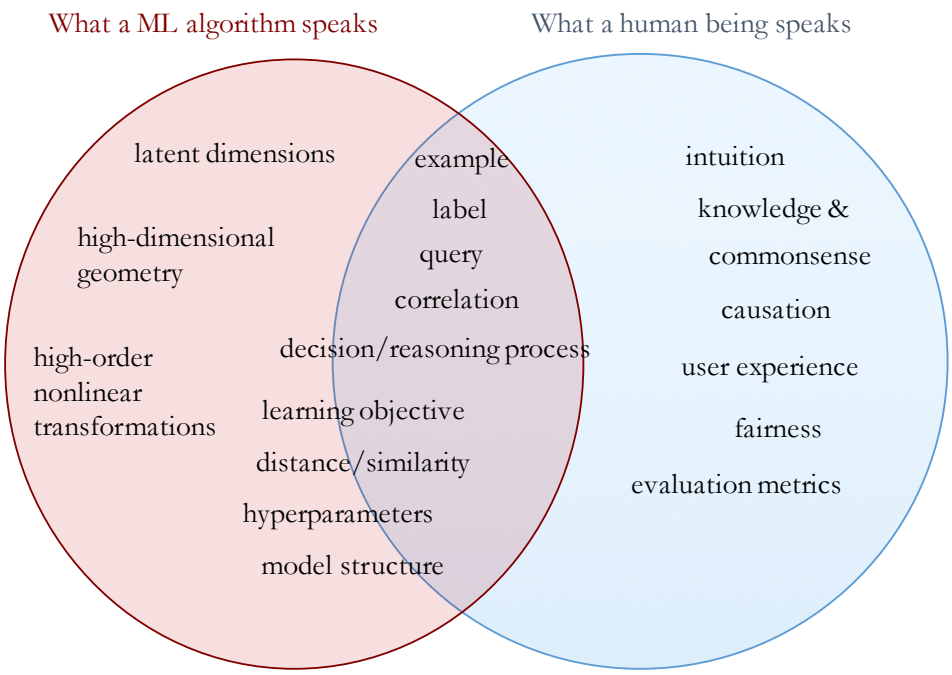


Figure 2.1: Humans and machine learning algorithms understand different languages.

Effective interactions between the algorithm and the human being can only happen in the shared space in Figure 2.1, as well as composition of semantics therein. A normal human user does not intuitively understand "margin" in a high-dimensional space. An algorithm does not understand "causal relation" if it is not programmed to do so. To expand the modes of interaction, we can (1) make the learning algorithm more powerful (e.g. to design more flexible ways of data representation, to target at more complex learning objectives, and search across wide range of hyperparameters) and (2) find more accessible ways to reveal the inner workings of the algorithm (e.g. through information visualization techniques).

Below we consider possible types of interaction that can happen between a human teacher and a machine student, and organize learning algorithms under different types of interaction. As a simple running example to facilitate presentation, we take a simple learning text mining task: to classify news text into "sports news" or "non-sports news". We assume that the human teacher is familiar with this task domain.

2.1.1 Labeling Instances

The most traditional and prominent type of interaction is to label existing instances, as used by a wide variety of active learning algorithms.

- Machine: What is the label of '`Michigan Football is ranked 3rd in Big Ten in 2015-16 season`'?
- Human: The label is 'Sports News'.

Extensive research efforts have been devoted in this line of research. Refer to [117, 3] for systematic reviews of the area. Recent years have seen increasing applications of active learning. It solves tasks beyond classification, such as recommendation [56, 63], ranking [16, 92], representation learning [162], etc. Active learning is applied to save labels for real-world, industry-scale problems, such as training search engines [157] and computational advertising [107].

2.1.2 Searching for Instances

When the training data is very sparse, there even lacks *un*labeled examples for active learning algorithms to query in the first place. This situation happens at the early stage of learning, or when the class distribution is highly imbalanced (few examples for the minority but interesting class) [10, 7]. In such cases, the algorithm can ask the human to retrieve or generate an example of a class.

- Machine: Can you give me an example of 'Sports News'?
- Human: '`Jim Harbaugh is the current coach of Michigan Football.`'

In text classification tasks, it is more convenient to allow the user to retrieve an example using keyword search than to write down an example from scratch. On the other hand, in image recognition tasks, it could be convenient to allow the user to sketch an example

(e.g. when the task is to classify handwritten digits 0-9). When the goal is to retrieve as much example in one class as possible (as in high-recall retrieval), the task is referred to as "active search" in literature [67, 142]. Theoretical analysis has shown the power of adding a search component into active learning [14]. This line of research is intimately related to Chapters 3 and 4.

2.1.3 Labeling Features

The algorithm can present a feature to the user, and ask the user to label which class is most strongly associated with that feature, if any.

- Machine: Which of the following features are indicative of the class 'Sports News': `football`, `water`, `tax`.
- Human: `football` is an indicative feature; `water` is neutral; `tax` is likely a negative feature.

This direction is called active feature labeling (AFL) [87, 105, 104, 36, 147, 102]. The features can be ranked by the machine-predicted correlation with each class. In this line of work, feature importance is used as either "soft data" or prior/regularization in training machine learning models [35, 58, 65, 101]. Usually, this type of feature labeling is coupled with instance labeling, and referred to as "active dual supervision" [106, 94, 9, 118, 71]. Labeling features out of context can be ambiguous, especially when the feature is not very indicative. In such scenarios, the human should be able to answer "neutral", "nonrelevant" or "I don't know".

2.1.4 Labeling Rationales

Since labeling features can be hard, it is sometimes more user-friendly to ask the human teacher to pinpoint a subpart of inside a labeled instance, showing why the instance is labeled as such.

- Machine: Please highlight the feature(s) indicating 'Sports News': '`Kobe reflects on his NBA career.`'.
- Human: `Kobe reflects on his NBA career.` ('NBA' is highlighted).
- Machine: In '`Loyal fans of Kobe beef crowded the restaurant`', the feature `Kobe` is a strong feature of 'Sports News'. Am I right?
- Human: No. In fact, the phrase `Kobe beef` is a negative feature of 'Sports News'.

In the first interaction above, the user highlighted the most informative word as the rationale. In the second interaction, when the machine has its own guess of rationale, the human can confirm or reject the guess. This direction is explored recently, referred to as "active learning with annotator rationales", or "transparent active learning" [159, 123, 15]. This line of work heralded the research area of interpretable/explainable machine learning [78, 46, 77, 17, 34].

2.1.5 Machine Teaching

As an inverse problem of machine learning, the subfield of *machine teaching* aims at constructing the smallest data set to train a desired machine learning model [168, 91]. Such a goal sounds very similar to that of active learning, except for an critical difference: machine teaching assumes the teacher knows the final model, including its structure and parameter values, while a teacher in an interactive machine learning setting does not know such a final model. For complex machine learning tasks in practice, there is no way even for human experts to know precise parameter values of a model.

The mixed-initiative classifier training proposal relaxes the above assumption: human experts can guide learning by providing good initial training examples, and later on active learning can query examples around the decision boundary to fine-tune the

model [133]. The ReQ-ReC framework in Chapters 3 and 4 can be seen as mixed-initiative learning procedures for real-world text mining tasks.

2.1.6 Other Modes of Interaction

To reduce labeling effort, enrich teaching channels, and account for real-world concerns, researchers have been proposing new modes of interaction for machine learning. This include active learning from pairwise comparisons [44, 66], rules of thumb [109, 24], noisy oracles [32, 127], and even manipulating confusion matrices [70]. Weak supervision signals are extracted from existing knowledge bases for information extraction tasks [59]. Different signals of supervision are then translated into loss terms or constraints in model training.

2.2 Human Factors

As machine learning and data analytics become increasingly popular in recent years, interactive machine learning is rising as a research topic and gaining increasing attention from the HCI community. While machine learning researchers focus more on the algorithm side, HCI researchers bring a holistic perspective from the human side.

HCI researchers emphasize that *humans are not oracles* [4]. This is in contrast with the standard assumption made by active learning: that the human annotator always provides accurate answers to each example. A real human user can have non-uniform labeling cost [121, 119, 12], fatigue, inaccuracy, sense of achievement and frustration when observing the progress of learning, desire to have more control on what and how the algorithm learns, curiosity of understanding why the algorithm make specific decisions, and subjective perception of performance (other than the reported accuracy or F_1-score). These human factors surface from recent work in human computation, where data annotation tasks are crowdsourced to non-expert users [82]. In such cases,

an interactive machine learning algorithm should account for quality of contributions from different annotators when querying for labels [22].

The HCI community places human users at the core of a machine learning application. Indeed, human interaction persists in the entire life cycle of machine learning: data acquisition, feature determination, class definition, objective design, annotation, hyperparameter tuning, model interpretation, and model revision as new data arrive. A human user's decision plays the central role in each step, most of which go through an interactive process. From an HCI perspective, the goal is to design systems that enable human users to have a smooth and intuitive experience in using and understanding machine learning algorithms. This perspective motivates us to see a bigger picture than devising better learning algorithms.

To support the full life cycle of interactive machine learning, the HCI community has been proposing general principles and advice on interface design [146, 62, 163, 77] and specific designs for visual data analytics, such as text mining [26, 155], time series analysis [100], mobile application clustering [21], semantic space exploration [38], and social networks [6]. In real-world annotation tasks, the human may revise her understanding of the task concept, which should be accommodated by the interface [76]. To facilitate the collection of labels, crowdsourcing techniques are proposed [31, 73]. To help the user better understand and manage learned machine learning models, researchers have been proposing interpretable and debuggable machine learning methods [77, 78, 46, 5, 17, 34]. Finally, even a well-trained machine learning model can still have blind spots if its training data were biased. The machine cannot be self-aware because it is too "confident" about its predictions. In such cases, we need to invite human users into the "machine debugging" loop and identify those blind spots [8, 79].

CHAPTER 3

Interactive High-Recall Retrieval

This chapter considers a scenario where a professional searcher requires both high precision and high recall from an interactive retrieval process. Such scenarios are very common in real life, exemplified by medical search, legal search, market research, and literature review. When access to the entire data set is available, an active learning loop could be used to ask for additional relevance feedback labels in order to refine a classifier. When data is accessed via search services, however, only limited subsets of the corpus can be considered — subsets defined by queries. In that setting, relevance feedback [114] has been used in a query enhancement loop that updates a query.

We describe and demonstrate the effectiveness of ReQ-ReC (ReQuery-ReClassify), a double-loop retrieval system that combines iterative expansion of a query set with iterative refinements of a classifier. This permits a separation of concerns: the query selector's job is to enhance recall, while the classifier's job is to maximize precision on the items that have been retrieved by any of the queries so far. The overall process alternates between the query enhancement loop (to increase recall) and the classifier refinement loop (to increase precision). The separation allows the query enhancement process to explore larger parts of the query space. Our experiments show that this distribution of work significantly outperforms previous relevance feedback methods that rely on a single ranking function to balance precision and recall.

Acknowledgment. The study in this chapter was conducted in close collaboration

with Dr. Cheng Li, who contributed equally to the design, implementation, evaluation, and presentation of this study. We published the results in SIGIR 2014 as co-first authors [85]. Dr. Li has generously granted me permission to present this work as a chapter in this dissertation.

3.1 Introduction

We are witnessing an explosive growth of text data in many fields, including millions of scientific papers, billions of electronic health records, hundreds of billions of microblog posts, and trillions of Web pages. Such a large scale has created an unprecedented challenge for practitioners to collect information relevant to their daily tasks. Instead of keeping local collections of data related to these tasks, many users rely on centralized search services to retrieve relevant information. These services, such as Web search engines (e.g., Google), literature retrieval systems (e.g., PubMed), or microblog search services (e.g., Twitter search API) typically return a limited number of documents that are the most relevant to a user-issued query. These existing retrieval systems are designed to maximize the precision of top-ranked documents; they are good at finding "something relevant," but not necessarily everything that is relevant.

We focus on scenarios where a user requires a high recall of relevant results in addition to high precision. Such scenarios are not uncommon in real life, exemplified by social search, medical search, legal search, market research, and literature review. For example: a social analyst needs to identify all the different posts in which a rumor spreads in order to reconstruct the diffusion process and measure the influence of the rumor; a physician needs to review all the patients that satisfy certain conditions to select cohorts for clinical trials; an attorney needs to find every piece of evidence related to her case from documents that are under legal hold; a scientist does not want to miss any piece of prior work that is related to his ongoing research. We denote all these

tasks generically as "high-recall" retrieval tasks.

Finding a needle in a haystack is hard; finding all the needles in a haystack is much harder. Existing retrieval systems do not naturally meet this type of information need. To conduct a comprehensive literature review using a search engine, we have to submit many alternative queries and examine all the results returned by each query. Such a process requires tremendous effort of the user to both construct variations of queries and examine the documents returned.

This high-precision and high-recall task becomes substantially harder as the collection grows large, making it impossible for the user to examine and label all the documents in the collection, and impractical even to label all the documents retrieved by many alternative queries. In some contexts such as e-discovery, a computer-assisted review process has been used that utilizes machine learning techniques to help the user examine the documents. Such a process typically casts high-recall retrieval as a binary classification task. At the beginning, the user is required to label a small sample of documents. A classifier trained using these labeled documents then takes over and predicts labels for other documents in the collection. An active learning loop can be used to ask for additional relevance labels in order to refine the classifier. These methods, however, require that the user has access to the full collection of documents and that it is feasible to execute her classifier on all the documents.

In other scenarios, the users either do not own the collection or it is too large, so they can only access documents in the collection through an external search service. This makes it unrealistic to either examine or classify the entire collection of documents. Instead, only limited subsets of the document corpus can be considered, subsets defined by queries.

Existing retrieval systems are not tuned for high-recall retrieval on the condition of limited access to the data via search services. In most cases, a system only aims to maximize the precision in the documents that are retrieved by the current query.

Relevance feedback has been used in a query enhancement loop that updates a query. Many search engines provide services to collect explicit and/or implicit feedback from the users or to suggest alternative queries to the users. These practices typically generate a new query that replaces the old one, which is expected to improve both precision and recall. Once a new query is issued, the results retrieved by the old queries are forgotten, unless they are manually harvested by the user.

We study a novel framework of retrieval techniques that is particularly useful for high-recall retrieval. The new framework features a ReQ-ReC (ReQuery-ReClassify) process, a double-loop retrieval system that combines iterative expansion of a query set with iterative refinements of a classifier. This permits a separation of concerns, where the query generator's job is to enhance recall while the classifier's job is to maximize precision on the items that have been retrieved by *any* of the queries so far. The overall process alternates between the query expansion loop (to increase recall) and the classifier refinement loop (to increase precision). The separation of the two roles allows the query enhancement process to be more aggressive in exploring new parts of the document space: it can explore a non-overlapping portion of the corpus without worrying about losing the veins of good documents it had found with previous queries; it can also use queries that have lower precision because the classifier will weed out the misses in a later stage. Our experiments show that this distribution of work significantly outperforms previous relevance feedback methods that rely on a single ranking function to balance precision and recall. The new framework also introduces many opportunties to investigate more effective classifiers, query generators, and human-computer interactive algorithms for labeling subsets, and especially to investigate what combinations work best together.

Unlike Web search engines that target users who have real-time, ad hoc information needs, the ReQ-ReC process targets users who care about the completeness of results and who are willing to spend effort to interact with the system iteratively and

judge many (but not all) retrieved documents. The process has considerable potential in applications like social media analysis, scientific literature review, e-discovery, patent search, medical record search, and market investigation, where such users can be commonly found.

The rest of this chapter is organized as follows. We discuss related work in Section 3.2. Section 3.3 gives an overview of the ReQ-ReC double-loop framework and its key components. Section 3.4 describes several instantiations of the framework. Section 3.5 provides a systematic evaluation of the proposed methods. Finally, we conclude in Section 3.6.

3.2 Related Work

The ReQuery-ReClassify framework integrates and extends two well-established "human-in-the-loop" mechanisms: relevance feedback in information retrieval, and active learning in text classification.

Relevance feedback was shown long ago to be effective for improving retrieval performance [114]. In a feedback procedure, the retrieval system presents the top-ranked documents to the user and collects back either explicit judgments of these documents or implicit feedback implied by certain actions of the user [69, 125]. The system then learns from the collected feedback and updates the query. The new query reflects a refined understanding of the user's information need [110, 160], which improves both precision and recall in the next round of retrieval. Even without real user judgments, retrieval performance may still benefit from simply treating the top-ranked documents as relevant, which is known as a process of pseudo relevance-feedback [18].

In a search session, relevance feedback can be executed for multiple rounds. Harman [55] studied multiple iterations of relevance feedback, and found that retrieval performance is greatly improved by the first two to three iterations, after which the

improvements became marginal. Multiple iterations of relevance feedback have received more attention in content-based image retrieval [29, 112, 165].

In complicated search tasks, the user is often involved in a search session consisting of a series of queries, clickthroughs, and navigation actions. *Session*-based retrieval aims at learning from these signals in order to better understand the user's information need, thus improving the relevance of results when the user issues the next query [125, 108]. Instead of improving the performance of the next query, ReQ-ReC aims to maximize the recall of the results collectively retrieved by all the queries in the search session.

Like traditional iterative relevance feedback, the ReQ-ReC process also adopts multiple iterations of user interaction. Indeed, as shown in Section 3.3, iterative relevance feedback is a special case instantiation of the ReQ-ReC framework. Instead of replacing the old query with a new query, however, ReQ-ReC can accumulate documents retrieved by any of the queries issued so far. By doing this, rather than optimizing both precision and recall through the choice of a single query, we place the burden of maximizing precision on a classifier, and new queries can be dedicated to improving only recall.

When it is feasible to process the entire collection of documents, the problem of high-recall retrieval can be cast as a binary classification problem where the positive class captures documents that are relevant to the information need and the negative class captures the rest. The practice of relevance feedback essentially becomes an active learning process, in which the system iteratively accumulates training examples by selecting documents and asking the user for labels [117]. This strategy is commonly used in computer-assisted reviews for e-discovery, often referred to as the process of 'predictive coding' [97]. Different active learning algorithms use specific strategies for selecting the documents to label, many of which attempt to maximize the learning rate of a 'base' classifier with limited supervision [117]. For text classification, a popular

choice of such a 'base' classifier is the support vector machine (SVM) [28]. Using SVM, a variety of document selection strategies have been explored. Tong and Koller [136] proposed to select documents closest to the decision hyperplane in order to rapidly shrink the version space and reduce model uncertainty. In contrast, Drucker et al. [37] selected documents with highest decision function values to avoid probing the user with too many non-relevant documents. Xu et al. [153] mixed these two strategies and achieved better retrieval performance.

Like active learning, the ReQ-ReC process also trains a binary classifier. The major difference is that ReQ-ReC does not require knowledge about the entire document collection and thus does not classify *all* documents. Instead, it starts from a limited subset defined by the original query and actively expands the space. This is a huge gain, as text classification and active learning are usually computationally prohibitive for modern IR collections containing a large number of documents [27]. Indeed, previous studies that apply active learning to retrieval can only evaluate their approaches using moderate-scale collections (such as the 11,000-documents Reuters collections used in [37] and [153]), or only focus on the documents retrieved by one query (top 100 documents in [154] and top 200 in [135]). Given its big advantage in efficiency, the ReQ-ReC process could potentially provide a new treatment for active learning, especially when the data collection is large and the positive class is very rare.

The idea of active learning has also been applied to relevance feedback for retrieval. Shen and Zhai [126] studied active feedback, where the system *actively* selects documents and probes the user for feedback instead of passively presenting the top ranked documents. It is shown that selecting diverse top-ranked documents for labeling is desirable, since it avoids asking for labels on similar documents and thus accelerates learning. Xu et al. [154] improved this heuristic by jointly considering relevance, diversity, and density in selected documents. Both techniques exploit density information among top-ranked documents, and select representative ones for feedback. Recently,

Tian and Lease [135] combined uncertainty sampling (*Simple Margin*) and density-based sampling (*Local Structure*) in iterative relevance feedback to minimize user effort in seeking several to many relevant documents. The difference between our work and theirs is articulated by the difference between the ReQ-ReC process and relevance feedback described above: the addition of a classifier and use of results from all queries allows more aggressive exploration of alternative queries.

3.3 The ReQ-ReC Framework

In this section, we introduce the general ReQuery-ReClassify (ReQ-ReC) framework, including its key components. Specific instantiations of the framework will be discussed in the next section. The basic idea of the framework is to distribute the burden of maximizing both the precision and recall to a *set* of queries and a classifier, where the queries are responsible for increasing the recall of relevant documents retrieved and the classifier is responsible for maximizing the precision of documents retrieved collectively by all of the queries in the set. The framework features a double-loop mechanism: the inner-loop classifies the retrieved documents, actively collects user feedback, and improves the classifier (ReClassify); the outer-loop generates new queries (ReQuery), issues API calls, and iteratively adds newly retrieved documents into the workset. In the rest of the chapter, we refer to the framework as "ReQ-ReC" or "double-loop" interchangeably.

3.3.1 The Double Loop Process

The ReQ-ReC framework can be viewed as a double-loop review process, as illustrated in Figure 3.1. The process maintains a set of queries, a pool of retrieved documents, and a binary classifier. With an initial query composed by the user, the system retrieves an initial set of documents using a search service. An inner-loop starts from there, in

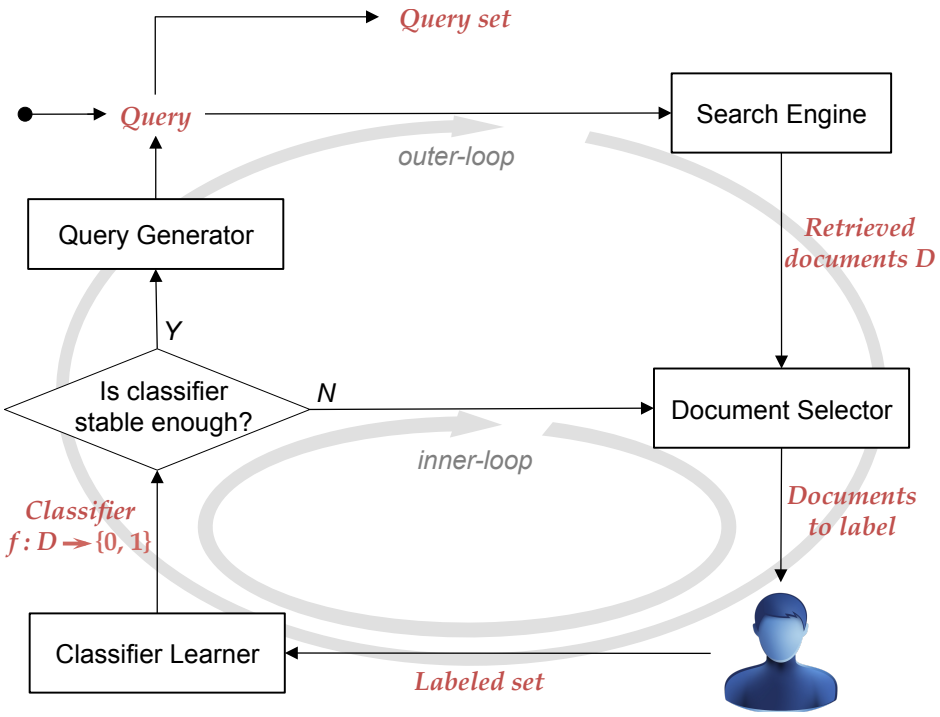
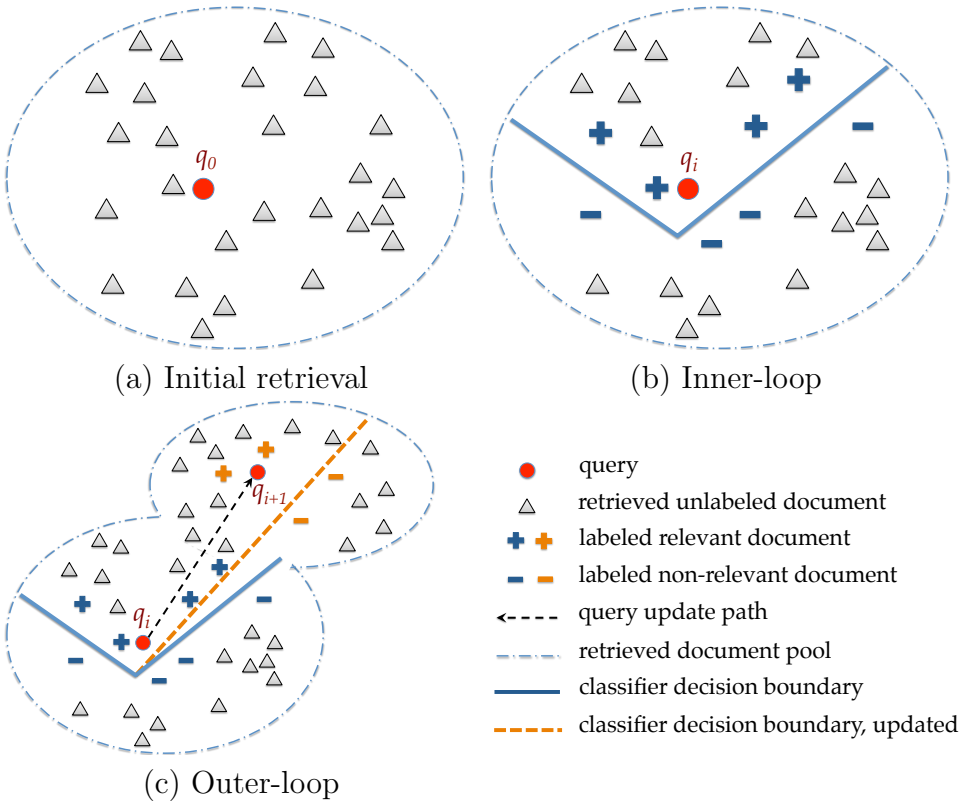


Figure 3.1: ReQ-ReC framework

which the system iteratively presents a small number of documents (e.g., 10) selected from the current pool of retrieved documents to the user and asks her to label them as either relevant or not. The classifier is consequently updated based on the accumulated judgments of the user, which is then used to *reclassify* the pool of documents. After a few iterations of the inner-loop, the the classifier's predictions stabilize. At this point, the inner-loop will suspend. The system then proposes to add a new query to the query set, aiming to retrieve more relevant documents from the collection. Upon the approval—and possible edits—of the user, the system will retrieve a new set of documents using the new query, and *merge* them into the pool of retrieved documents. The *requery* process makes up one iteration of the outer-loop of the framework. After new documents are retrieved and added into the pool, the system starts a new inner-loop and continues to update the classifier left from the last iteration. The whole review process will end when no more relevant documents can be retrieved by a new query or when the user is satisfied.



(a) Initial retrieval (b) Inner-loop

(c) Outer-loop

Figure 3.2: A double-loop process of search in the information space.

(a) Each query only retrieves its surrounding region under inspection. (b) The inner-loop updates a classifier that refines the boundary between relevant and non-relevant documents. (c) The outer-loop expands the subspace which includes more relevant documents.

Another way to look at the framework is to imagine a search process in the information space (e.g. a vector space of documents and queries), as illustrated in Figure 3.2. The system interacts with the user as it navigates through the information space, aiming to delineate a manifold that contains as many relevant documents and as few non-relevant documents as possible. Each query can only reveal a small region of the information space that surrounds it. The "first guess" on such a manifold is, of course, the region surrounding the initial query. A classifier clarifies the boundary of the manifold (to maximize precision), which is iteratively refined with newly labeled data points selected from the revealed regions. To explore other regions in the space so as to expand the relevant manifold (to maximize recall), the system will estimate a promising

direction and will make a new query to move in that direction into the uncharted space. This new region and all previously unveiled regions are combined as the current search space, in which the system continues to refine the boundary of the relevant manifold. The search process will end if the relevant manifold stops expanding, or if the user decides to terminate early.

From this perspective, each query contributes a new region to the search space without giving up any already discovered regions. Such a pure "expansion" of the search space will include many non-relevant documents, but the classifier is able to filter the non-relevant documents at the end and recover the true boundary of the relevant manifold. By contrast, in a relevance feedback procedure, every new query will "redefine" the search space as the region surrounding the new query. Given a good query, this region indeed contains fewer non-relevant documents than our "expanded" search space (i.e., achieves a higher precision), but it is also likely to contain fewer new relevant documents. In relevance feedback, the challenge is to find a new query that both retrieves the relevant documents from the old query and also retrieves new ones. In ReQ-ReC, the challenge is simply to find a query that retrieves new relevant documents.

3.3.2 Anatomy of the ReQ-ReC Framework

Given the high-level intuitions of the ReQ-ReC framework, we now discuss the key components in the double-loop. To facilitate the discussion, we introduce the notations in Table 3.1 and summarize the framework in Algorithm 1.

3.3.2.1 Search

The ReQ-ReC framework assumes neither ownership nor full access to the document collection, but instead relies on a standard search service to retrieve documents from the index. The retrieval service's ranking function can use any reasonable retrieval model

Table 3.1: Notations of the double-loop process

$\mathcal{D}$	index of the document collection
q_i	the i-th query submitted
$\mathcal{D}_q$	the union of all unjudged documents retrieved by the set of queries $\{q_i\}$
$\mathcal{D}_s$	documents selected for user judgments
$\mathcal{D}_l$	set of documents labeled already
$retrieve(\mathcal{D}, q_i)$	a retrieval function that returns a subset of documents from index $\mathcal{D}$ by query q_i
Θ_A	model for document selection
Θ_R	model for relevant/non-rel classification
$train_A(\mathcal{D}_q, \mathcal{D}_l)$	function to train/update Θ_A using labeled and unlabeled documents
$train_R(\mathcal{D}_q, \mathcal{D}_l)$	function to train/update Θ_R using labeled and unlabeled documents
$selectK(\mathcal{D}_q, \Theta_A)$	function to select K documents using the document selection model
$label(\mathcal{D}_s)$	function to obtain relevance labels of $\mathcal{D}_s$
$predict(\mathcal{D}_q, \Theta_R)$	function to predict the relevance labels and rank unlabeled documents
$query(\{q_i\}, \cdot)$	function to generate a new query

Algorithm 1 The double-loop process

Input: Initial query q_0, index of document collection $\mathcal{D}$

Output: A set of labeled documents $\mathcal{D}_l$ and a set of unjudged documents in $\mathcal{D}_q$ with system predicted labels.

$\mathcal{D}_q \leftarrow \emptyset$
$\mathcal{D}_l \leftarrow \emptyset$
repeat // outer loop
  $\mathcal{D}_q \leftarrow retrieve(\mathcal{D}, q_i) \cup \mathcal{D}_q$
  repeat // inner loop
    if $\mathcal{D}_l == \emptyset$ **then**
      $\mathcal{D}_s \leftarrow selectK(\mathcal{D}_q)$
    else
      $\Theta_A \leftarrow train_A(\mathcal{D}_q, \mathcal{D}_l)$
      $\mathcal{D}_s \leftarrow selectK(\Theta_A, \mathcal{D}_q)$
    end if
    $\mathcal{D}_l \leftarrow \mathcal{D}_l \cup label(\mathcal{D}_s)$
    $\mathcal{D}_q \leftarrow \mathcal{D}_q - \mathcal{D}_s$
    $\Theta_R \leftarrow train_R(\mathcal{D}_q, \mathcal{D}_l)$
    $predict(\Theta_R, \mathcal{D}_q)$
  until meet stopping criteria for inner loop
  $q_{i+1} \leftarrow query(\{q_i\}, \mathcal{D}_q, \mathcal{D}_l, \Theta_A, \Theta_R)$
until meet stop criteria for outer loop

that takes the input of a query q_i and outputs a certain number of ranked documents from the index (e.g., using a vector space model, a language modeling approach, or a boolean retrieval model). In most cases, the user has no knowledge about the algorithm that is employed by the external search service. In that case, the retrieval function is treated as a black box in the framework.

After each search process the retrieved documents will be merged into the pool of unlabeled documents $\mathcal{D}_q$, which expands the workset for document selection and classification.

3.3.2.2 Document Selection

In every iteration of the inner-loop, during steps 6-10 of the algorithm the system selects K (e.g., 10) documents $\mathcal{D}_s$ from the pool of retrieved documents that are yet unlabeled, $\mathcal{D}_q$, and asks the user for judgments. At the beginning of the double-loop process, where there are no judged documents, this process can simply return the top documents ranked by the retrieval function, select a more diverse set of documents through an unsupervised approach, or even randomly sample from $\mathcal{D}_q$. Once labeled documents have been accumulated, the process is able to select documents based on an active learning strategy. Such a process aims to maximize the learning rate of the classifier and thus reduce the user's effort on labeling documents.

3.3.2.3 Classification

Given an accumulated set of labeled documents, the classification component learns or updates a binary classifier (i.e., Θ_R) at step 14 and reclassifies documents from $\mathcal{D}_q$ at step 15. Any reasonable classifier can be applied here.

In many high-recall retrieval tasks such as medical record search, it is important to find all patients that "match" certain conditions, but it is not necessary to rank the records identified as relevant [52]. In those cases, the labels of documents in $\mathcal{D}_q$ can

be directly predicted by the classifier. In cases where ranking is desired, documents in $\mathcal{D}_q$ and $\mathcal{D}_l$ can be ranked/reranked using either the confidence values or the posterior probabilities output by the classifier, or by using an alternative machine learning method such as a regression or learning-to-rank model.

3.3.2.4 Query Expansion

When the classifier appears to be achieving a stable precision on the current workset of documents $\mathcal{D}_q$, the system proceeds to expand $\mathcal{D}_q$ in order to increase the recall. This is done through constructing a new query (step 17) and retrieving another set of documents through the search service. Any reasonable query expansion method can be applied here, including the classical relevance feedback methods such as Rocchio's [110] or model-based feedback [160]. Other query reformulation methods can also be applied, such as synonym expansion [139] and semantic term matching [40].

3.3.2.5 Stop Criteria

Stop criteria of the inner-loop: new labels stop being requested when either of the following conditions is met:

- The performance of the classifier converges. The system correctly predicts the user's labels of a new batch of documents $\mathcal{D}_s$ and, after adding those labels, there is no evident change in the classifier's predictions.
- The user runs out of energy or patience.

Stop criteria of the outer-loop: new queries stop being submitted when either of the following conditions is met:

- New queries no longer pick up new relevant documents. This can be assessed heuristically by running the existing classifier on a new result set, or can be veri-

fied by running the inner loop again to check whether any new positive documents are identified.

- The user runs out of energy or patience.

3.4 Instantiations of ReC-ReQ

The key components of the general ReQ-ReC framework, document selection, classification, and query expansion can be instantiated in many ways. To illustrate the power of the framework, we describe five instantiations, beginning with iterative relevance feedback as a degenerate form and progressively substituting elements that take greater advantage of the broader framework. Section 3.5 will provide performance comparisons of these instantiations.

3.4.1 Iterative Relevance Feedback

Interestingly, an iterative relevance feedback process can be interpreted as a special case of the ReQ-ReC framework, if both the classification component and the document selection component simply adopt a ranking function that is based on the current query, q_i. More specifically, define Θ_R to classify a document as relevant if it is in $retrieve(\mathcal{D}, q_i)$, and define Θ_A to always select the next highest ranked unlabeled item from $retrieve(\mathcal{D}, q_i)$. There is no difference in whether the results retrieved by the previous queries are kept in the document pool $\mathcal{D}_q$ or not, if the results are eventually ranked based on the last query, q_i.

Note that many query updating methods (in the context of relevance feedback) can be applied to generate the new query at each iteration. To establish a baseline for performance comparison, we choose Rocchio's method [110], by which the next query

is selected according to Equation 3.1:

$$\vec{q_i} = \alpha\vec{q_0} + \beta\frac{1}{|D_r|}\sum_{\vec{d_j}\in D_r}\vec{d_j} - \gamma\frac{1}{|D_{nr}|}\sum_{\vec{d_k}\in D_{nr}}\vec{d_k}, \tag{3.1}$$

where $\vec{q_0}$ is the original query vector, D_r and D_{nr} are the set of known relevant and nonrelevant documents, and α, β, and γ are parameters. The basic idea of Rocchio's method is to learn a new query vector from documents labeled as positive or negative, and then interpolate it with the original query vector. When the parameters are well tuned, this achieves performance comparable to alternatives such as model-based feedback [160] and negative feedback [141].

3.4.2 Passive

The next two instantiations modify the relevance feedback process by introducing a separate classifier, Θ_R, rather than using the retrieval function as a degenerate classifier. This classifier is involved to maximize the precision of labels for $\mathcal{D}_q$. Here, keeping the documents retrieved by previous queries does make a difference, because Θ_R will operate at the end to rank all of the results from all of the queries.

Any machine learning-based classifier, as well as any reasonable selection of features, can be used to identify relevant documents in $\mathcal{D}_q$. We adopt the support vector machine (SVM) [28] with unigram features and linear kernel. In cases where a ranked list of documents is desired, documents in $\mathcal{D}_q$ are ranked by the score of the decision function $w^T x + b$ output by linear SVM.

We call this second instantiation of ReQ-ReC *Passive*. It is passive in the sense that the classifier is not used to control the interactive process with the user; we still choose the top-ranked documents for labeling and use Rocchio's method of query expansion, as in our iterative RF instantiation. By comparing the performance of *passive* and the Iterative RF baseline, we can determine the effect of the classifier acting solely as a

post-hoc reranking function.

3.4.3 Unanchored Passive

Note that in Rocchio updating, the parameter that interpolates the new query vector with the original query is quite sensitive. This is because when one relies on the query to maximize both precision and recall, the expansion has to be conservative so that the new query does not drift too far from the original query. When the burden of maximizing precision is transferred from the query to the classifier, we anticipate that this interpolation should become less critical. To test this, we introduce another simple instantiation by removing the original query vector (i.e., the $\vec{q}_0$ component in Equation 3.1) from Rocchio, by setting $\alpha = 0$. Note that this is a rather extreme case for test purposes. In reality, keeping closer to the original query may still be important even for the purpose of increasing recall. We call this instantiation *Unanchored Passive*, because the updated queries are no longer anchored to the initial query.

3.4.4 Active

Next, we consider an instantiation of RecQ-ReC that makes use of the classifier to select documents for labeling in the inner loop. As before, we train the classifier using SVM. We select documents for labeling using uncertainty sampling [136], a simple active learning algorithm that selects examples closest to the decision hyperplane learned by the classifier. In each inner-loop iteration, we present to the user ten documents that are the most uncertain by the current classifier. Specifically, five are chosen from each side of the hyperplane. We call this instantiation *Active* because the classifier is active in choosing which documents to label.

Note that after the very first search process, the system has *no* labeled documents in the pool. A classifier cannot be trained and thus the uncertainty sampling cannot be applied. At this *cold start*, we simply select the top 10 documents returned by the

search service as the first batch of documents to request user judgments.

As uncertainty-based active learning gradually refines the decision boundary of the classifier, every new query to the search service may affect its performance. This is because a new query expands the pool of documents $\mathcal{D}_q$ with newly retrieved documents, which might dramatically change the distribution and the manifold of data in the search space. At this point, instead of gradually refining the old decision boundary, the classifier may need a bigger push to quickly adapt to the new distribution of data and approach the new decision boundary. In other words, it is important for the classifier to quickly explore the newly retrieved documents. Therefore, in the first inner-loop iteration after each new query brings back new documents, we select top ranked documents for labeling instead of the most uncertain ones. Uncertain ones are picked in the following inner-loop iterations.

3.4.5 Diverse Active

The final instantiation we consider modifies the query expansion algorithm used in the Active instantiation. Previously, we considered an unanchored version of Rocchio's method of selecting the next query. Here, we consider a different modification of Rocchio's method.

To maximize recall, we naturally want a new query to retrieve as many relevant documents as possible. Even more importantly, these relevant documents should overlap as little as possible with the documents retrieved by previous queries. In other words, a new query should retrieve as many *new relevant* documents as possible.

Our idea is inspired by the theory of "weak ties" in sociology [50]. While strong ties trigger social communication, weak ties can bring in novel information. If we think of the top-ranked documents in a retrieved list as "strong ties" to the query, we can think of the lower-ranked documents as "weak ties." We thus exploit documents that are judged as relevant, but ranked lower in the list returned by the search service. These

documents are likely to act as bridges to expand the search space into other clusters of relevant documents.

Are there many such documents? In a relevance feedback process, there might be few, as the user always labels the top-ranked documents. In a ReQ-ReC process that actively selects documents, however, documents ranked lower by the retrieval function are more likely to be viewed and judged by the user.

In Equation 3.1, instead of using all relevant documents D_r, we use its subset D_{rl}, which includes the documents that are judged as relevant but ranked low by the original retrieval function. We employ a simple criterion to determine which documents should be included in D_{rl}. For each document d, we maintain its rank returned by the retrieval function, denoted as r_d. If the document has been retrieved by multiple queries in the past, its highest rank in those retrieved lists is kept. Let r_l be the lowest rank r_d of all the documents in D_r. We include documents that are ranked lower than $r_l/2$ in D_{rl}. This leads to inclusion in the next query of terms from relevant documents that were not highly weighted in previous queries. Since this method aims to diversify new queries, while still using the classifier to actively choose documents for labeling, we refer to this method as *Diverse Active*.

3.5 Experiments

In this section, we present empirical experiments to evaluate the effectiveness of the ReQ-ReC framework and its instantiations. We start with a description of the data sets, metrics, and methods included in the comparisons.

3.5.1 Data Sets

There are several criteria for selecting the right data sets for evaluating ReQ-ReC. Ideally, the data sets should be large enough and standard search APIs should exist. A

representative set of queries should also exist, and each query should have a reasonable number of relevant documents in the data set. To avoid the high variance of real-time user judgments and to facilitate comprehensive and fair comparisons, we use existing judgments for each query to 'automate' the actual user feedback in the process. The same approach is used in most existing work on relevance feedback (e.g., [55, 126, 141]). We therefore require that many relevant judgments exist for each query.

We first select four large scale TREC data sets, the data sets used in TREC-2012 Microblog Track (MB12) [130], TREC-2013 Microblog Track (MB13)[1], the TREC-2005 HARD Track (HARD), and the TREC-2009 Web Track (ClueWeb09[2], category A)[3]. These data sets normally provide 50–60 queries and 500–1,000 relevant judgments for a query.

Note that there is a natural deficiency of using TREC judgments for the evaluation of a high-recall task, simply because not all documents in a TREC data set have been judged. Instead, judgments are provided for only a pool of documents that consist of the top-ranked documents submitted by each participating team. In many cases, only a sample of the pool is judged. Therefore, it is likely that many relevant documents for a query are actually not labeled in the TREC provided judgments. This creates a problem for a 'simulated' feedback process—when the system requests the label of a document, the label may not exist in the TREC judgments. It is risky to label that document either as relevant or as irrelevant, especially because mislabeling a relevant documents as irrelevant may seriously confuse a classifier. In such situations, we ignore that document and fetch the next document available. The same treatment has been used in the literature [126]. When measuring the performance of a retrieved list, however, we follow the norm in the literature and treat a document not judged by TREC as negative.

To better understand the behavior of ReQ-ReC, it is desirable to include a data

[1] `https://github.com/lintool/twitter-tools/wiki/`
[2] `http://lemurproject.org/clueweb09/`
[3] `http://trec.nist.gov/data/web09.html`

Table 3.2: Basic information of data sets

	#docs	avg dl	#topics(IDs)	#qrels
20NG	18,828	225	20 categories	18,828
HARD	1,033,461	353	50 (303-689)	37,798
MB12	15,012,766	19	59 (51-110)	69,045
MB13	≈243,000,000	14	60 (111-170)	71,279
ClueWeb09	503,903,810	1570	50 (1-50)	23,601

* HARD has non-consecutive topic IDs. Topic 76 of MB12 has no judgment hence is removed.

set that is fully judged, even though a large data set like that is rare. Therefore, we include the 20-newsgroup data set (20NG) [80] for this purpose. As every document belongs to one of the 20 topics, we use the titles of 20 topics as the queries, following the practice in [37]. For words that are abbreviated in the topic titles, we manually expand them into the normal words. For example, "rec" is converted to "recreation," and "autos" to "automobiles." Although it is feasible to apply a classifier to the entire 20NG data set, we only access the data using rate-limited retrieval functions. The statistics of all five data sets in our experiments are presented in Table 3.2.

Both the 2013 Microblog Track[4] and the ClueWeb09[5] provide official search APIs, which are implemented using the Dirichlet prior retrieval function (Dirichlet) [161]. For other data sets, we maintain a similar search service using Lucene [1], which also implements the Dirichlet prior function. Documents are tokenized with Lucene's StandardAnalyzer and stemmed by the Krovetz stemmer [74]. *No* stopwords are removed.

3.5.2 Metrics

Many popular metrics for retrieval performance, such as *precision*@K and NDCG, are not suitable for high-recall tasks. We use two standard retrieval metrics that depend more on recall, namely the mean average precision (MAP) [93] and the R-precision (R-Prec) [93]. R-precision measures the precision at the R-th position for a query

[4] `https://github.com/lintool/twitter-tools/wiki/TREC-2013-API-Specifications`
[5] `http://boston.lti.cs.cmu.edu/Services`

Table 3.3: Baselines and methods included in comparison.

Method	**Document Selection**	**Classification**	**Query Expansion**	**# outer loops**	**# inner loops**
Relevance Feedback (RF)	top	-	Rocchio	1	1
Iterative RF	top	-	Rocchio	M	1
Passive	top	SVM at end	Rocchio	M	1
Unanchored Passive (Unanchored)	top	SVM at end	Rocchio - $\vec{q}_0$	M	1
Active	uncertainty	SVM	Rocchio	M	M
Diverse Active (Diverse)	uncertainty	SVM	divRoc	M	M

* **M**: multiple iterations; **top**: select 10 top-ranked documents; **uncertainty**: uncertainty-based active document selection; **divRoc**: diverse Rocchio; **Rocchio -** $\vec{q}_0$: Rocchio without interpolation of the original query.

with R relevant judgments. The R-th position is where precision equals recall. To increase R-precision, a system has to simultaneously increase precision and recall. For each query, we use the top 1,000 relevant documents (either labeled or predicted) to compute the measures.

When measuring performance, we include documents that the user labeled during the process. This is because a high-recall retrieval task is successful when more relevant documents can be found, whether they are actually judged by the user or predicted by the system. If an interactive process does a good job of presenting more relevant documents to the user, it should not be punished by having those documents excluded from the evaluation. In all methods included in comparative evaluation, we put the documents judged as relevant at the top of the ranked list, followed by those predicted to be relevant using Θ_R.

3.5.3 Methods

We summarize all baseline methods and ReQ-ReC instantiations included in our evaluation in Table 3.3. The most important baseline we are comparing with is the iterative

relevance feedback as described in Section 3.4.1, in which a new query is expected to maximize both precision and recall. We then include four instantiations of the ReQ-ReC framework, as described in Section 3.4.

In *Passive* and *Unanchored Passive*, we employed a negative form of pseudo-relevance feedback: the lowest ranked 1,000 documents retrieved by the final query are treated as negative examples to train the classifier. The positive examples for training came from the actual judgments.

3.5.4 Parameters

For the MB13 and ClueWeb09 datasets, we used the official search APIs, which returned, respectively, 10,000 and 1,000 documents per query. For the three data sets without official search APIs, the parameter of the Dirichlet prior μ for the base retrieval function was tuned to maximize the mean average precision and each query returned the top 2,000 matching documents.

To obtain the strongest baseline, we set the parameters of Rocchio to those that maximize the mean average precision of a relevance feedback process using 10 judgments. We fix α to be 1 and conduct a grid search on the other two. For ClueWeb09, we set the parameters according to the recommendation in [93] as the rate limits of the API prevent us from tuning the parameters. We do not further tune the parameters in the ReQ-ReC methods even though the optimal parameters for the baseline may be suboptimal for ReQ-ReC. The values of all the parameters used are shown in Table 3.4. In all our experiments, we also use the default parameter of SVM ($c = 1$). We stop the inner-loops when SVM confidence value produces stable ranking of $\mathcal{D}_q$, i.e., Spearman's rank correlation coefficient of previous and current rankings of $\mathcal{D}_q$ is above 0.8 for two consecutive inner-loops.

Table 3.4: Parameter settings: μ in Dirichlet prior; β and γ in Rocchio (α fixed as 1); Results per query: number of documents returned by a search API call.

	MB12	MB13	ClueWeb09	HARD	20NG
μ	2100	-	-	1100	3200
β	0.95	0.85	0.75	0.6	0.5
γ	0.4	0.15	0.15	0.05	0.4
Results/query	2,000	10,000	1,000	2,000	2,000

3.5.5 Overall Performance

Table 3.5 summarizes the performance of all included methods, with one additional criterion to stop the process when the "user" has judged 300 documents for a topic. Statistical significance of the results are provided by comparing to the baseline, iterative relevance feedback, and by comparing to another ReQ-ReC method. In general, methods developed under the ReQ-ReC framework significantly outperform iterative relevance feedback. *Diverse Active*, which uses an active document selection strategy and a diverse query expansion, achieves the best performance. For most data sets, the improvement over iterative relevance feedback is as large as 20% – 30% of MAP and R-Precision. This is promising given the difficulty of improvements based on those two metrics. On the largest data set, ClueWeb09, the best ReQ-ReC algorithm achieves more than 120% improvement over iterative relevance feedback.

We make the following remarks:

- (Compare *Relevance Feedback* with *Iterative RF*) Multiple iterations of relevance feedback indeed outperforms a single iteration of feedback, even if the same number of judgments (i.e., 300) are used in this single iteration. The only exception is the ClueWeb09 data, for which the collection is too large and the relevance judgments are very sparse. In this case, an iterative relevance feedback method may stop earlier if none of the top 10 results brought back by a new query are relevant. In that situation, presenting more documents to the user at once may

Table 3.5: Retrieval performance of competing methods. At most 300 judgments per topic. ReQ-ReC methods significantly outperform iterative relevance feedback.

	MB13		MB12		HARD		20NG		ClueWeb09	
	R-prec	MAP	R-prec	MAP	R-prec	MAP	R-prec	MAP	R-prec	MAP
Dirichlet	0.268	0.203	0.233	0.183	0.247	0.174	0.327	0.107	0.101	0.058
RF	0.417	0.415	0.466	0.479	0.440	0.447	0.451	0.356	0.256	0.229
Iterative RF	0.532	0.552	0.633	0.649	0.592	0.597	0.474	0.421	0.237	0.216
Passive	0.568**	0.585**	0.646**	0.661**	0.615**	0.637**	0.548**	0.490**	0.275**	0.247**
Unanchored	0.603$^{**\triangledown}$	0.618$^{**\triangledown}$	0.667$^{**\triangledown}$	0.673$^{**\triangledown}$	0.609	0.624**	0.527*	0.464*	0.268**	0.236**
Active	0.653$^{**\triangledown}$	0.661$^{**\triangledown}$	0.727$^{**\triangledown}$	0.740$^{**\triangledown}$	0.729$^{**\triangledown}$	0.737$^{**\triangledown}$	0.595$^{**\triangledown}$	0.562$^{**\triangledown}$	0.493$^{**\triangledown}$	0.493$^{**\triangledown}$
Diverse	**0.675**$^{**\triangle}$ (+27%)	**0.692**$^{**\triangle}$ (+25%)	**0.760**$^{**\triangle}$ (+20%)	**0.771**$^{**\triangle}$ (+19%)	**0.789**$^{**\triangle}$ (+33%)	**0.799**$^{**\triangle}$ (+34%)	**0.620**$^{**\triangledown}$ (+31%)	**0.580**$^{**\triangledown}$ (+38%)	**0.533**$^{**\triangle}$ (+125%)	**0.533**$^{**\triangle}$ (+147%)

** and * indicate the improvement over *Iterative Relevance Feedback* is statistically significant according to Wilcoxon signed rank test at the significance level of 0.01 and 0.05; $\triangledown$: the improvement over *Passive* is significant at the level of 0.05; $\triangle$: the improvement over *Active* is significant at the level of 0.05; (+x%) indicates the percentage of improvement over the baseline *Iterative RF*.

be less risky.

- (Compare *Iterative RF* with *Passive* and *Unanchored-Passive*) Distributing the burden of maximizing precision to a classifier is effective, even if the classifier is only involved at the end of the process. Iterative relevance feedback relies on the new query to maximize both precision and recall. By simply keeping the results retrieved by all previous queries and classifying them at the end (by an SVM trained on accumulated judgments), the retrieval performance increases significantly on *all* the data sets (*Passive*). Since the involvement of the classifier releases the burden of the queries to maximize precision, we anticipate that the queries no longer have to be tied closely to the original one. Indeed, even if we strip the effect of the original query from every expanded query (*Unanchored-Passive*), the ReQ-ReC process still yields results comparable to—and sometimes even better than—anchored query expansion (*Passive*). The performance is further improved when the classifier is involved in all the iterations instead of being applied at the end (*Active*).

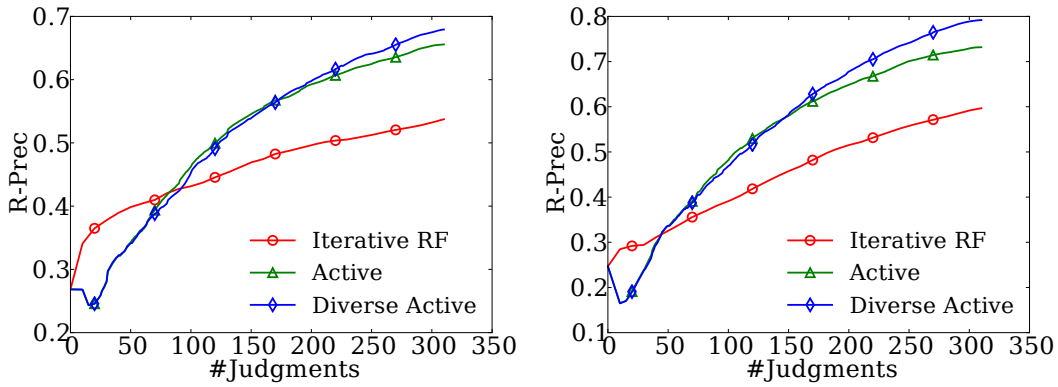
- (*Active*) A straightforward active document selection approach (which picks the documents that the classifier is the least certain about) outperforms picking documents from the top of the ranked list. This is consistent with the observations in literature [135]. By actively selecting documents to present to the user, her effort of labeling documents is significantly reduced.

- (*Diverse Active*) The diverse query expansion method inspired by the weak-tie theory is clearly the winner on all five data sets. By moving the burden of precision to a classifier, the objective of a new query is purely to bring new relevant documents into the pool of retrieved documents. This gives freedom to the queries to expand the search space aggressively, and provides a great opportunity to investigate new algorithms that are particularly suitable for this

goal.

3.5.6 Learning Behavior Analysis

The previous section summarizes the performance of ReQ-ReC methods when the stop criteria are met. To better understand the behavior of a ReQ-ReC process, we provide the following analysis that plots the intermediate performance of three instantiations (*Iterative RF*, *Active*, and *Diverse Active*) throughout the user-interaction process. Note that each topic may accumulate judgments at a different pace and meet stop criteria earlier or later. We interpolate a per-topic curve by a piecewise linear function, and extrapolate it by extending the end-point constantly to the right. These per-topic curves are then averaged to generate the aggregated curve.



(a) MB13 (b) HARD

Figure 3.3: R-Precision vs. Labeling effort

Figure 3.3 plots the performance of each method against the number of documents the "user" has judged so far throughout the ReQ-ReC process, measured using R-precision.

All three curves start at the same point where there is no user judgment. At that point the ranking is essentially based on the original retrieval function (i.e., Dirichlet prior). When user judgments are beginning to be collected, there is a significant gain by iterative relevance feedback. Performance increases rapidly at the first 2 runs (20

judgments), and the growth becomes much slower after that. This is consistent with the findings in literature.

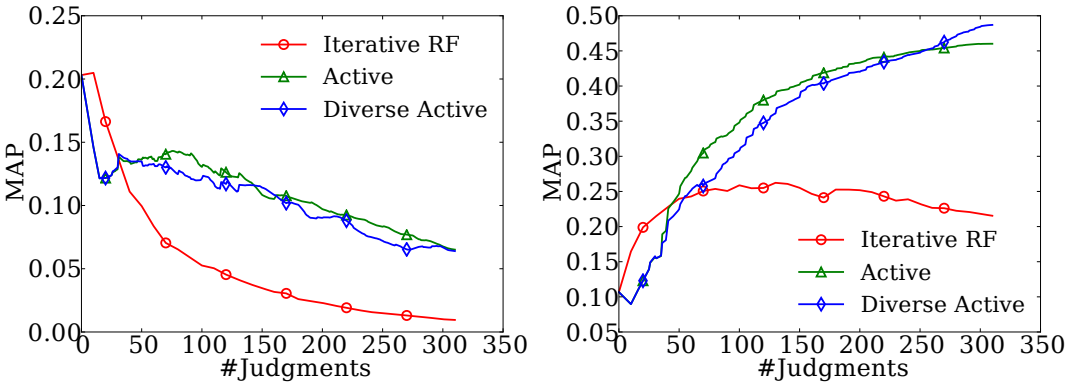
Methods developed under the ReQ-ReC framework (*Active* and *Diverse Active*) do not really take off until we obtain a reasonable number of judgments (50 on the HARD data set and 90 on the microblog data set). This is ascribed to the "cold start" problem of supervised classification. When few labeled documents are available, the performance of a classifier does not outperform a simple ranking function.

As stated before, a ReQ-ReC process targets users who truly seek a high recall of relevant documents and are therefore willing to spend more effort on interacting with the system and labeling more results. Indeed, after the first few iterations, the two methods developed under ReQ-ReC framework improve dramatically and become significantly better than iterative relevance feedback. For the users who are reluctant to label more than 50 documents, conventional relevance feedback may still be a better choice.

The cold start implies that there is considerable room for improving the performance of the ReQ-ReC. For example, a semi-supervised classifier may be used early on to achieve better precision with few training examples.

We also notice that the benefit of *Diverse Active* over *Active* kicks in later in the process, when there are around 150 judgments collected. At that point, getting new relevant documents becomes more challenging, as many documents retrieved by the new query may have already been retrieved by a previous query. At this stage, introducing some diversity to the query expansion brings in considerable benefit. Similar observations are made on the other three data sets.

Another interesting analysis is how well a method works with documents that have not been selected for labeling so far. We are particularly interested in this behavior because we have decided to include all judged documents when measuring the performance of the system (see Section 3.4).



(a) MB13 (b) 20NG

Figure 3.4: Residual Analysis

We plot the residual MAP in Figure 3.4, which is the mean average precision computed purely based on documents that have not been presented to the user so far in the process. In general, the two ReQ-ReC methods (*Active* and *Diverse Active*) do a much better job in finding the relevant documents and ranking them high, even if they are not judged by the user. On the microblog data set, we see that the residual MAP decreases when more documents are presented to and labeled by the user. This may be simply because there are fewer relevant documents remaining in the collection. However, it is also likely due to the fact that the TREC judgments are not complete. There might be many relevant documents that were not judged by TREC at all. If a method successfully finds those documents, its performance may be significantly undervalued simply because we have to treat these documents as negative in computing the metrics.

We are therefore interested in how ReQ-ReC behaves if the data set is fully judged. Looking at the curves on the 20NG, we observe a contrary pattern, where the two ReQ-ReC methods actually enjoy a continuous growth of residual MAP, while the same metric for iterative feedback is still dropping. This is a promising finding that indicates the performance of ReQ-ReC may be underestimated on data sets with incomplete judgments (i.e., TREC data sets).

3.6 Conclusion

We present ReQ-ReC (ReQuery-ReClassify), a double-loop retrieval framework that is suitable for high-recall retrieval tasks without sacrificing precision. The interactive process combines iterative expansion of a query set with iterative refinements of a classifier. The work of maximizing precision and recall is distributed so that the queries increase recall and the classifier handles precision.

The ReQ-ReC framework is general, which includes classical feedback methods as special cases, and also leads to many instantiations that use different combinations of document selection, classification, and query expansion methods. The framework is very effective. Some instantiations achieved a 20% – 30% improvement of mean average precision and R-precision on most data sets, with the largest improvement up to 150% over classical iterative relevance feedback.

In order to clearly illustrate the power of the framework, we have intended to keep all the instantiations simple. It is a promising future direction to optimize the choices and combinations of the key components of the ReQ-ReC framework. Findings from our experiments also indicate possibilities for investigating new classification and query expansion algorithms that are particularly suited to this framework.

CHAPTER 4

Medical Word Sense Disambiguation through Interactive Search and Classification

A vast amount of health data take the form of unstructured text, including biomedical literature, clinical notes, health forum discussions, and health-related news articles. Valuable knowledge and insights can be mined from these text data. For instance, one can evaluate the effectiveness of a treatment on patients with a specific medical condition, the adverse effects of simultaneously taking two or more medications, and public concerns on a health-related policy. Natural language processing (NLP) algorithms are powerful tools to unlock such knowledge from large amounts of text. In this chapter, we consider a specific medical NLP task: word sense disambiguation (WSD).

Resolving word ambiguity in clinical text is critical for many NLP applications. Effective WSD systems rely on training a machine learning based classifier with abundant clinical text that is accurately annotated, the creation of which can be costly and time-consuming. In this chapter, I show that the high-recall retrieval framework ReQ-ReC (ReQuery-ReClassify) in Chapter 3 is versatile and can be repurposed for interactive WSD model training. Using ReQ-ReC, a human expert first uses her domain knowledge to include sense-specific contextual words into requery loops and searches for instances relevant to each sense. Then in reclassification loops, the expert only

annotates the most ambiguous instances found by the current WSD model. Even with machine-generated queries only, the framework is comparable with or faster than current active learning methods in building WSD models. The process can be further accelerated when human experts use their domain knowledge to guide the search process. Its effectiveness is demonstrated using multiple evaluation corpora.

4.1 Introduction

Clinical documents contain many ambiguous terms, the meanings of which can only be determined in the context. For example, the word malaria appearing in a clinician note may refer to the disease or the vaccine for the disease; the abbreviation "AB" may mean "abortion," "blood group in ABO system," "influenza type A, type B," or "arterial blood," depending on the context. Assigning the appropriate meaning (a.k.a., sense) to an ambiguous word, based on hints provided in the surrounding text, is referred to as the task of word sense disambiguation (WSD) [64, 116]. WSD is a critical step towards building effective clinical natural language processing (NLP) applications, such as named entity extraction [137, 134] and computer-assisted coding [43, 53].

Among different approaches to inferring word senses in clinical text, supervised machine learning has shown very promising performance [90, 68]. Supervised machine learning methods typically build a classifier for each ambiguous word, which is trained on instances of these words in real context with their senses annotated, usually by human experts with required domain knowledge. To train an accurate WSD model, a large number of such annotated instances are needed [150], the curation of which can be costly as every instance has to be manually reviewed by domain experts. Many methods have been explored in the past to reduce this annotation cost [88, 158, 83, 98, 23]. Among them, active learning, by inviting human experts to directly participate in the machine learning process, has proven to be an effective approach. The premise of active

learning is its ability to reduce the number of judgment calls that human experts need to make while achieving the same results as having a fully annotated corpus, thus significantly reducing the amount of human labeling needed [23]. As such, how to select the most informative instances to present to human experts to annotate is the key to success for the family of active learning based methods.

Existing active learning methods use different strategies to select the most informative instances for annotation [117]. For example, some select the instance with the least confident prediction or the instance with competing label assignments. However, these strategies suffer from the "cold-start" problem: a number of precisely annotated examples for every sense are usually required to kick off the classifier. Further, a classical active learning procedure does not fully utilize the domain knowledge of human experts. For example, practicing physicians frequently write or read ambiguous words in their notes without any difficulties in conveying or understanding their meaning. They are able to do so largely because of the surrounding context of the ambiguous words; e.g., when AB is used as shorthand for "blood group in ABO system," physicians know that it commonly appears as "blood type AB," "AB positive," or "AB negative." These contextual words are strong indicators of the sense of an ambiguous word, which is invaluable to a WSD model but remains largely untapped by existing active learning methods.

In this chapter, we demonstrate a method that capitalizes on human experts domain knowledge to improve the performance of interactive machine learning. We apply the framework developed in Chapter 3, referred to as ReQ-ReC (ReQuery-ReClassify), to the problem of word sense disambiguation in clinical text. Originally designed for high-recall microblog and literature search [84, 85], ReQ-ReC features a double-loop interactive search and classification procedure that effectively leverages the domain knowledge of human experts. In an outer loop (ReQuery) of the procedure, an expert searches and labels the instances of an ambiguous word along with sense-specific con-

textual words. Then, a ReQ-ReC system helps the expert compose additional search queries by suggesting other potentially useful contextual words. In an inner loop (Re-Classify), the framework requests the expert to annotate the most informative instances selected from those retrieved by all previous queries and then use the results to update the classifier accordingly. An expert can flexibly switch between these two "teaching strategies:" (1) to generate initial examples of a particular sense by launching a keyword search, and (2) to provide fine-grained clarification by labeling the instances selected by the system. Empirical experiments on three different clinical corpora show that this framework is more effective in building accurate WSD models than current active learning methods, even if the expert solely relies on system suggested keywords.

4.2 Interactive WSD in ReQ-ReC Framework

4.2.1 Sample scenario

To illustrate how ReQ-ReC works, let us consider the following scenario. Suppose we have a set of clinical text snippets (e.g. sentences) all containing the word "AB," which means either "blood group in ABO system" or "influenza type A, type B." Our task is to assign the actual sense to each instance. Based on the domain knowledge, a human expert would know that if "AB" co-occurs with the phrase "blood type," then it likely means "blood group in ABO system;" if it co-occurs with the word "influenza," then it likely means "influenza type A, type B." Naturally, the expert would use keywords "blood type AB" to retrieve a set of instances from the text corpus and label them as "blood group in ABO system;" she or he would then search for "influenza AB" and label the retrieved instances accordingly, as shown in Figure 4.1 (a). These context-sense pairs are used as an initial corpus to warm-start the first round of WSD model learning. The learned model will then be applied to predicting unlabeled instances and ask the expert to further clarify a few boundary cases, e.g. "Labs include influenza AB

swab and blood typing," as shown in Figure 4.1 (b). Determining the senses of these boundary cases would allow the model to capture the nuances in language use and quickly improve model accuracy. Later on, the expert may switch between searching for instances and labeling instances. After a few iterations, the expert may start to realize that in phrases such as "AB positive," "AB" also means "blood group in ABO system." Through a new search, she or he can quickly label another batch of instances of "AB positive," which further improves the WSD model, as shown in Figure 4.1 (c).

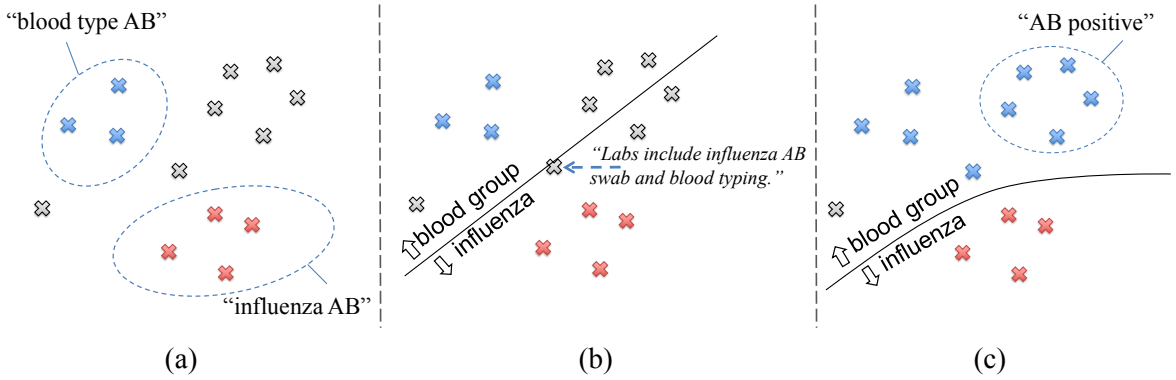


Figure 4.1: An illustrative example of the searching and labeling process of the ambiguous abbreviation "AB."

From this sample scenario several observations can be made. First, keyword search is a natural interface for domain experts to retrieve cases of ambiguous usage of words and to provide high-yielding, targeted annotation. This process can significantly reduce annotation cost, as human experts are only asked to label instances that are most informative to train the WSD model, while avoiding the need of labeling all instances in a corpus, most of which contribute little to improving the model performance. Additionally, search also benefits the learning algorithm: it provides a warm start in generating an initial model, and subsequent searches further refine the model by covering other potential senses of an ambiguous word or additional contextual words. Second, while classifying individual instances retrieved by keyword search is necessary for training the model, it is only able to produce a simplistic model, similar to rules. The ReQ-ReC framework therefore asks domain experts to also clarify boundary cases, which informs

the model on how to weigh the nuances of language use in clinical text for better sense disambiguation. After being re-trained on these cases, the model becomes more robust and more accurate. In addition, answering these clarification questions might also inspire the human expert to come up with new search queries covering other potential senses of an ambiguous word or additional contextual words that might have not been thought about. Therefore, the two stages keyword search and active classification can be used iteratively to inform each other.

4.2.2 Connection to ReQ-ReC

The above scenario resembles the double-loop procedure of the ReQ-ReC framework introduced in Chapter 3. Compared to the scenario in Chapter 3, the key difference is task in consideration: the framework was previously applied in high-recall retrieval, and now we use it for text classification in general, and word sense disambiguation in specific. From a classification perspective, high-recall retrieval is a binary classification task (separating relevant documents from large number of nonrelevant ones).

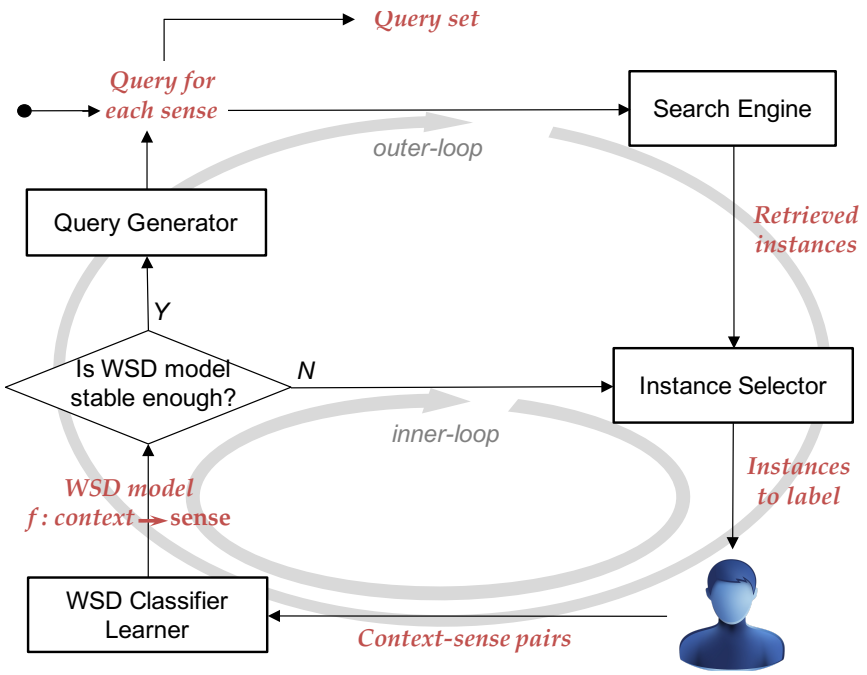


Figure 4.2: The ReQ-ReC framework for word sense disambiguation (WSD).

The procedure is depicted in Figure 4.2. It operates on an inverted index of the context instances so that all keywords, including the ambiguous words and the contextual words, are searchable. The procedure maintains a set of search queries, a pool of retrieved instances, and a WSD model. To start, a human expert first uses her domain knowledge to compose a search query for each known sense, and then the system retrieves an initial set of contexts using the search function. The inner-loop kicks in there, in which the system iteratively presents a small number of instances selected from the current pool of retrieved instances to the expert and asks her/him to assign senses. The WSD model is consequently updated based on the accumulated annotations by the expert, which is then used to reclassify the pool of instances. After a few iterations of the inner-loop, the WSD models predictions stabilize on the currently unlabeled instances. At this point, the outer-loop of the system will kick in to recommend new search queries for each sense (the requery process), aiming to retrieve more diverse instances with additional contextual words. These new search queries will be presented to the human expert for review and for further modification. Then, the system will retrieve a new set of instances using the new queries and add them to the existing pool of retrieved instances. After this requery process, the system will start a new inner-loop and continue to update the WSD model. The learning process ends when the expert is satisfied with the predictions made by the WSD model on those unlabeled instances in the newly retrieved pool.

4.2.3 Instantiating the ReQ-ReC framework

Below we describe the instantiation details of the ReQ-ReC for the WSD task.

(1) Search. In our current research implementation of the ReQ-ReC framework, we use the Lucene Package to build a search index for each ambiguous word [1]. Instances are tokenized with Lucenes StandardAnalyzer. To preserve the original form of ambiguous words ("nursing," "exercises") and negations ("no," "without"), we do

not perform stemming or stopword removal. We use the Dirichlet prior retrieval function with the parameter μ set to 2000, a typical setup in information retrieval literature [161].

(2) WSD classifier. We use logistic regression with linear kernel for WSD classification, implemented by the LIBLINEAR package [39]. If an ambiguous word has two senses, we build a binary classifier; otherwise we build a one-versus-rest multiclass classifier. Logistic regression classifiers output probability predictions $p(y|x;\theta)$ for each sense y and each instance x, which will be used by active learning algorithms (θ is the classification model parameter). We use presence/absence of the all unigrams appeared in the instance as features. For the L_2-regularization hyperparameter C, we set it to 1.0 across all ambiguous words. This setting is comparable to previous reported studies [23].

(3) Instance selection. In the inner-loop, there are multiple possible methods for selecting the next instance for labeling:

a) *Random Sampling.* The algorithm simply selects an instance from the unlabeled pool uniformly at random.

b) *Least Confidence.* The algorithm selects the instance with the least predicted probability $p(y^*|x;\theta)$, where $y^* = \text{argmax}_y\, p(y|x;\theta)$ is the most probable sense. Intuitively, the model has little confidence in predicting the sense of instance x as y^*, therefore it is most uncertain about the sense of x. In this case, expert advice would be needed.

c) *Margin.* The algorithm selects the instance x with the least predicted $p(y_1|x;\theta) - p(y_2|x;\theta)$, where y_1 and y_2 are the most and second most probable senses. Intuitively, the model may not be able to determine if y_1 or y_2 is the appropriate sense, therefore it needs further clarification from the human expert.

d) *Entropy.* The algorithm selects the instance x with the highest prediction entropy $\sum_y -p(y|x;\theta)\log p(y|x;\theta)$. High entropy means that the current WSD model considers any sense assignment as almost equally probable. Expert advice is thus needed to resolve the confusion.

In our implementation, we use the margin based active learning strategy to select instances. Note that all four methods can be launched without the search component, which in effect reduces the ReQ-ReC into a classical active learning system. In the evaluation experiments reported in this study, these methods will be used as baselines for comparison.

(4) Query expansion. In the outer-loop, a new query can either be automatically generated by the system and reviewed and improved by human experts, or be composed manually. In this study, we consider the following two extreme strategies: (a) the system automatically generates a new query based on the current status of the WSD model with no human input; and (b) the human expert composes new queries solely based on her or his domain knowledge. These two strategies represent the worst scenario and a desirable scenario of ReQ-ReC. We use the Rocchios method to automatically generate the next query $\mathbf{q}_y$ for every sense y [110]. The basic premise of Rocchios method is to learn a new query vector that is related to sense y and far away from other senses.

In fact, we hope that the new query $\mathbf{q}_y$ will not be too close to the known contexts in which sense y may appear. This would allow the framework to suggest to human experts other contexts of the sense that might not have been thought of. To achieve this goal, we use the diverse method developed for high-recall retrieval [85], which generates a new query that balances its relevance to the sense and the amount of diverse information it introduces to the current pool of instances. In the rest of the chapter, this strategy is referred to as machine-generated queries or the worst case of

ReQ-ReC.

We also simulate the scenario where human experts use domain knowledge to include contextual words into search queries. To do this, we rank all the contextual words, words appearing in at least one instance of the ambiguous word, by the information gain, i.e. the reduction of uncertainty on the sense of the ambiguous word after seeing a contextual word [156]. Top-ranked contextual words are considered as informative and used as search queries to warm-start the initial model learning. In our experiment, the simulated expert guides the first 6 queries using the top 30 contextual words[1]. As a simulation of domain knowledge, information gain is computed based on the entire set of labeled instances. Note that information gain is only a crude measure for selecting informative contextual words; human experts can do better with their domain knowledge. This simulation would result in an underestimate of the true performance of ReQ-ReC. We denote this scenario as ReQ-ReC with "simulated expert" queries.

4.3 Experiments

4.3.1 Data Sets

In this study, we used three biomedical corpora to evaluate the performance of the ReQ-ReC framework.

The MSH corpus contains MEDLINE abstracts automatically annotated using MeSH indexing terms [68]. Originally, it has 203 ambiguous words, including 106 abbreviations, 88 words, and 9 terms that are a combination of abbreviations and words. Following previous work,14 we only included ambiguous words that have more than 100 instances so we have sufficient data for training and evaluation. This results

[1]The first two queries use the top 10 words; the next two queries use the next top 10 words, and so forth.

in 198 ambiguous words.

The UMN corpus contains 75 ambiguous abbreviations in clinical notes collected by the Fairview Health Services affiliated with the University of Minnesota [95]. 500 instances for each abbreviation were randomly sampled from a total of 604,944 clinical notes. Each instance is a paragraph in which the abbreviation appeared. In this study, we excluded unsure and misused senses in training and evaluation.

The VUH corpus contains 25 ambiguous abbreviations that appeared in admission notes at the Vanderbilt University Hospital [148]. Similar to the MSH corpus, we only retained 24 abbreviations that have more than 100 instances. Each instance is a sentence in which the abbreviation appeared.

The statistics of the three corpora are summarized in Table 4.1. We can see that the MSH corpus has the richest context in an instance and the least skewed distribution of senses for an ambiguous word. Because our main goal in this study was to compare the effectiveness of different learning algorithms, we did not further tune the context window size for each corpus.

Table 4.1: Summary statistics of three evaluation corpora.

	MSH	UMN	VUH
# of ambiguous words	198	74	24
Avg. # of instances per word	190	500	194
Avg. # of senses per word	2.1	5.5	4.3
Avg. # of tokens per instance	202.84	60.59	18.73
Avg. percentage of majority sense (%)	54.2	73.4	78.3

4.3.2 Metrics

In this study, we used learning curves to evaluate the cost-benefit performance of different learning algorithms. A learning curve plots the learning performance against the effort required in training the learning algorithm. In our case, learning performance is measured by classification accuracy on a test corpus and effort is measured by the

number of instances labeled by human experts. For each ambiguous word, we divided its data into an unlabeled set and a test set. When a learning algorithm is executed over the unlabeled set, a label is revealed only if the learning algorithm asks for it. As more labels are accumulated, the WSD model is continuously updated and its accuracy continuously evaluated on the test set, producing a learning curve. To reduce variation of the curve due to differences between the unlabeled set and the test set, we ran a 10-fold cross validation: 9 folds of the data are used as the unlabeled set and 1 fold used as the test set. The learning curve of the algorithm on the particular ambiguous word is produced by averaging the 10 curves. The aggregated learning curve of the algorithm is obtained by averaging the curves on all ambiguous words in an evaluation corpus.

To cope with the cold start problem of active learning algorithms, we randomly sampled one instance from each sense as the initial training set. To facilitate comparison, we used the same initial training set for random sampling and ReQ-ReC. The batch size of instance labeling was set to 1 for all learning algorithms, so that we could monitor the performance improvement by every increment in the training sample.

To summarize the performance of different learning algorithms using a composite score, we also generated a global ALC (Area under Learning Curve) for each algorithm on each evaluation corpus. This measurement was adopted in the 2010 active learning challenge [51]. The global ALC score was normalized by the area under the best achievable learning curve (constant 1.0 accuracy over all points).

4.3.3 Results

We evaluated six interactive WSD algorithms (one trained on randomly sampled instances, three trained using active learning methods, and two using the worst case and the simulated expert case of ReQ-ReQ) on three biomedical text corpora (MSH, UMN, and VUH). Table 4.2 shows the global ALC scores for each learning algorithm on

different evaluation corpora. ReQ-ReC with simulated expert queries consistently outperforms all other methods on all three corpora. On the MSH and VUH corpora, even the worst case of ReQ-ReC achieves higher ALC scores than all existing active-learning algorithms. On the UMN corpus, the worst case of ReQ-ReC is slightly outperformed by the margin active learning algorithm. Compared to other active learning methods, the worst case of ReQ-ReC has the highest ALC scores for 164 out of 297 words across three corpora (55.22%) (129/198 in MSH, 20/75 in UMN, and 15/24 in VUH). With simulated expert queries, ReQ-ReC has the highest ALC scores for 206 out of 297 words across the three corpora (69.36%) (156/198 in MSH, 35/75 in UMN, and 15/24 in VUH).

Table 4.2: Average ALC scores for six learning algorithms.

	MSH	UMN	VUH
Random	0.862	0.854	0.863
Least Confidence	0.899	0.885	0.871
Margin	0.900	0.893	0.872
Entropy	0.899	0.878	0.870
ReQ-ReC worst case	0.904	0.889	0.878
ReQ-ReC expert	0.913	0.894	0.885

Figure 4.3, 4.4, and 4.5 shows the aggregated learning curves of all algorithms on three evaluation corpora, respectively. Results on the MSH corpus present the clearest patterns: the two ReQ-ReC methods learn faster than other algorithms, especially in the beginning stage (first 30 labels). The learning curves of three active learning algorithms are almost identical and much higher than that of random sampling, as previously reported.14 To achieve 90% accuracy, the best active learning algorithm requires 26 labels on average, while ReQ-ReC with simulated expert queries requires only 17 labels, saving 35% labeling effort.

Patterns on the other two corpora are less significant, due to highly skewed sense distributions. In general, ReQ-ReC with simulated expert queries still achieves the best learning curve than other methods, but with a smaller margin, followed by an active

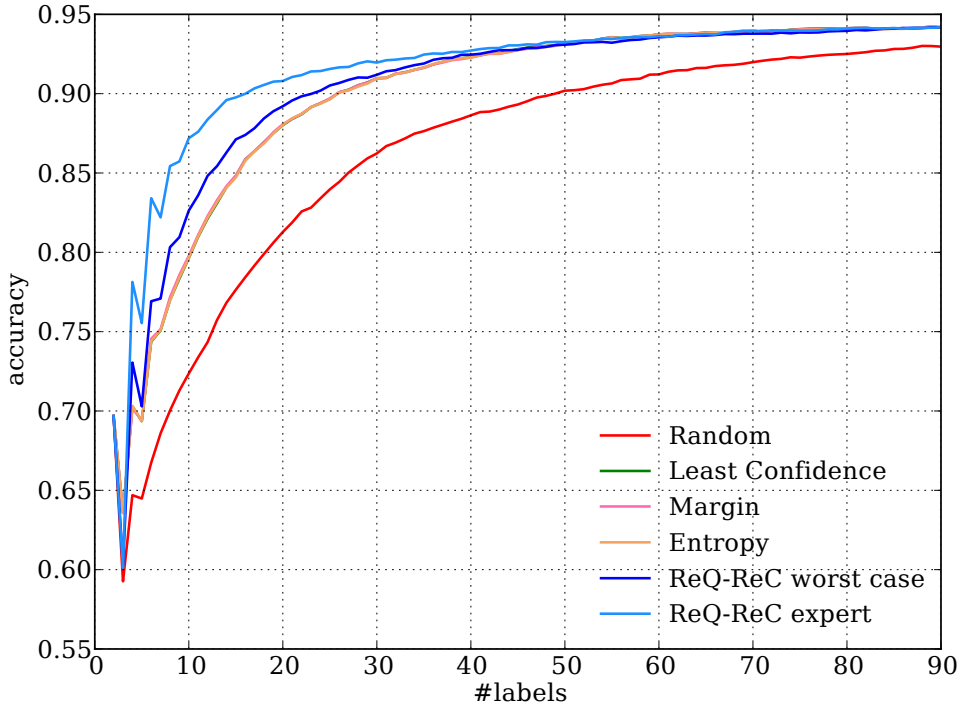


Figure 4.3: Aggregated learning curves of 198 ambiguous words in the MSH corpus.

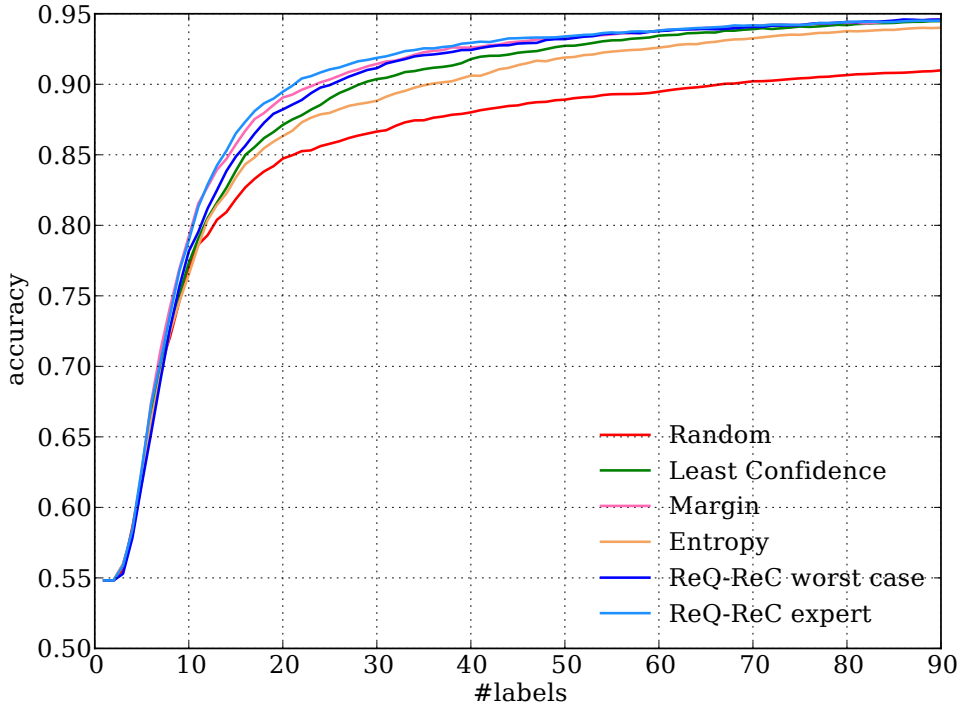


Figure 4.4: Aggregated learning curves of 74 ambiguous words in the UMN corpus.

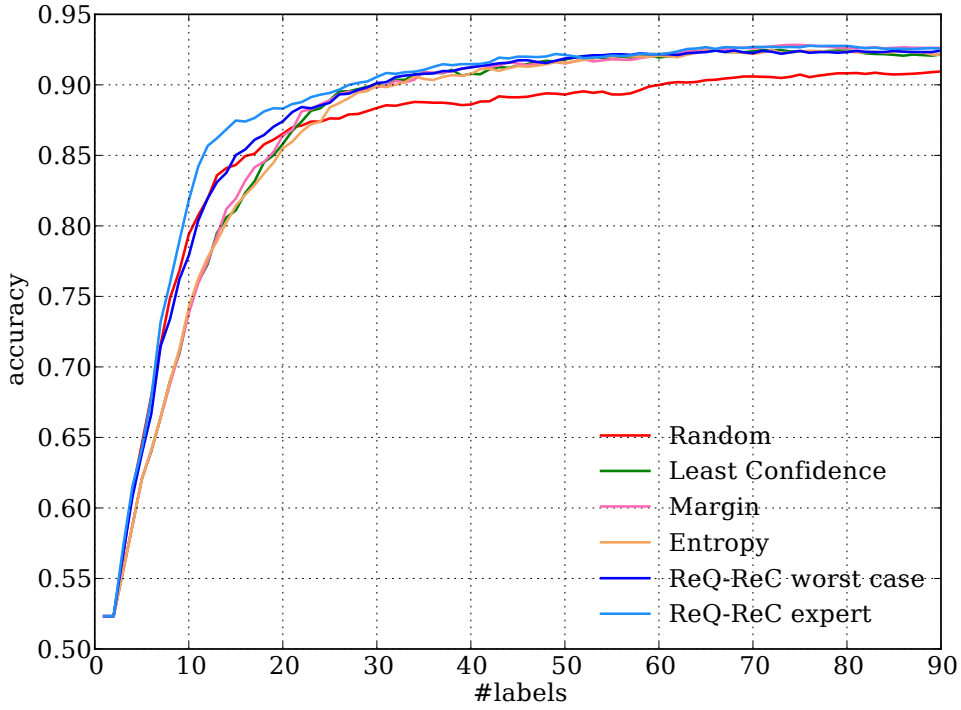


Figure 4.5: Aggregated learning curves of 24 ambiguous words in the VUH corpus.

learning algorithm on the UMN corpus and by the worst case of ReQ-ReC on the VUH corpus. Surprisingly, on the VUH corpus, random sampling learns faster than active learning methods at the very beginning. The benefit of active learning kicks in after 20 labels.

4.3.4 Discussion

The goal of inviting human experts into the machine learning process is to achieve large performance gains with relatively small labeling effort [117]. An active learning process tries to select the next instance such that it brings in as large amount of fresh information as possible for the model to learn from, therefore giving rise to large gains. When asking for the next label, an active learner prefers to ask those instances that represent an unexplored subpopulation and/or instances whose labels the current model is still uncertain about. In contrast, a passive learner randomly picks the next instance

from the unlabeled set, regardless of whether it overlaps with a previously labeled one, or whether the model can accurately guess its label, neither of which make the best use of the labeling effort.

WSD model learning benefits considerably from expert queries as a warm start. When the first few queries are informative contextual words, they construct a pool of representative instances. The initial WSD model learned on this representative pool inherits the domain knowledge from the search queries. Human experts can do even better than the simulated expert in composing these queries. Even when the queries are machine-generated, the query expansion procedure also picks up potentially informative contextual words. On the other hand, active learning methods select instances from the entire corpus rather than a representative pool. In the initial learning stage, models are usually poor and their predictions are unreliable [11]. Thus the uncertain instances selected by such predictions may not benefit the learning as much as the representative ones. As the model becomes more robust in the later learning stage, the clarification questions raised by active learning will make more sense and labeling these instances can better improve the model.

Different characteristics of text documents affect learning process [115]. In biomedical papers that are formally written (the MSH corpus), an ambiguous abbreviation often appears with its full form for clarification purposes, e.g. "high-risk (HR)" and "heart rate (HR)." The co-occurrence of the abbreviation with its full form greatly makes it easier for both the annotation process and the WSD model. In contrast, an ambiguous abbreviation in clinical notes (the UMN and VUH corpora) is almost never expanded to its full form as abbreviations are typically used to save the time of input. A clinical abbreviation can have many senses that are used in many different contexts. As a result, the annotation process for clinical abbreviations requires extensive search and labeling. Compared to active learning, the ReQ-ReC framework can better assist human experts in building clinical WSD models.

When an ambiguous word has many senses, the sense distribution is often highly skewed: one or two major senses cover more than 90% use cases, while many other senses are rarely used. As we can see in Table 4.1, word senses of the two clinical corpora are highly skewed (for more than 4 senses, a majority guess has above 70% accuracy). Skewed sense distribution presents challenge to machine learning [57]. Without abundant labeled instances, it is difficult to learn a WSD model that accurately identifies a rare sense. The classification model will bias towards predicting the major senses and hurt the recall of the rare sense, which becomes an issue for high-stake events such as a rare disease. A straightforward way to cope with the rare sense learning problem is to harvest and label more data for the rare class, for which the first step is to search using contextual words. ReQ-ReC, originally designed for high-recall information retrieval, can be useful in searching for more rare senses.

This study has several limitations. First, in this study we assume the senses of an ambiguous word are known upfront and one instance is already available for each sense, which is a standard setup in the active learning literature. In reality human expert may have knowledge of some but not all of the senses; it is more natural to discover senses on the fly. Second, instead of using the simple bag-of-unigram features, we can use more elaborate features for WSD, e.g. part-of-speech tags, medical concepts (extracted by MetaMap), and word embedding. This could further improve the WSD performance. Third, the framework is only evaluated through simulated experiments and is not evaluated with real users.

4.4 Conclusion

In this chapter, we describe a novel interactive machine learning framework that leverages interactive search and classification to rapidly build models for word sense disambiguation in clinical text. With this framework, human experts first use keyword search

to retrieve relevant contexts in which an ambiguous word may appear to enable targeted, high-yielding annotation. This interactive active learning process, capitalizing on human experts domain knowledge, could therefore significantly reduce the annotation cost by avoiding the need to have a fully annotated corpus. Experiments using multiple biomedical text corpora show that the framework delivers comparable or even better performance than current active learning methods, even if human wisdom is not used to aid in the search process (i.e., all search queries are automatically generated by the algorithm). In future work, we will conduct more evaluation studies to assess the performance of the framework using real-world scenarios and real human experts.

CHAPTER 5

Medical Word Sense Disambiguation through Feature Labeling and Highlighting

In the beginning of an active learning process, the very few examples inevitably train a poor model. Based on the model's inaccurate predictions, the active learning algorithm often acquires low-quality training data, which in turn train a poor model. This vicious cycle haunts the early stage of active learning until sufficient quantity of training data are queried. This problem is known as "cold start" in machine learning literature. To break out from the vicious cycle, one needs to start with either good selection of training data, or a good initial model.

The ReQ-ReC framework in Chapter 4 allows human experts to identify and label typical instances using their domain knowledge, essentially selecting good training data in the beginning stage. This chapter explores the alternative: to have a good initial model. It presents an novel interactive learning algorithm that directly acquires domain knowledge from human experts through new input modalities: labeling and highlighting features. Such knowledge provides a much stronger "warm start" to the initial model than ReQ-ReC with expert queries. A good initial model informs the subsequent active learning algorithm to acquire high-quality training data, which in turn improves the model, thus entering a virtuous cycle. We apply this method in medical word sense disambiguation (WSD) tasks, demonstrating that interactive machine learning has great potential to tap into the rich knowledge in the health domain

to train high-performance medical natural language processing (NLP) models with minimal effort.

5.1 Introduction

Medical documents contain many ambiguous terms, the meaning of which can only be determined from the context. For example, the word "ice" may refer to frozen water, methamphetamine (an addictive substance), or caspase-1 (a type of enzyme); and the acronym "PD" may stand for "peritoneal dialysis" (a treatment for kidney failure), "posterior descending" (a coronary artery), or "police department". Assigning the appropriate meaning (a.k.a. "sense") to an ambiguous word based on the context is referred to as the process of word sense disambiguation (WSD) [64, 116]. WSD is a critical step for many medical NLP applications, such as text indexing and categorization, named entity extraction, and computer-assisted chart review.

The research community has proposed and evaluated many WSD methods in the past, including supervised learning [90, 150, 152], semi-supervised learning [89, 151, 42], and knowledge-driven [88, 158] approaches. Collectively, these studies have shown that a substantial volume of high-quality training data annotated by human experts is required for existing WSD models to achieve desirable performance. However, annotating training data is a labor-intensive process, and the quality may deteriorate as the volume required to be annotated increases [103]. This is particularly true for medical WSD as assigning correct sense for ambiguous medical terms requires great attention and highly specialized domain knowledge.

To address this issue, the machine learning community has been exploring approaches that involve human experts just-in-time during a machine learning process, in contrast to conventional approaches wherein human experts are only involved in creating static annotated training or evaluation datasets. Such approaches are generally

referred to as active learning. An active learning approach [117] prioritizes instances to be labeled and presents to human experts the most informative ones that would help the algorithm achieve desirable performance with fewer iterations. This family of learning methods has shown far superior performance over that of random sampling in medical WSD tasks [23].

In Chapter 4, we described ReQ-ReC, a step further by incorporating an information retrieval component in active learning that allows human experts to identify and label typical instances using their domain knowledge through keyword search. It demonstrated better performance than active learning in medical WSD tasks. However, even though experts are brought into the loop, existing interactive learning approaches still suffer from the "cold start" problem. That is, without any prior knowledge about a new WSD task, an algorithm based on artificial intelligence (i.e., a statistical WSD classifier) needs a large amount of training data to reach a reasonable accuracy. In contrast, well-trained human experts do not have the cold start problem because they come to a WSD task with established domain knowledge, which helps them directly determine the correct sense of an ambiguous word.

In this chapter, we describe a novel interactive learning algorithm that is capable of directly acquiring domain knowledge from human experts by allowing them to articulate the evidence that leads to their sense tagging decisions (e.g., the presence of indicative words in the context that suggest the sense of the word). This knowledge is then applied in subsequent learning processes to help the algorithm achieve desirable performance with fewer iterations, thus solving the cold start problem. That is, besides labeling instances, the expert can provide domain knowledge by two means: (1) to specify informative words of a sense, and (2) to highlight evidence words in labeled instances. These interaction modes enable experts to directly express their prior knowledge and thought process when they perform WSD, without adding much burden. The two channels complement each other: it is sometimes hard to specify strong

informative words a priori, but easier to highlight these words in situ. The statistical classifier can learn from both labeled instances and informative words (i.e. labeled features), and query new labels using active learning.

Simulated experiments on three WSD corpora show that experts domain knowledge gives the model a warm start at the beginning stage, significantly accelerating the learning process. On one biomedical literature corpus and two clinical notes corpora, the proposed algorithm makes better use of human experts in training WSD models than all existing approaches, achieving the state-of-the-art performance with least effort.

5.2 Incorporating WSD Knowledge through Feature Labeling

5.2.1 Instance Labeling vs. Feature Determination

Below, we use an example to illustrate how the interactive learning algorithm works. Suppose the word "cold" (or its spelling variants, e.g., "COLD") is mentioned across a set of medical documents. Depending on the context, it could mean "chronic obstructive lung disease," "common colds," or "low temperature." The task of WSD is to determine the correct sense of each appearance of this word (i.e., each instance of the word).

A human expert performing this task may apply a number of rules based on her or his domain knowledge. For example, she or he may know that when all letters of the word are spelled in capital case, i.e., "COLD," it is more likely the acronym of "chronic obstructive lung disease" than any other possible senses. This judgment could be further strengthened when there are indicative words (or phrases) such as "chronic," "obstructive," or "lung" in the adjacent text. Likewise, if the word is not spelled in all

capitals, and is accompanied by words such as "common," "cough," and "sneeze," it likely means "common cold." For certain senses, contextual cues may appear in other forms rather than indicative words. For example, a numeric value followed by a unit of temperature (e.g. "5 degrees C") may give out that the word "cold" in the current context likely refers to "low temperature," instead of a medical condition.

Unfortunately, such domain knowledge is not leveraged by conventional supervised learning approaches, which only ask human experts to label the sense of the instances of an ambiguous word, rather than capture how human experts make such judgments. In other words, conventional approaches only try to "infer" human wisdom from annotated results, instead of acquiring it directly – even if such wisdom is readily available and can be formally expressed. The interactive learning algorithm described in this chapter addresses this limitation by allowing human experts to create *labeled features* in addition to labeling instances.

A *labeled instance* for an ambiguous word is a [*context, sense*] pair, following the conventional definition in supervised learning. For example, a labeled instance of the word "cold" can be:

`['The patient developed cold and experienced cough and running nose.', common cold].`

A *labeled feature* for an ambiguous word is a [*feature, sense*] pair, where the *feature* is a textual pattern (a word, a phrase, a skip n-gram, or a regular expression in general). The pair encodes the (most likely) *sense* of the ambiguous word if the *feature* appears in its context. For example, human experts can express domain knowledge of the sense of "cold" by creating the following labeled features:

Human experts can also express domain knowledge by highlighting a contextual cue after labeling an instance of "cold", as in

`['The tissue was exposed to a cold environment (5 degrees C).', low temperature].`

```
['COLD' : All cap,              chronic obstructive lung disease]
['chronic' :  Non all-cap,      chronic obstructive lung disease]
['obstructive' :  Non all-cap,  chronic obstructive lung disease]
['lung' :  Non all-cap,         chronic obstructive lung disease]
['common' :  Non all-cap,       common cold]
['cough' :  Non all-cap,        common cold]
['sneeze' :  Non all-cap,       common cold]
...                             ...
```

The highlighted text snippet essentially creates another labeled feature for "cold":

```
['<digit> degrees C', low temperature] .
```

A labeled feature encodes certain domain knowledge that human experts use to solve a WSD task, which can be directly applied to train machine-learning models. As a result, it improves WSD performance and, at the same time, reduces the amount of manual effort required to create a large quantity of labeled instances as training data.

5.2.2 Overall Workflow

The interactive learning algorithm consists of several distinct components; illustrated in Figure 5.1.

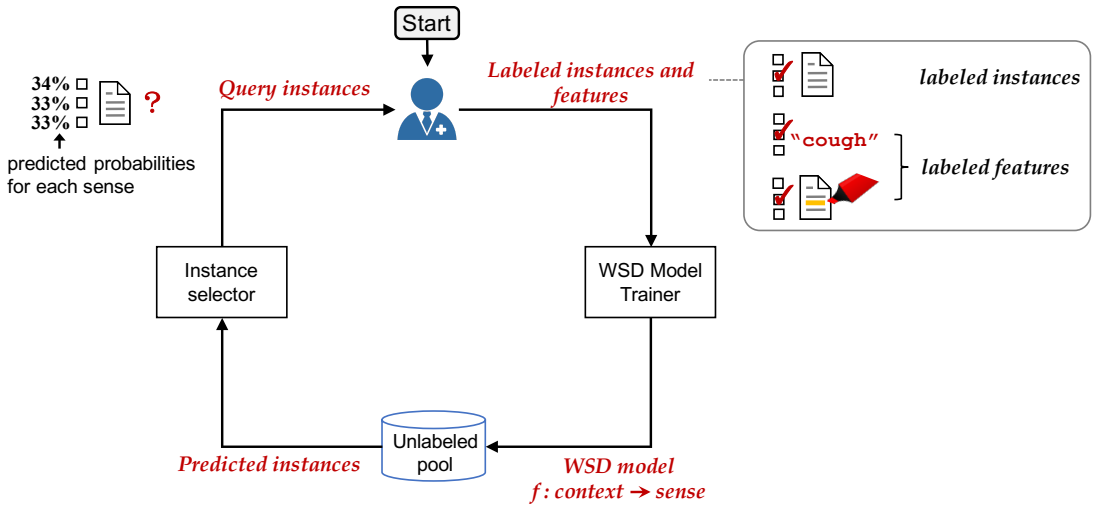


Figure 5.1: Interactive learning with labeled instances and features.

When the human expert can come up with good features for each sense of an ambiguous word, the algorithm can directly use them to train an initial WSD classifier. When such domain knowledge is not available, we assume that the human expert can identify at least one instance for each sense. She or he can then label the instance and highlight contextual cues in that instance. This kicks off the interactive learning process.

The algorithm contains an *instance selector* that determines how to best select instances from an unlabeled pool to present to the human expert. Then, the human expert labels the sense of the instance, followed by potentially suggesting features that were used as the "rationale" for the labeling decision (i.e. feature labeling). Next, the algorithm uses both labeled instances and labeled features to retrain the WSD classifier, then begins another iteration by selecting additional instances for manual labeling till satisfactory WSD result is achieved. This process is described in more detail in the next few sections.

5.2.3 WSD Model Training

The algorithm of training and retraining a WSD model consists of 2 stages: feature representation and parameter estimation.

5.2.3.1 Dynamic Feature Representation

In conventional supervised learning, a model uses a fixed set of features throughout the training process. For text classification, this feature set is often all of the words in the corpus. In our interactive learning algorithm, labeled features may contain arbitrary textual patterns that are difficult to know ahead of time. Rather than trying to include all possible features from the beginning as conventional machine-learning methods do, we use a dynamic feature representation by starting with a set of base features and gradually expanding it as new features emerge. This method helps to prevent severe

overfitting when the size of the feature set is large.

We use presence/absence of unigrams as the base features to represent an instance: $\mathbf{x}^{base} \in \mathbb{R}^V$, where V is the number of distinct unigrams. A labeled feature defines a real-valued function $\phi(\cdot)$ of an instance, such as "1 if the instance contains 'COLD' in all caps; 0 otherwise". Suppose we have m labeled features at iteration t, then an instance is represented by a $(V+m)$-dimension vector $\mathbf{x} = [\mathbf{x}^{base}, \phi^{(1)}, \cdots, \phi^{(m)}]$.

5.2.3.2 Parameter Estimation

We use logistic regression with linear kernel as the WSD classifier. If an ambiguous word has 2 senses, we build a binary classifier, otherwise a softmax multiclass classifier. Logistic regression classifiers output probability predictions in $[0, 1]$, which are then used by the active learning algorithm.

Below, we describe the algorithm for training the logistic regression model. Suppose at a certain iteration, we have l labeled instances $\left\{\left(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}\right)\right\}_{i=1}^{l}$, and m labeled features $\left\{\left(\phi^{(j)}, \mathbf{y}^{(j)}\right)\right\}_{j=1}^{m}$. For an ambiguous word with k senses, $\mathbf{y}^{(i)}$ is a one-hot k-dimensional vector that encodes the assigned sense, and $y_c^{(i)}$ is its k-th dimension. We train a logistic regression model $p(y|\mathbf{x};\theta)$ by minimizing the following loss function (θ denotes the parameters of the model):

$$J(\theta) = \sum_{i=1}^{l}\sum_{c=1}^{k} -y_c^{(i)} \log p\left(y_c|\mathbf{x}^{(i)};\theta\right) + \lambda_1 \sum_{j=1}^{m}\sum_{c=1}^{k} -\tilde{y}_c^{(j)} \log p\left(y_c|\phi^{(j)};\theta\right) + \frac{\lambda_2}{2}\left\|\theta\right\|_2^2 \tag{5.1}$$

$p\left(y_c|\phi^{(j)};\theta\right)$ is the expectation for any instance containing feature $\phi^{(j)}$ to have sense c. Let S_j be the set of instances (both labeled and unlabeled) with non-zero feature values for $\phi^{(j)}$, then

$$p\left(y_c|\phi^{(j)};\theta\right) = \frac{\sum_{i\in S_j} p\left(y_c|\mathbf{x}^{(i)};\theta\right)}{|S_j|} \ .$$

$\tilde{y}_c^{(j)} = (y_c + \epsilon)/(1 + k\epsilon)$ is the smooth version of feature label distribution, because unlike labeled instances, labeled features should be interpreted as preferences rather than as absolute assignments. $\lambda_1 \geq 0$ and $\lambda_2 \geq 0$ are trade-off weights for different loss terms. We set $\epsilon = 0.1, \lambda_1 = \lambda_2 = 1$.

In the loss function (5.1), the first term is the cross-entropy loss on labeled instances; the second term is the cross-entropy loss on labeled features; and the third term is a regularization term of parameter θ. If the loss function only consists of the first and the third term, then it reduces to the loss function of a traditional softmax logistic regression classifier. The second term expresses a preference on the expected behavior of the WSD classifier, i.e., the presence of a feature strongly suggests a label (i.e., the most probable sense). This is a so-called generalized expectation criterion [35]. Because of the second term, (5.1) is a nonconvex function. We use gradient descent to find a local minimum for the model parameter w. In practice, we find the local minimum yields a sufficiently performing classification model.

5.2.4 Instance Selection

The proposed algorithm kicks off the first iteration by a labeled feature for each sense. Once the WSD classifier is trained, active learning can be applied to select a small set of unlabeled instances to present to human experts for labeling. Specifically, we use minimum margin-based active learning as the instance selection algorithm which has shown superior performance in classification settings [117, 143]. It selects the unlabeled instance $\mathbf{x}$ that satisfies the smallest $Q(\mathbf{x}) = p(y_1|\mathbf{x};\theta) - p(y_2|\mathbf{x};\theta)$, where y_1 and y_2 are the most and second most probable senses. Intuitively, the classifier cannot determine whether y_1 or y_2 is the correct sense, therefore it needs to solicit input from human experts.

5.3 Experiments

5.3.1 Data Sets

In this study, we used three established medical corpora to evaluate the performance of the interactive learning algorithm.

The MSH corpus contains a set of MEDLINE abstracts automatically annotated using MeSH indexing terms [68]. Similar to how it was handled in previous work [23, 143], for this corpus, we only included ambiguous words that have at least 100 instances, providing adequate data for training and evaluation. This gave us 198 ambiguous words, including 102 abbreviations, 86 non-abbreviated words, and 10 abbreviation-word combinations.

The UMN corpus contains 74 ambiguous abbreviations from a total of 604,944 clinical notes created at the Fairview Health Services affiliated with the University of Minnesota; each abbreviation has 500 randomly sampled instances [95]. Each instance is a paragraph in which the abbreviation appeared. 4 abbreviations have a general English sense (*FISH, IT, OR, US*).

The VUH corpus contains ambiguous abbreviations from the admission notes created at the Vanderbilt University Hospital [148]. Similar to the MSH corpus, we only retained 24 abbreviations that have more than 100 instances. Each instance is a sentence in which the abbreviation appeared. One abbreviation is a loanword in English (*AD* as in ad lib).

The summary statistics of these three evaluation corpora is shown in Table 4.1. The MSH corpus has the richest context in an instance (i.e., highest average number of tokens per instance), and the least skewed distribution of senses (i.e., lowest proportion of dominating majority senses). Because the main objective of this study was to evaluate the performance of the interactive learning algorithm in comparison with other machine-learning algorithms, we did not further tune the context window size

for each corpus. The three corpora share 3 abbreviations (*SS, CA, RA*). MSH and UMN share another 6 abbreviations. UMN and VUH share another 5 abbreviations. The same abbreviation may have different senses in different corpora.

5.3.2 Baseline Methods

To comparatively evaluate the performance of the interactive learning algorithm, we included three other machine-learning algorithms in the analysis. These algorithms vary mainly based on how labeled instances or features are obtained from human experts.

(1) *Random sampling.* The algorithm selects the next in- stance at random from the unlabeled pool. It starts with one labeled instance for each sense. Later iterations use random sampling to obtain instance labels.

(2) *Active learning.* The algorithm selects the next instance using the minimum margin criterion [117, 23]. It starts with one labeled instance for each sense. Later iterations use minimum margin to obtain instance labels.

(3) *ReQ-ReC expert.* The algorithm extends active learning by inviting human experts to search for typical instances for each sense using keywords [143]. It starts with one labeled feature for each sense. Later iterations use minimum margin to obtain instance labels.

(4) *Informed learning*: the proposed interactive learning algorithm. It starts with one labeled feature (or one labeled instance with a highlighted feature) for each sense. Later iterations use minimum margin to obtain instance labels.

5.3.3 Simulated Human Expert Input

To derive evaluation metrics, we simulated human expert input using labeled data from each corpus, which is a method commonly used to evaluate active learning algorithms[12]. This method reduces potential influences that may be introduced due to performance variation by human experts. More specifically:

(1) Labeling instances: We used the validated labels in these evaluation corpora as the oracle of instance labels.

(2) Labeling features: To implement simulated human expert input (i.e. the "oracle") that ***provides*** labeled features, we computed information gain for each unigram feature using the entire labeled corpus [156], and selected the most informative features as oracle features. A feature is associated with a sense when the feature co-occurs most frequently with the sense. To make it more realistic, we simulated the oracle that knows the q-th best feature among all unigram features, where $q = 1, 5, 10$. This oracle was also used in the "ReQ-ReC expert" algorithm when composing the first search query. The labeled features generated in this way were mostly the words in the definition of each sense.

Since, in reality, a human expert is unlikely able to come up with all features achieving the highest information gain, we also implemented a weaker, supplementary oracle that better resembles true human performance in realistic WSD tasks. It simulates the action of the expert ***highlighting*** a feature in a labeled instance while she or he is doing the annotation. In the first iteration, a random instance in each sense was given to the oracle. It identified the most informative n-gram ($n = 1, 2, 3$) feature in that instance. We used n-grams instead of unigrams to allow the oracle to highlight consecutive words in a sentence. To make the oracle more realistic, we simulated the oracle that knows the q-th best n-gram feature in that instance, where $q = 1, 2, 3$.

5.3.4 Metrics

We used learning curves to evaluate the cost-benefit performance of different learning algorithms. A learning curve plots the learning performance against the effort required in training the algorithm. In the context of this chapter, learning performance is measured by classification accuracy on a test corpus; and effort is measured by the number of instances that need to be labeled by human experts. For each ambiguous word, we split its instances into an unlabeled set and a test set. When a learning algorithm is executed over the unlabeled set, a label is revealed only if the learning algorithm asks for it. With more and more labels becoming available, the WSD model is continuously updated and its accuracy continuously evaluated, producing a learning curve.

To reduce variation of the curve due to differences between the unlabeled set and the test set, we ran a 10-fold cross validation: 9 folds of the data are used as the unlabeled set and one fold used as the test set. The learning curve of the algorithm on a particular ambiguous word is produced by taking the average of the 10 curves. The overall aggregated learning curve of the algorithm is obtained by taking the average of all curves on all ambiguous words in an evaluation corpus.

In reality, human experts are unlikely to provide an inclusive set of features with the highest information gain prior to the annotation process. On the other hand, a well-trained human annotator should be able to identify the best (or one of the best) features after seeing and labeling an instance. Therefore, we hypothesize that the true performance of a human expert will be between the oracle that provides the best feature (best-case scenario) and the oracle that highlights the 3rd best feature in a labeled instance (worst-case scenario). We average the learning curves of the best- and the worst-case scenarios to generate the learning curve of "informed learning".

To summarize the performance of different learning algorithms using a composite score, we also generated a global Area under Learning Curve (ALC) for each algo-

rithm on each corpus. This method was introduced in the 2010 Active Learning Challenge [51]. The global ALC score was normalized by the area under the best achievable learning curve (constant 1.0 accuracy over all points).

To test the significance of performance difference between the algorithms in terms of average ALC scores, we used Wilcoxon signed rank test [145], a non-parametric test for paired examples. We set the type I error control at $\alpha = 0.01$.

5.3.5 Results

5.3.5.1 Aggregated learning curves

The aggregated learning curves obtained by applying each of the learning algorithms on the evaluation corpora, including drill-down analyses of imperfect feature labeling and highlighting oracles, are exhibited in Figures 5.2, 5.3, and 5.4.

Overall, learning curves of informed learning algorithm demonstrated a "warm start" substantially better than the other algorithms evaluated. This is as a result of applying directly acquired domain knowledge from human experts at the beginning of the learning process. The warm start not only helps to achieve desired performance faster with fewer instance labels, but also makes the proposed algorithm (potentially) less susceptible to highly skewed sense distribution. As shown by the curves on the two clinical WSD corpora, UMN and VUH. To reach 90% accuracy, informed learning saved 42% instance labels compared to active learning on the MSH corpus (15 vs. 26), 35% instance labels on the UMN corpus (15 vs. 23), and 16% instance labels on the VUH corpus (26 vs. 31).

5.3.5.2 Area under learning curve

The ALC scores for each corpus and each learning algorithm, as well as the results of statistical significance tests, are reported in Table 5.1. On all three corpora, Wilcoxon signed rank test showed that the ALC scores of informed learning were statistically

significantly better than margin-based active learning. On two corpora (MSH and UMN), the ALC scores of informed learning were statistically significantly better than ReQ-ReC expert, the previous state of the art. These significance results hold even when the feature oracles were imperfect, demonstrating that the proposed algorithm was applicable in a broad range of conditions.

Table 5.1: Area under learning curve (ALC) scores of evaluated interactive learning algorithms. The bottom two sections are variants of Informed learning with different feature labeling (highlighting) oracles.

Learning algorithm	MSH	UMN	VUH
Random sampling	0.8159	0.8146	0.8311
Active learning	0.8676	0.8522	0.8309
ReQ-ReC expert	0.8928	0.8550	0.8524
Informed learning	$0.9094^{*,\dagger}$	$0.9074^{*,\dagger}$	0.8706^{*}
Provide the best feature in Iteration 1	$0.9141^{*,\dagger}$	$0.9122^{*,\dagger}$	0.8792^{*}
Provide 5th best feature in Iteration 1	$0.9087^{*,\dagger}$	$0.9038^{*,\dagger}$	0.8773^{*}
Provide 10th best feature in Iteration 1	$0.9052^{*,\dagger}$	$0.9029^{*,\dagger}$	0.8777^{*}
Highlight the best feature in Iteration 1	$0.9119^{*,\dagger}$	$0.9091^{*,\dagger}$	0.8675^{*}
Highlight 2nd best feature in Iteration 1	$0.9072^{*,\dagger}$	$0.9035^{*,\dagger}$	0.8639^{*}
Highlight 3rd best feature in Iteration 1	$0.9047^{*,\dagger}$	$0.9047^{*,\dagger}$	0.8620^{*}

"$*$" means the score is significant compared to "Active learning" at level $\alpha = 0.01$.
"†" means the score is significant compared to "ReC-ReQ" expert at level $\alpha = 0.01$.

5.3.6 Discussion

5.3.6.1 Warm-start effect

The informed learning algorithm is perfectly positioned to address the "cold start" problem. Active learning works best when the model has a reasonably good understanding of the problem space so that the selected instances are the most informative. At the beginning, the model trained on very few labeled instances can perform poorly and waste data selection. In informed learning, human experts can start the learning process by specifying an informative keyword of a sense, which essentially provides weak labels to many instances containing that keyword, resulting in a "warm start".

It significantly reduces total number of instance labels to reach high accuracy.

5.3.6.2 Error analysis

In Table 5.2, we break down the performance of each algorithm on different subsets of words in three corpora. In the MSH corpus, as abbreviations often co-occur with its full forms, they were easier to disambiguate than non-abbreviated words. The abbreviations in UMN and VUH were harder to disambiguate than those in MSH, because the unbalanced sense distribution presented a challenge to machine learning models.

We studied the cases where Informed Learning (IL) underperformed Active Learning (AL) or ReQ-ReC expert (RR). The main reason was that the simulated feature oracle sometimes provided low-quality labeled features. In fact, words with high information gain could be rare words, not generalizing to many examples; they could also be common words (e.g., that, of), which happened to appear more frequently in one sense than others but were too noisy to be useful in classification. IL works well when a labeled feature is representative of and specific to a sense. We hypothesize that real human experts are more capable of providing such high-quality features than simulated experts.

AL and RR start learning with equal number of instances in each sense, i.e. assuming a uniform prior distribution over senses. As for IL, initial labeled features induce a sense distribution through feature popularity (a frequent feature indicates a major sense), naturally giving rise to a skewed sense distribution. When the true sense distribution is indeed uniform (MSH), AL and RR may have an advantage over IL. However, when the true sense distribution is skewed (UMN and VUH), AL and RR may suffer as they need more instance labels to correct their uniform prior assumption.

In this study, we set 90% accuracy as the target and measured the number of instances required for achieving that performance. In secondary analysis of EHRs

Table 5.2: Average ALC scores of evaluated interactive learning algorithms across different subsets of ambiguous words.

	Average ALC score				ALC advantage (%)	
	Random sampling	Active learning	ReQ-ReC expert	Informed learning	Informed over Active (%)	Informed over ReQ-ReC (%)
MSH						
102 A	0.8617	0.9189	0.9349	0.9548	101/102 (99)	98/102 (96)
10 AT	0.8265	0.8623	0.8922	0.9150	10/10 (100)	10/10 (100)
86 T	0.7603	0.8074	0.8430	0.8549	86/86 (100)	66/86 (77)
UMN						
70 A	0.8145	0.8520	0.8545	0.9076	70/70 (100)	70/70 (100)
4 AT	0.8176	0.8540	0.8635	0.9048	4/4 (100)	4/4 (100)
VUH						
23 A	0.8332	0.8343	0.8552	0.8710	21/23 (91)	18/23 (78)
1 AT	0.7820	0.7535	0.7877	0.8490	1/1 (100)	1/1 (100)

A: abbreviations; **T**: nonabbreviated words; **AT**: abbreviation-word combinations.

data for clinical research, NLP systems with over 90% accuracy are often viewed as reasonable[22-24] and have been widely used. However, for NLP systems that will be used for clinical practice (e.g., clinical decision support systems), higher performance would be required. Therefore, the target performance is dependent on specific tasks. In the future, we will further investigate our approaches when required performance changes.

5.4 Conclusion

This chapter introduces a novel interactive machine learning algorithm that can learn from domain knowledge to rapidly build statistical classifiers for medical WSD. Human experts can express domain knowledge by either prescribing informative words for a sense, or highlighting evidence words when labeling an instance. In addition, active learning technique is employed to query instance labels. Experiments using three biomedical WSD corpora showed that the algorithm delivered significantly better performance than strong baseline methods. Future studies will focus on assessing the performance of the algorithm using real-world scenarios with real human experts.

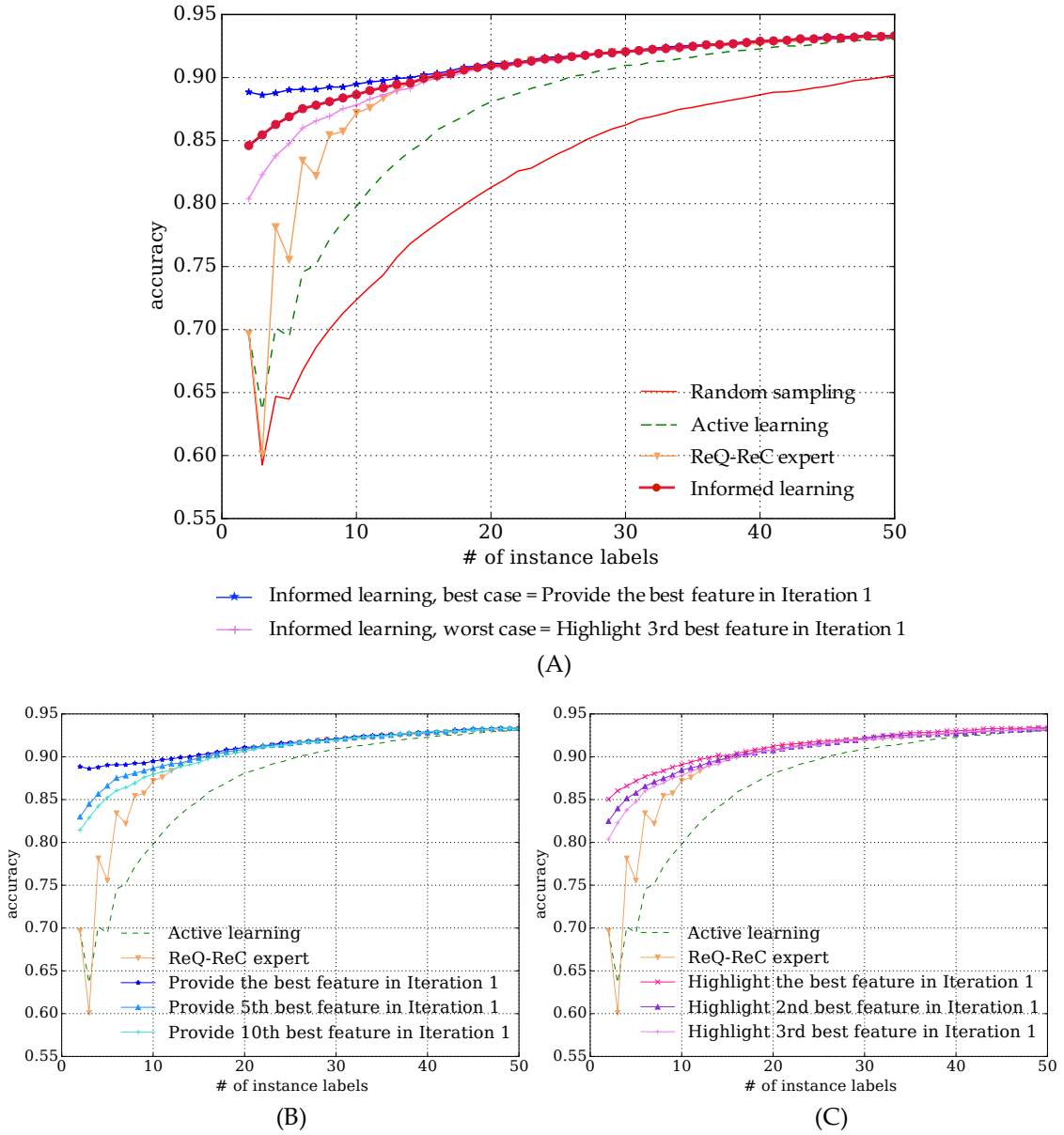


Figure 5.2: Aggregated learning curves of 198 ambiguous words in the MSH corpus, with drill-down analysis of "informed learning".

(A): interactive learning algorithms in comparison, including the best- and worst-case scenarios of "informed learning". To achieve 90% accuracy, "random sampling" required 49 instance labels, and "active learning" required 26 instance labels. "ReQ-ReC expert" used labeled features as instance search queries and required 17 instance labels to achieve 90% accuracy. "Informed learning" directly learned from feature labels and only required 15 instance labels to achieve 90% accuracy. (B and C): drill-down analysis of informed learning using imperfect feature labeling (highlighting) oracles, respectively. Even using imperfect feature labeling oracles, variants of "informed learning" still significantly outperformed both "active learning" and "ReQ-ReC expert," according to Wilcoxon signed rank test (see Table 5.1).

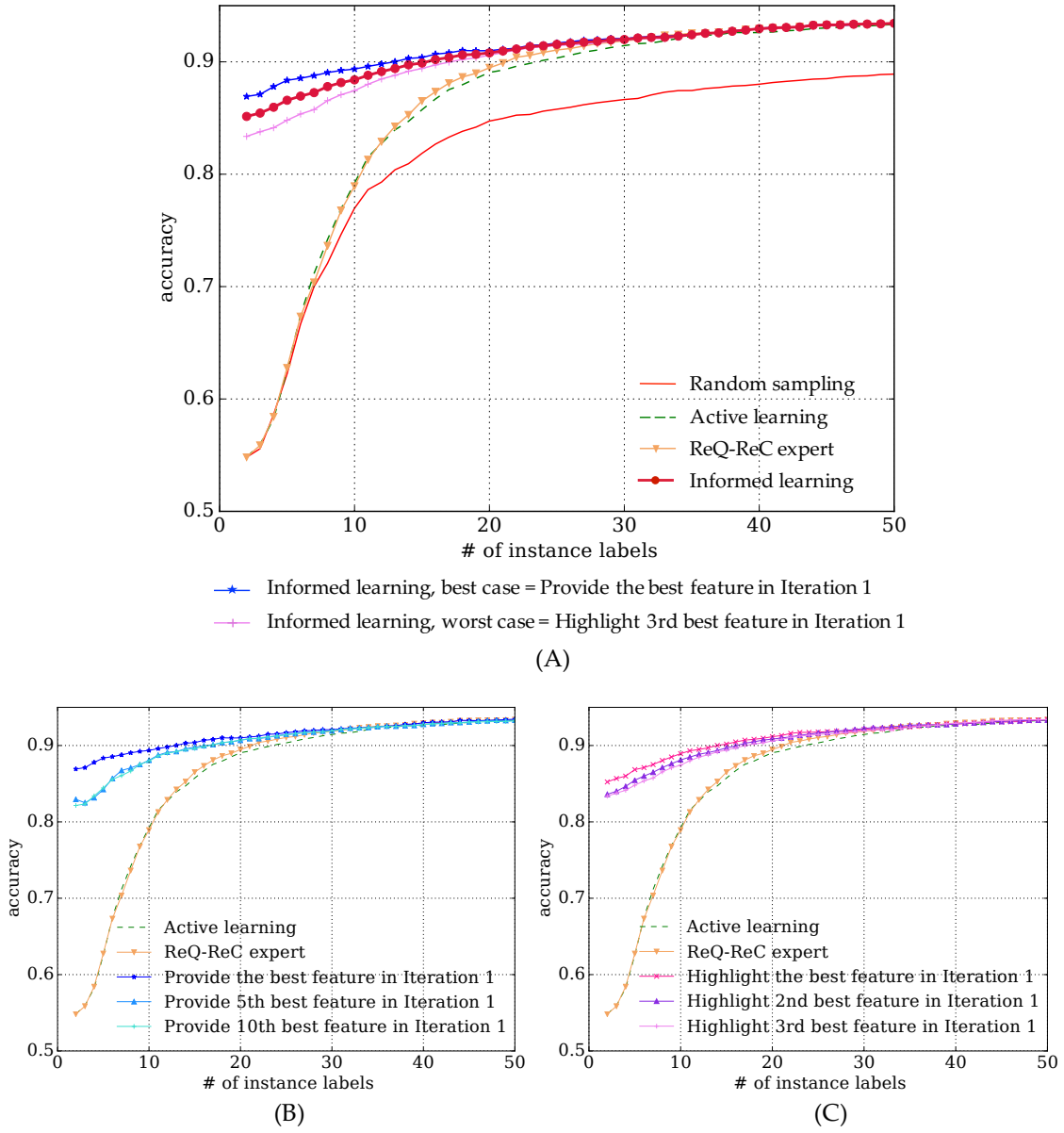


Figure 5.3: Aggregated learning curves of 74 ambiguous words in the UMN corpus, with drill-down analysis of "informed learning".

(A): interactive learning algorithms in comparison, including the best- and worst-case scenarios of "informed learning". To achieve 90% accuracy, "random sampling" required more than 50 instance labels, "active learning" required 23 instance labels, and "ReQ-ReC expert" required 21 instance labels. "Informed learning" required only 15 instance labels. (B and C): drill-down analysis of informed learning of imperfect feature labeling (highlighting) oracles, respectively. Even using imperfect feature oracles, variants of "informed learning" still significantly outperformed both "active learning" and "ReQ-ReC expert", according to Wilcoxon signed rank test (see Table 5.1).

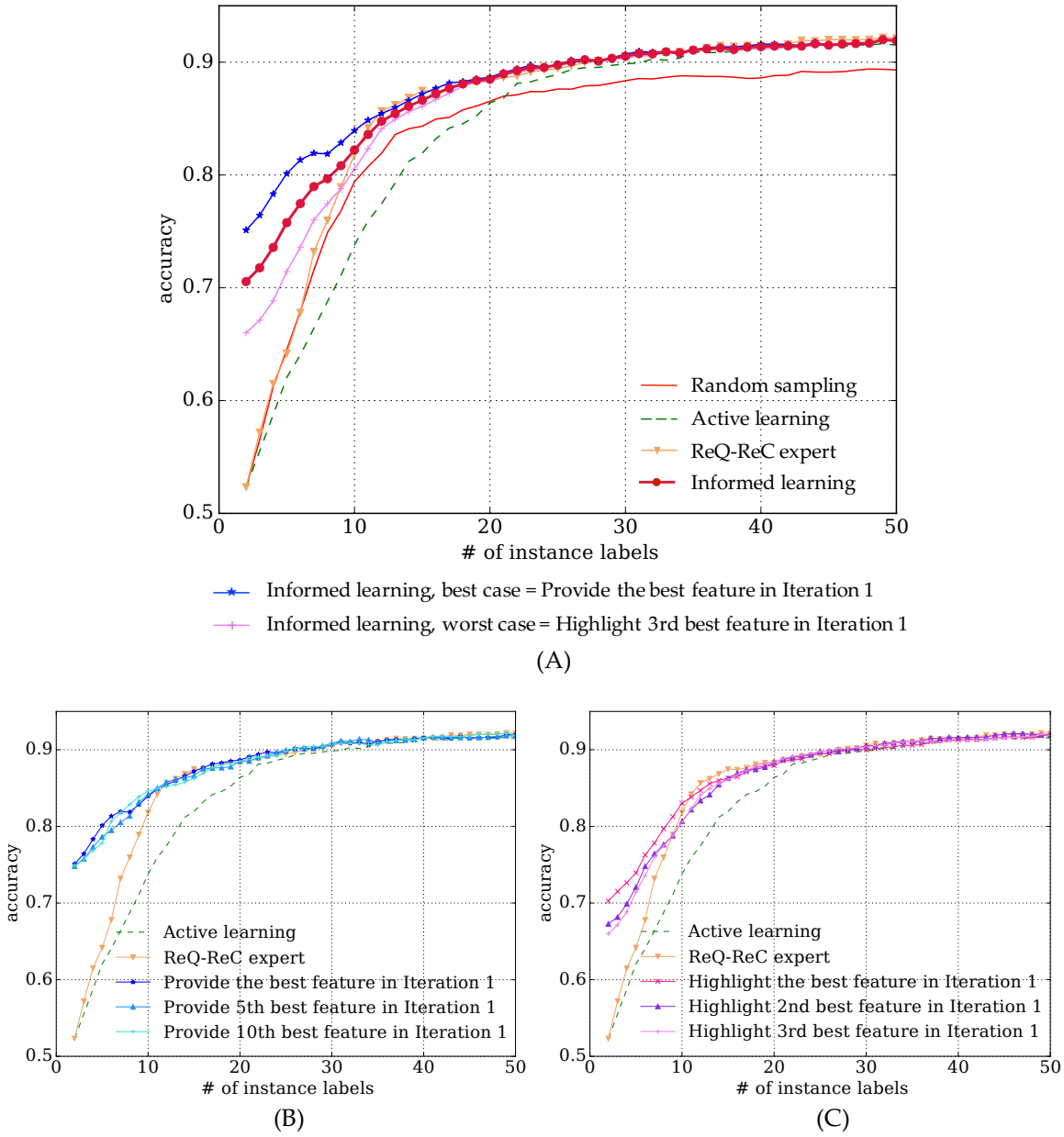


Figure 5.4: Aggregated learning curves of 24 ambiguous words in the VUH corpus, with drill-down analysis of "informed learning".

(A): Interactive learning algorithms in comparison, including the best- and worst-case scenarios of "informed learning". To achieve 90% accuracy, "random sampling" required more than 50 instance labels, "active learning" required 31 instance labels, "ReQ-ReC expert" and "Informed learning" required 26 labels. (B and C): drill-down analysis of learning curves of imperfect feature labeling (highlighting) oracles, respectively. Even using imperfect feature oracles, variants of "informed learning" still significantly outperformed "active learning", according to Wilcoxon signed rank test (see Table 5.1).

CHAPTER 6

A General Framework of Interactive Machine Learning

Previous chapters present novel interactive machine learning algorithms and their applications in various text mining tasks. Over the past few decades, research along this direction has generated an increasing bulk of literature. Each algorithm tends to have different inspirations. The active learning literature survey [117] described six different "query strategy frameworks", but still cannot cover many variants such as active feature learning, dual supervision, and batch-mode active learning. This raises a natural question for researchers in this field: **what is the common underlying principle for the myriad of interactive machine learning algorithms?**

In this chapter, I propose a general framework that unifies many interactive machine learning algorithms. The hope is that such a framework can reveal the essence of these algorithms, just like the structural risk minimization framework reveals the core idea behind supervised learning algorithms [138]. Are interactive learning processes trying to optimize some objective function? Clearly, the object being optimized is not the model parameters, but the selection of data inputs. In light of this, the objective function should be defined over the subsets of a larger data set (or the probability distributions over a larger data set, if the selection is probabilistic). This chapter presents such an objective function, discusses how it connects different interactive machine learning algorithms, and uses the principle to design new algorithms.

6.1 Introduction

The best-performing machine learning models nowadays are also the most data-hungry. In order to train a high-performance machine learning model, one has to prepare a large quantity of labeled examples. While this may be feasible in tasks like movie rating prediction and news topic classification, in many practical scenarios obtaining large-scale training data is extremely expensive and requires highly specialized expertise. This is especially the case for professional domains like medicine and law.

What is the effective way for human experts to produce high-performance machine learning models with low manual effort? Machine learning communities have been proposing many different approaches. Semi-supervised learning algorithms tap into large amount of unlabeled data to reduce label requirement [167]. Transfer learning algorithms make use of knowledge in either previously labeled data or learned models to save labels needed in the current task [99]. Weakly supervised learning algorithms make use of inexact supervision signals that may not take the form of labeled examples but can be acquired at scale with relatively low cost [166, 59].

Interactive machine learning algorithms seek to optimize the label acquisition mechanisms, i.e. the teaching-learning processes between the human and the machine, so as to reduce the overall labeling effort of human experts; refer to Chapter 2 for a comprehensive review. These algorithms aim to achieve the following goals:

- **Asking for labels selectively**: The subfield of *active learning* aims to achieve this goal by developing algorithms that only ask for labels on carefully-selected examples [117]. By asking "smart questions", an active learner can sometimes save a significant portion of labeling effort to achieve high performance compared to random sampling. In all the learning algorithms developed in Chapters 3, 4, and 5, we used an active learning component to reduce labeling effort.

- **Learning from diverse channels**: In many domains, experts' domain knowl-

edge can be expressed in forms such as key entities and relations in knowledge bases, keywords and rules of thumb in everyday practice, other than a set of labeled examples. Previous works in active feature labeling [35, 159, 36] and feature-instance dual supervision [128, 9] aim to select and learn from informative features or rationales, in addition to labeled instances. The keyword search component in Chapter 4 and feature labeling/highlighting component in Chapter 5 are along this line. Learning from such prior knowledge is particularly helpful at the beginning of the learning process.

However, despite the increasing bulk of literature on interactive machine learning algorithms, we do not know if there is a common underlying principle that can summarize many, if not all, of these algorithms. Specifically, it would be ideal if the interactive machine learning processes can be viewed as working towards some goal, e.g. optimizing some objective function.

Recall that many supervised learning algorithms can be unified under the structural risk minimization (or regularized loss minimization) principle [138]. Given training data $S = \{(x_i, y_i)\}_{i=1}^n$, supervised learning algorithms aim to minimize an objective of the form

$$\min_{h \in \mathcal{H}} \sum_{i=1}^{n} \ell\left(h(x_i), y_i\right) + \lambda \Omega(h) \ , \tag{6.1}$$

where $\mathcal{H}$ is the model function class, $\ell(\cdot, \cdot)$ measures the empirical loss, $\Omega(\cdot)$ is a regularization term which measures the complexity of a model, and $\lambda > 0$ is a hyperparameter that balances bias and variance of the learned model.

Many supervised learning algorithms can be seen as different implementations of the above principle, with different choices of the loss function $\ell(\cdot, \cdot)$, the function class $\mathcal{H}$, and the regularization term $\Omega(\cdot)$. In linear regression, $\mathcal{H}$ is the space of linear functions: $h = w^\top x$, $\ell(y, y') = (y - y')^2$. If $\Omega(h) = \|w\|_2^2$, we have ridge regression. If $\Omega(h) =$

$\|w\|_1$, we have lasso regression, which encourages sparse coefficients as solutions. In logistic regression with linear kernel, $\mathcal{H}$ is the space of linear functions, $\ell(y, y') = \log(1 + \exp(-yy'))$ is the logistic loss. In support vector machines with linear kernel, $\mathcal{H}$ is the space of linear functions, $\ell(y, y') = [1 - yy']_+$ is the hinge loss, where $[\cdot]_+$ denotes the positive part. We obtain more complex models when we use function classes more complex than linear functions. Kernel machines use high-dimensional functions implicitly defined by kernels. Neural networks construct high-order nonlinear functions by connecting layers of logistic regression units (so-called "neurons"). Decision trees approximate a complex function with a collection of piecewise constant functions, where each "piece" is defined on an axis-aligned rectangle.

The structured risk minimization principle allows us to gain deeper insights into various supervised learning algorithms. It further inspires us to design new algorithms in a principled way. For instance, we may design a new loss function that better suits a particular task, design a proper model structure that better reflects our understanding of the problem domain in question, or choose a proper level of model complexity to fit a given amount of training data, so that the model generalizes well on unseen data.

Similarly, having a common principle for interactive machine learning algorithms can also help us gain better understanding of various interactive learning algorithms and design new ones in a principled way. Indeed, deploying these algorithms in practice is not without difficulties and frustrations [11], therefore the guidance from an underlying principle is urgently needed. However, current algorithms with theoretical guarantees lack overlap with those widely used by practitioners [3]. In this chapter, I propose a unified framework that connects many interactive machine learning algorithms widely adopted in practice.

The framework formulates an interactive machine learning process as a two-player zero-sum game between the human teacher and the machine learner. This setup naturally depicts learning as an interactive, turn-based, continuous process, which can

facilitate algorithmic design better than the conventional non-interactive, one-off setup in supervised learning. In this game, the machine learner tries to minimize its expected error by adjusting model parameters, while the human teacher tries to maximize the model's expected error by selecting data examples. Surprisingly, this seemingly adversarial setup turns out to help the model converge to the optimal parameters.

Through by the value function of this game, we obtain a general optimization objective for interactive machine learning for finite iterations/samples. It unifies a broad range of algorithms, including uncertainty-based sampling, density-weighted sampling, batch-mode active learning, expected error reduction, and ReQuery-ReClassification (ReQ-ReC, introduced in Chapter 3). This suggests that the objective is very general. To further demonstrate the power of this framework, I discuss new algorithms it inspires, and show promising preliminary results.

6.2 A Two-Player Game

Recent developments in generative adversarial networks (GAN) show that state-of-the-art generative models can be trained in a framework of minimax game [48, 140]. In this game, a generator tries to generate synthetic examples that are very similar to organic ones, and a discriminator (a binary classifier) tries to distinguish which examples are organic and which are synthetic. In the theoretical equilibrium where both parties have sufficiently large model capacity and computational power, the generator generates data from the true data distribution, while the discriminator can only perform random guess.

Inspired by this framework, I formulate the teacher-student interaction in an interactive machine learning process as a minimax game. In this game, a generator tries to generate training examples that are hard to classify, while the discriminator (the classifier) tries to correctly classify examples provided by the generator. In the

theoretical equilibrium where both parties have sufficiently large model capacity and computational power, the generator identifies the Bayes decision boundary and only draw training examples from there, while the discriminator can only perform random guess. This provides the theoretical rationale for the design of a unified objective for interactive machine learning algorithms in Section 6.3.

6.2.1 Game Formulation

For clarity of presentation, let us consider a binary classification setting. Suppose our data (x, y) comes from an underlying joint distribution P_{XY}, $x \in \mathcal{X} \subset \mathbb{R}^d$ and $y \in \mathcal{Y} = \{0, 1\}$. When conditioning on x, we obtain a posterior label distribution $\eta(x) = \Pr(Y = 1|X = x)$. For any given classifier $h : \mathcal{X} \to \mathcal{Y}$, we define the loss function as

$$R(P_{XY}, h) = \mathbb{E}_{(x,y)\sim P_{XY}} \left[\mathbf{1}_{\{h(x)\neq y\}}\right] \tag{6.2}$$

which uses zero-one loss. Note that this loss function depends on both the data distribution P_{XY} and the classifier h. When h has sufficient capacity, $R(P_{XY}, h)$ is minimized by the Bayes classifier

$$h^*(x) = \begin{cases} 1 & \text{if } \eta(x) \geq 1 - \eta(x); \\ 0 & \text{otherwise.} \end{cases} \tag{6.3}$$

The two players in a **discriminative adversarial game** are generator G and discriminator D.

- The generator G selects a probability distribution over $\mathcal{X}$, with density $g \in \mathcal{G}$, where $\mathcal{G}$ is a class of probability density functions with $\text{supp}(g) = \mathcal{X}$.[1] G induces a joint distribution G_{XY} over $\mathcal{X} \times \mathcal{Y}$ by first drawing unlabeled data from g and

[1] $\text{supp}(g)$ is the *support* of distribution g, i.e., $\forall x \in \text{supp}(g), g(x) > 0$.

then sample its label from the posterior label distribution η. The goal of G is to select g (or equivalently, to induce G_{XY}) and generate examples on which D has a high classification error.

- The discriminator D selects a classifier $h \in \mathcal{H}$, where $\mathcal{H}$ is a class of functions mapping from $\mathcal{X}$ to $\mathcal{Y}$. The goal of D is to achieve low classification error on the examples given by G.

G and D play the following two-player minimax game with value function $R(G_{XY}, h)$:

$$\min_D \max_G R(G_{XY}, h) = \mathbb{E}_{(x,y)\sim G_{XY}} \left[\mathbf{1}_{\{h(x)\neq y\}}\right] \ . \tag{6.4}$$

6.2.2 Theoretical Results

We assume that G can draw infinite data $x \sim g$ and query the corresponding labels; g and h can take arbitrary function form (their model classes have sufficient capacity), and $\eta(x)$ is continuous on $\mathcal{X}$. We have the following results (see Appendix A for proofs):

Theorem 6.2.1. *For any fixed distribution selected by G, D's optimal strategy is to choose the Bayes classifier h^*.*

Theorem 6.2.2. *For any $\epsilon > 0$, the generator G's can select a distribution g that achieves $R(G_{XY}, h^*) = 1/2 - \epsilon$.*

Please see Appendix A for the proofs of the theorems. The theoretical results show that if G samples data points with high probability from the Bayes decision boundary (region B, where $\eta(x) = 1/2$), then even the optimal classifier will make errors almost half of the time. This characterizes the equilibrium state, where G concentrates on the Bayes decision boundary and D selects the Bayes classifier. The value of the game is arbitrarily close to $1/2$ from below.

The minimax formulation has been used to analyze active learning in theoretical work [20, 54]. Reinforcement learning algorithms (multi-armed bandits [45]), submodu-

lar function maximization [47], and max-min formulation [61], are used to solve active learning. Recently, the minimax formulation is proposed to develop data selection procedures for curriculum learning [164], where the data set is *fully labeled*, unlike in an active learning setting. To our best knowledge, the literature has not explicitly connected active learning to a zero-sum game with two players as we described here.

6.3 A Unified Objective for Interactive Machine Learning

In this section, we consider the interactive learning scenario of the game in Section 6.2.1, where G samples finite number of data points and D selects a classifier adaptively at each round. We seek for optimal strategies for both G and D in finite-sample scenario, aiming to reach the equilibrium quickly. This is known as the best response dynamics to find the equilibrium [96].

The above game-theoretic analysis shows that when the sample size and model capacities are infinite, both G and D should optimize – maximize or minimize, respectively – the *expected* loss over a data distribution. In the finite-sample scenario, both parties optimize an *empirical* mean instead of an expectation. To mitigate the risk of high variance in a finite sample, we need to further regularize the empirical mean.

The discriminator D can follow the convention of supervised learning. In each iteration, D's strategy is to train a classifier h that minimizes the regularized empirical loss on training set S produced by G:

$$h \leftarrow \operatorname*{argmin}_{h \in \mathcal{H}} \sum_{i=1}^{|S|} \ell(h(x_i), y_i) + \lambda \Omega(h) \ . \tag{6.5}$$

The regularizer $\Omega(h)$ puts preference on the classifier h, such as being less complex. By trading variance for bias, it can effectively reduce overfitting when S is small. As S

gets larger, D can decrease λ to obtain a consistent h. This is well-studied in structural risk minimization [138].

As for the generator G, we have to redefine its strategy space in finite samples. We assume that the unlabeled set $\mathcal{X}$ is sufficiently large so that the distribution g can be well approximated by a subset $S \subset \mathcal{X}$.[2] What does it mean for G to play optimally? Inspired by D's strategy, we can design G's strategy symmetrically: it aims to sample a training set S that *maximizes the regularized guessed loss* made by the classifier h:

$$S \leftarrow \operatorname*{argmax}_{S \subset \mathcal{X}} \sum_{i=1}^{|S|} \ell(h(x_i), y_i^*) + \mu\Phi(S) \ . \tag{6.6}$$

y_i^* are guessed labels as we don't have true labels for all data in $\mathcal{X}$. In Section 6.4.1 below, we show that the guessed loss $\ell(h(x_i), y_i^*)$ actually corresponds to the selection criteria of different uncertainty sampling strategies. The *set regularizer* $\Phi(S)$ puts preference on S. Without the regularizer, Eq. (6.6) reduces to uncertainty sampling, where the greedy strategy is to select the next example with maximum guessed loss. The interesting question is what kind of preference should $\Phi(S)$ put on S.

Recall that the model regularizer $\Omega(h)$ prefers *less complex models*, e.g. h with a smaller norm $\|h\|_2$ for linear function classes. Again using the symmetry argument, $\Phi(S)$ should prefer *more complex sets*, e.g. a set of feature vectors spanning a larger volume, or a diverse set that better covers the data space. As S gets larger, the generator G can decrease the weight μ to concentrate more on uncertain regions, eventually on the decision boundary, where the data is complex in a task-specific manner, not complex *a priori*.

Another way to understand the rationale behind the set regularizer $\Phi(S)$ is to recall the ultimate goal of learning a classifier: to minimize the *expected* loss, the loss on the joint distribution P_{XY} as defined in (6.2). However, this quantity cannot be

[2]With some abuse of notation, we use S to denote a subset of unlabeled data $\mathcal{X}$ as well as the labeled set $\{(x, y) : x \in S\}$.

directly measured as we do not have access to P_{XY} in practice. As an approximation, we minimize the *empirical* loss over a finite sample S. To better approximate the expected loss, S should start as a *representative* sample of the entire population of P_{XY}. Therefore $\Phi(S)$ is a prior measurement of the representativeness of S with respect to P_{XY}. This rationale implies that, importantly, $\Phi(S)$ should encourage not just a wide coverage of the data manifold, P_X, but also a wide coverage of labels over the data manifold, $P_{Y|X}$. The former can be achieved by considering the data manifold structure (e.g. through a data similarity matrix or clustering structure). The latter is often hard to measure *a priori*, since $P_{Y|X}$ is what we set out to learn. However, in some cases we may still have weak prior of $P_{Y|X}$, e.g. a data set organized under a related classification scheme, or a pretrained model on related tasks.

G's strategy can also be interpreted as a reinforcement learning policy: the two terms in (6.6) are a combination of exploitation and exploration. On the one hand, G aims to identify the regions in the feature space with the maximum expected loss. The term $\sum_{i=1}^{|S|} \ell(h(x_i), y_i)$ corresponds to this exploitation. On the other hand, G has to explore the feature space and identify all regions surrounding the Bayes decision boundary, where D would make most mistakes. $\Phi(S)$ corresponds to this exploration.

The optimization problem in (6.6) is called **regularized loss maximization**. The data selection criterion is now a set function, instead of a pointwise ranking criteria adopted by many active learning algorithms. The set regularizer $\Phi(S)$ gracefully handle the "cold start" period: at the very beginning of the game when we have no labeled data to train h, $\Phi(S)$ will guide G to sample diverse and representative data points.

6.4 Explaining Existing Algorithms

In this section, we show that the general framework unifies a wide range of existing active learning algorithms and the ReQ-ReC introduced in Chapter 3.

6.4.1 Uncertainty Sampling

Uncertainty sampling strategies can be understood as selecting the next unlabeled data point that maximizes the guessed loss $\ell(h(x_i), y_i^*)$ of the current classifier h. Table 6.1 unifies three uncertainty sampling strategies through the lens of different loss functions. The three equivalent losses are *zero-one loss*, *hinge loss*, and *cross entropy loss*, respectively. They are guessed losses because they assume that the current classifier h produces the true label posterior distribution.

Table 6.1: Uncertainty sampling algorithms: pick $x = \text{argmax}_x\, q(x)$.

Strategy	Criterion $q(x)$	Equivalent guessed loss
least confident	$1 - p_h(y_1\|x)$	$\mathbb{E}_{p_h(y\|x)}\left[\mathbf{1}_{\{y^* \neq y\}}\right]$
margin	$1 + p_h(y_2\|x) - p_h(y_1\|x)$	$\max(0, 1 + \max_{y \neq y_1} p_h(y\|x) - p_h(y_2\|x))$
entropy	$-\sum_y p_h(y\|x) \log p_h(y\|x)$	$\mathbb{E}_{p_h(y\|x)}\left[-\log p_h(y\|x)\right]$

y^* and y_1 denote the predicted most probable label;
y_2 denotes the second most probable labels.

Uncertainty sampling is a greedy strategy, since it does not optimize the representativeness of selected data as suggested by the regularized loss maximization objective (6.6). This means that uncertainty sampling only trains a model that generalizes locally but not globally. Because of this, uncertainty sampling algorithms are often distracted by local noisy labels and ignore the entire landscape of the task, leading to slow learning rates. In Section 6.5.1.1, we follow the guidance of (6.6), fix the myopic behavior of uncertainty sampling by adding a regularization term, and show promising preliminary results.

6.4.2 Density-Weighted Sampling

Density-based methods aims to query data points that are "representative" and "uncertain". The information density criteria selects data point with the maximum product of uncertainty and density [120]. The DUAL strategy starts with density-based sampling and gradually moves to uncertain regions [33]. In our framework, the guessed

loss corresponds to uncertainty sampling, and the set regularizer $\Phi(S)$ corresponds to density estimation. This is because $\Phi(S)$ evaluates the "coverage" of selected data points S, which will attain higher values if S consists of data points from *dense and diverse* regions.

6.4.3 Batch-Mode Active Learning

The objective function (6.6) is defined on a set instead of a single data point, therefore is a general batch-mode active learning objective. The two competing objectives optimized by batch-mode active learning algorithms are uncertainty and representativeness [117, 60, 144], which resonates well with the guessed loss and set regularizer terms in (6.6). The unified objective is also more general, as it is not specific to a particular classification model family, number of classes, or loss function, which were assumed in previous works on batch-mode active learning.

6.4.4 ReQuery-ReClassification (ReQ-ReC)

The ReQ-ReC algorithm (Chapter 3 and 4) alternates between two loops: the inner loop performs uncertainty sampling, and the outer loop computes a diverse query and select more unlabeled data points into the unlabeled pool. This double-loop process can be understood as taking alternating steps to increase the objective (6.6): the uncertain data points in the inner loop increase the guessed loss, while the diverse and relevant documents retrieved in the outer loop increase the set regularizer.

6.4.5 Expected Error Reduction

Expected error reduction retrains the current model h with the guessed label of a data point x, and selects the x that causes the maximum expected error reduction [111, 169]. Below we show that it is related to maximizing the guessed loss *one step ahead*.

Let us denote the current labeled set as S, and $S_X = \{x : (x, y) \in S\}$. U_X is the set of currently unlabeled data, therefore all unlabeled data $\mathcal{X} = S_X \cup U_X$. In each round of active learning, one element is removed from U_X, labeled, and enters S_X. After training on S, we have our current model h. Denote the guessed loss of model h on a set of unlabeled data A as $L(h, A) = \sum_{x_i \in A} \ell(h(x_i), y_i^*)$. Then $L(h, U_X) = L(h, \mathcal{X}) - L(h, S_X)$.

In a look-ahead step, a data point $x_k \in U_X$ will be assigned a label $h(x_k)$ guessed by the current model h. Let us use h_k to denote the model retrained on the pseudo training data set $S \cup \{(x_k, h(x_k))\}$. $S_k = S_X \cup \{x_k\}$ and $U_k = U_X \setminus \{x_k\}$ are the new labeled and unlabeled X-data sets.

Expected error reduction aims to select x_k according to:

$$x_k \leftarrow \operatorname*{argmin}_{x_k \in U_X} L(h_k, U_k) \tag{6.7}$$

$$= \operatorname*{argmin}_{x_k \in U_X} L(h_k, \mathcal{X}) - L(h_k, S_k) \tag{6.8}$$

Since h_k is trained on the same set as h, plus a data point labeled by h itself (self-confirming), it is guaranteed that h_k achieves a lower guessed loss on all the data $\mathcal{X}$. That is, $\forall k, L(h, \mathcal{X}) \geq L(h_k, \mathcal{X})$. Therefore $\forall k$,

$$\underbrace{L(h, \mathcal{X})}_{\text{constant w.r.t. } k} - L(h_k, S_k) \geq L(h_k, \mathcal{X}) - L(h_k, S_k) \ . \tag{6.9}$$

The left-hand side (LHS) is an upper bound of the right-hand side (RHS), and the RHS is the objective function (6.8). Because the first term in the LHS is a constant with respect to k, selecting x_k to minimize the LHS amounts to:

$$x_k \leftarrow \operatorname*{argmax}_{x_k \in U_X} L(h_k, S_k) = \sum_{x_i \in S_k} \ell(h_k(x_i), y_i^*) \tag{6.10}$$

Equ. (6.10) can be understood as "maximizing the guessed loss one step ahead", which minimizes an upper bound of the original criteria Equ. (6.7) or (6.8).

Both $L(h_k, U_k)$ and $L(h_k, S_k)$ are "future" guessed losses. It is the selection criteria for one unlabeled data point x_k. To calculate it, one needs to retrain a whole model h_k. Therefore to execute the expected error reduction algorithm, one needs to retrain as many models as the number of unlabeled data points in U_X: $h_k, k = 1, \cdots, |U_X|$. This can be very computationally expensive, because the unlabeled data set U_X is usually very large.

6.5 Design Implications for New Algorithms

Previous sections have shown that the regularized loss maximization framework is quite general and unifies many existing learning algorithms. This section presents two new interactive learning algorithms as novel instantiations of the general framework.

6.5.1 The Representativeness Term

In the objective function (6.6), the set function $\Phi(S)$ encourages high coverage, or representativeness, of the selected set S. $\Phi(S)$ serves as a prior when the guessed loss is inaccurate, especially at the beginning of the learning process.

$\Phi(S)$ can be defined by the unlabeled data manifold structure (e.g. revealed by the pairwise data similarity matrix or the cluster structure). Intuitively, it should encourage S to cover dense and diverse areas of the feature space $\mathcal{X}$, i.e. to explore the space. It can also be defined by a set of known topics/aspects, e.g. a part of a knowledge graph related to the current task. Intuitively, it should encourage S to cover heterogeneous X-regions that might bear different labels. In different applications scenarios, we should seek for a $\Phi(S)$ that truly captures the notion of representativeness in that scenario. For instance, in high-recall retrieval, a good $\Phi(S)$ should aim to cover

all possible aspects of the query. In a word sense disambiguation task, $\Phi(S)$ should aim to cover all senses of the ambiguous word. Below we discuss concrete implementations of $\Phi(S)$.

6.5.1.1 Representativeness Defined on Data Manifold Structure

A function family well-suited for this purpose is the subset selection objective, such as those used in submodular maximization algorithms [72] and determinantal point processes [75]. Concrete examples include the coverage function, the facility location, and the mutual information function. They all prefer representative and diverse subsets. A notable advantage in terms of computational complexity is that monotone submodular set functions permit greedy and near-optimal solutions.

(1) Coverage function: for every data point $x \in \mathcal{X}$, its neighborhood is a set of points $\Gamma(x) \subset \mathcal{X}$, including itself. Coverage function is the sum of weights of every point in S and their neighbors:

$$\Phi(S) = \sum_{e \in \cup_{x \in S} \Gamma(x)} w(e) \ . \tag{6.11}$$

When $w(e) = 1$, it counts the number of elements covered by S. This function is convenient for graph data representation but less so for vector data representation, unless we can clearly define the neighborhoods using a good similarity function.

(2) Facility location: let $s(\cdot, \cdot)$ measure the similarity between data points. Facility location function measures the similarity between S and the pool of unlabeled data $\mathcal{X}$:

$$\Phi(S) = \sum_{e \in \mathcal{X}} \max_{x \in S} s(e, x) \ . \tag{6.12}$$

When $s(e, x)$ is interpreted as the utility of opening a facility x for a customer

e, this function measures the total utility of opening a set of facilities S for all customers $\mathcal{X}$ if a customer only goes to the facility with maximum utility.

(3) Mutual information: we can model all feature data points by a Gaussian process, where we define an appropriate covariance function $k(\cdot, \cdot)$ that measures the relation between any two data points. Given a finite data set X, we have the Gram matrix K with elements $k_{i,j} = k(x_i, x_j), \forall x_i, x_j \in X$. We aim to select subset S such that S and $U = X \backslash S$ have the maximum mutual information. In other words, observing S tells us a lot about the unobserved U.

$$\Phi(S) = I(S; U) = H(S) + H(X \backslash S) - H(X) \tag{6.13}$$

$$= \log\det(K_S) + \log\det(K_{X \backslash S}) - \text{const.} \tag{6.14}$$

Since the determinant of a matrix can be viewed as the volume of subspace spanned by its column vectors, we can view this function as selecting columns in the full kernel matrix K that span the largest volume.

Preliminary Results

To test the regularized loss maximization framework, we conducted preliminary experiments on the `20NewsGroup` dataset, where simple uncertainty sampling methods are known to underperform random sampling (e.g. see Figure 2 and 4 in [144]).

We implemented regularized loss maximization using hinge loss of the multiclass classifier (Table 6.1, 2nd row). The regularizer is the *facility location* function (6.12), with regularization weight $\mu = 1/|S|^2$. The learning curves are shown in Figure 6.1 and 6.2. The 20 classes in `20NewsGroup` dataset has non-trivial overlap each other, therefore the class boundaries are not "clean". Simple uncertainty sampling methods suffer in such case because the queried examples may be noisy boundary cases that are not representative enough to carry useful information for discerning the classes.

Regularized loss maximization strikes a balance between exploration (querying representative examples) and exploitation (querying uncertain examples), outperforming random sampling. Figure 6.2 shows that regularized loss maximization is the fastest in discovering new classes, as a result of explicit exploration.

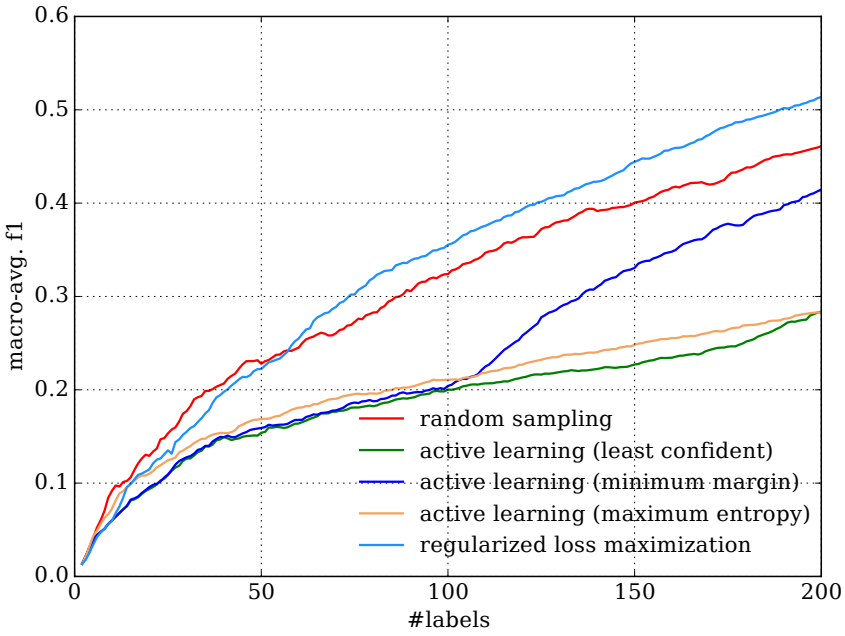


Figure 6.1: Learning curves on the `20NewsGroup` data set. Three simple uncertainty sampling methods underperform random sampling. In contrast, regularized loss maximization outperforms random sampling by an increasing large margin after querying 50 labels.

6.5.1.2 Representativeness Defined on Semantic Categories

Given the classification task at hand, suppose we can identify a related subgraph inside a general knowledge graph that well covers the task domain. Let the topics in the knowledge subgraph be V, where each node $v \in V$ is a known semantic category. Given an unlabeled data point x, also suppose we have a similarity function or a topical classifier $s(v, x)$ to evaluate the similarity between data x and topic v. We can use, for

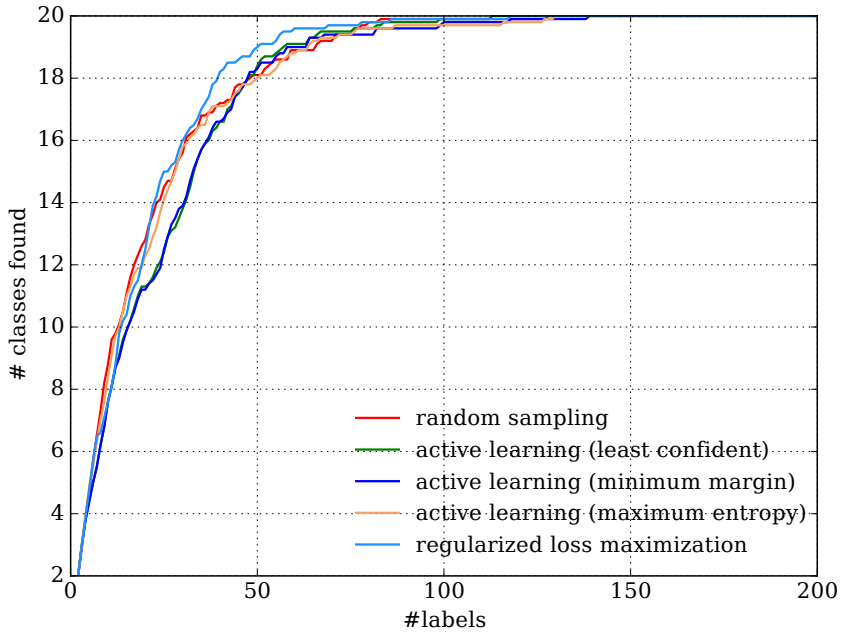


Figure 6.2: Regularized loss maximization discovers new classes faster than other methods, because of its balanced exploration-exploitation strategy.

example, the facility location function to measure the coverage of S over all topics V:

$$\Phi_{\text{topic}}(S) = \sum_{v \in V} \max_{x \in S} s(v, x) \ . \tag{6.15}$$

The representativeness term $\Phi(S)$, together with the guessed loss term in Eq. (6.6) defines a novel active learning algorithm. We can further use nonnegative linear combinations of (6.12) and (6.15) as the representativeness term, such that the queried data points are from dense and topically diverse regions. This enables the algorithm to explore the data space efficiently.

6.5.2 Text Classification with Query Recommendation

Instead of querying data points one by one, the set-based objective function also inspires the choice of set S by retrieving a set of data points using a computer-suggested query. This can be seen as a principled method for query generation in the ReQ-ReC

framework (Chapters 4 and 5).

In text classification tasks, the data examples are documents, and the query/concept can be a term in information retrieval, or a statistical topic (multinomial distribution over terms). It is different from an "organic" document, but a generated/synthetic document. We assume that the user can still interpret the query and assign a classification label. Once labeled, the query can induce weak labels on a related set of documents S_θ. For instance, if θ is a statistical topic, then S_θ contains the top k_θ documents that are most likely to be generated by θ. The goal is to find the next query/concept such that once it is labeled, the classification model performance will improve by a large margin.

Let S_0 be the set of documents retrieved by previous queries collectively. According to Eq. (6.6), the new query θ should aim to maximize the objective:

$$\theta \leftarrow \operatorname*{argmax}_{\theta \in \Theta} \sum_{i \in S_0 \cup S_\theta} \ell(h(x_i), y_i^*) + \mu \Phi(S_0 \cup S_\theta) \tag{6.16}$$

Although there are concerns in active learning field that synthetic examples will likely be uninterpretable to human annotators [13], queries and statistical topics may still be amenable to human interpretation. When the query is closely related to a relatively large cluster (a subclass), it may as well be an interpretable topic. When the query is only close to a small amount of documents, we can directly present the closest document to the user and obtain labels.

When new modes of interaction is introduced in an algorithm, such as providing feedback on queries, automated evaluation can be difficult. To approximate the human judgment on synthetic queries, we can use the label(s) of its closest document(s) as a surrogate. To truly evaluate the effectiveness of this algorithm, we need to run real user studies to qualitatively evaluate the interpretability of generated queries, especially when the query becomes close to the decision boundary.

6.6 Conclusion

This chapter introduced a general framework for interactive machine learning algorithms. The framework starts by formulating the interactive process between the human teacher and the machine student as a two-player, zero-sum game. The payoff is the expected loss of the machine student on unseen examples, which is also the gain for the human teacher. In finite sample case, the expected loss can only be approximated by regularized empirical loss, the optimization objective for the machine. Inspired by this objective, we design the objective of a human teacher in a symmetric way: the human teacher aims to maximize regularized guess loss. The uncertainty sampling criteria corresponds to the guessed loss, while a high coverage (representativeness) of unlabeled data corresponds to the regularization term.

This objective turns out to be able to explain many existing interactive learning algorithms, including several classical active learning algorithms and the new ReQuery-ReClassification framework proposed in Chapter 3. This suggests that the framework is quite general and it has the potential to guide the design of new algorithms. As a sanity check, we enhanced the uncertainty sampling algorithm with a simple regularization term, which indeed outperformed the uncertainty sampling algorithm without regularization. This demonstrates that the framework could provide some meaningful guidance to algorithm design. Encouraged by this result, we then proceed to discuss a new text classification algorithm with a new kind of teaching modality: query recommendation and labeling, which connects the practice of machine learning and information retrieval. We believe this framework has the potential to inspire more flexible algorithms in the future.

CHAPTER 7

Summary and Outlook

This chapter summarizes the work and contribution presented in this dissertation, points out its limitations, and presents numerous directions for future work.

7.1 Summary

The successes of current machine learning methods crucially rely on a massive amount of labeled training data. For tasks like predicting ratings based on customer review text, training data may come for free. For tasks like recognizing human faces in images, training data can be crowdsourced at large scale. But for tasks like extracting medications in clinical notes, training data can only be provided by medical experts who may not have much spare time for data analysis and labeling. Without sufficient training data, current machine learning methods cannot hope to produce accurate models. In critical domains like health care, we have an increasing need for high-performance machine learning models to extract knowledge from data, yet domain expertise for creating training data remains scarce. To bridge the gap in these domains, we need new machine learning models that are less data-hungry. In particular, the new algorithm should be able to understand and make use of existing knowledge and be proactive and curious throughout the learning process. This dissertation studies interactive machine learning algorithms to achieve this goal.

Throughout the dissertation, with the exception of the last chapter, I designed and evaluated a spectrum of interactive machine learning algorithms.

The first is an algorithmic framework for solving high-recall information retrieval problems (Chapter 3). Commonly found in medical, legal, academic, and other professional domains, high-recall retrieval tasks aim to retrieve not just one, but *all* pieces of relevant information from an large collection, with reasonably high precision. Oftentimes all relevant documents cannot be retrieved by just one query, so a multi-query, iterative search process is needed. The results returned by each query may contain some relevant and inevitably some nonrelevant documents. An active learning classifier is employed to quickly distinguish relevant documents from nonrelevant ones. Thus we combine two iterative processes: active classification and iterative retrieval. This design allows a separation of concerns: the iterative retrieval loop can focus on recall by exploring document manifolds where relevant data may be located (high recall), while the active classification loop filters out any nonrelevant documents returned by the retrieval loop (high precision). When searching for the next queries, documents returned by all previous queries are not discarded, but accumulated into a pool for active classification. This process is named ReQuery-ReClassification, or ReQ-ReC for short. The framework saves significant human judgment efforts to achieve a high recall on multiple Text Retrieval Conference (TREC) collections.

Why this algorithm saves human effort? First, the re-classification loop uses active learning to save the total amount of labels needed to train a classifier. Second, the re-query step generates diverse queries to efficiently cover new areas in the document space, without discarding areas visited previously. Efficient exploration maximizes the discovery rate of new relevant documents, which in turn saves the total effort.

The ReQ-ReC is a versatile framework. From an information retrieval (IR) perspective, it provides an effective approach to high-recall retrieval, an important and hard problem in professional IR. From an active learning (AL) perspective, it enables

an AL algorithm to work in scenarios where the data collection is only accessible via a search interface, which is commonly the case for very large data collections to which a user does not have full access.

The second interactive machine learning algorithm tackles medical word sense disambiguation tasks, a type of text classification tasks. The human experts are invited to search for examples in each class to kick off the learning process (Chapter 4). The ReQ-ReC framework is adapted to fulfill this need: we just need to bring the human expert into the requery loop to compose queries, in addition to the reclassification loop to provide labels. We call this algorithm "ReQ-ReC expert". This allows the human experts to use their domain knowledge to guide the learning process in the beginning. Specifically, they are able to compose good queries to quickly search and provide representative examples for each class and kick off classifier training. Representative examples train a better classifier than randomly selected, probably non-representative examples. A more accurate classifier will enable the active learning algorithm to ask more sensible questions, which in turn helps collect more informative training examples to enhance the model. The "warm start" critically accelerates the whole learning process later on.

If expert's domain knowledge is the source of a warm start, then ReQ-ReC is using it in an indirect way. The knowledge, expressed in search keywords, is first used to retrieve documents, then affect the model learning by retrieved examples. What the machine learner sees is a set of labeled examples, not the search keywords; it has to infer the important keywords from the examples again. Why don't we let the learner directly learn from the search keywords in the first place? This reflection leads onto the design of the next algorithm.

The third interactive machine learning algorithm, which we call Informed Learning (Chapter 5), starts by learning directly from experts domain knowledge: keywords, or labeled features, for each class. The learner aims to respect this supervision signal by

constraining the prediction of any instance to match the desired label if that instance contains the keyword. This essentially creates an expansive set of weakly labeled examples to train the initial model, which gives rise to an even stronger initial model than trained by ReQ-ReC expert, which saves even more total efforts to train a high-performance machine learning model.

Throughout the experiments, a consistent observation is that a "warm start" at the very beginning, followed by active learning in the later stage, can reliably accelerate the overall learning process. The warm start can be obtained by leveraging prior knowledge, either through expert search (as in ReQ-ReC expert) or labeled features (as in Informed Learning). Then we enter a virtuous cycle of interactive learning: an accurate initial model will strategically ask good questions using active learning, which in turn collects informative training examples to train an even more accurate model. The opposite of this is the "cold start" vicious cycle: the random sparse training examples will train a poor model, and based on the prediction of the poor model, active learning questions are less useful in improving the model. In specialized domains such as medicine and law, there often exists a wealth of domain knowledge about how to perform certain tasks, either in the form of knowledge base entries or rules of thumbs as domain experts' experience. We should do our best to leverage such knowledge to provide a warm start to an interactive machine learning process.

The three interactive machine learning algorithms can be aligned on a spectrum. At one end, we have ad hoc information retrieval rankers, or we can interpret them as a weak classifiers trained only with one training example – the search query. At the other end, we have supervised learning model trained with a large set of labeled examples. At the information retrieval end, a user accomplishes a search tasks by composing search queries and interacting with the retrieval system. It is an easy and intuitive mode of interaction, but a search engine cannot help with more complex data analysis tasks. At the supervised learning end, the human user is only responsible for labeling data

and do not need to interact with an information system. Labeling data takes arduous work, but the resulting model can perform complex data analysis tasks with high performance. High-recall information retrieval and interactive classifier training are in the middle of this spectrum. They try to combine the best of both worlds: the human user can interact with the algorithm to easily get an initial good classifier, and then improve the classifier by providing feedback for only a subset of selected examples. In this perspective, future development of interactive machine learning should draw inspiration from the two well-established camps: interactive information retrieval and interactive (and interpretable) machine learning.

The last chapter looks back at the myriad of interactive learning algorithms and asks the question: *what is the common principle underlying these algorithms*? A close analogy of such a principle is the regularized loss minimization principle for supervised learning algorithms. This principle provides guidance to alleviate the overfitting problem: a balance between model complexity and its fitting of training data. It unifies the objective of many algorithms like linear regression, lasso regression, logistic regression, support vector machines, etc, and guides the design of new objectives. In general, an underlying principle helps us gain fundamental understanding of existing algorithms and guide the design of new ones.

As a key contribution of this dissertation, I propose a general framework for interactive machine learning algorithms. Such a principle is much needed in new algorithm design, while it is lacking in current literature. In our formulation, the machine learner and the human teacher are engaged in a repeated game. The machine learner aims to find a model to minimize the regularized empirical loss, while the human teacher aims to find a subset of training data to maximize the *regularized guessed loss*. It turns out that the regularized guessed loss is a unified objective function. It explains many existing interactive machine learning algorithms, including uncertainty sampling, density-weighted sampling, batch-mode active learning, expected error reduction, and

the newly proposed ReQ-ReC algorithm. As a sanity check, we enhanced the uncertainty sampling algorithm with a simple regularization term, which is indeed more robust than the uncertainty sampling algorithm without regularization. This demonstrates that the unified objective can indeed provide meaningful guidance to algorithm design.

7.2 Limitations

Below I discuss several limitations of this dissertation and the ways to improve them.

The first limitation is that all interactive machine learning algorithms are evaluated with simulated human inputs, which are derived from labeled corpora. On the one hand, using simulated input allows us to have stable and scalable comparison of many learning algorithms on numerous tasks. On the other hand, the observed learning behaviors are only approximations of that would happen on real users. An immediate next step is to design and implement user studies to evaluate these interactive machine learning algorithms in practice.

A second limitation is that the proposed algorithms all assume that the human teacher will provide accurate labels. While this may be true for well-trained domain experts on specific tasks, the assumption can fail because (1) the annotators may have different levels of expertise, (2) examples are intrinsically hard, or different annotators may have different answers; (3) human annotators make mistakes and experience fatigue over time [4]. These problems will surface when annotation tasks are crowdsourced to a group of people with varying expertise [82]. In such cases, an interactive machine learning algorithm should account for quality of contributions from different annotators when querying for labels [22]. In general, a better understanding of the user's needs, expectations, and interaction behaviors will greatly benefit the design of interactive learning systems.

When the interaction including labeling features, the human teacher may not be able to assign a clear label, especially when such features are ambiguous and do not strongly indicate a particular class. To account for this issue, Chapter 5 implemented feature labeling oracles with varying quality. When designing real world interfaces, one can allow the human teacher to answer "I don't know", or provide both a label and a low confidence score, suggesting that the label may not be useful.

A third limitation is that we assume the same cost for different annotation actions: labeling an instance, labeling a feature, highlighting an indicative feature in an instance, and composing a search query. These actions take different efforts in practice; the same action may have varying costs on different data objects [121]. For instance, labeling a feature generally takes shorter time than labeling an instance, but may take longer time if the feature is hard to interpret as it is out of context. Such a limitation can be addressed by using different estimated costs per action, or measuring the time in real world experiments and user studies.

7.3 Future Directions

The general framework of interactive machine learning, as well as lessons learned in the design, evaluation, and application of specific learning algorithms, suggests many research directions for further exploration.

Rich interaction channels for interactive machine learning

This dissertation developed algorithms that allow domain experts to provide supervision signals via search queries, keywords, rationales, in addition to labels. Through empirical experiments, we demonstrate that these interaction channels allow high "throughput" of expert's domain knowledge than providing labels. A future research direction is to explore richer interaction channels/modalities (1) for human users to

teach the machine, and (2) for the machine to talk back to human users. We expect to see faster learning rates with these new channels. Note that the two directions may or may not share the same form, e.g. to provide interpretable rules is easy for humans but hard for machines, while to visualize data is easy for machines but hard for humans.

Along with these new channels is the evaluation problem: which channels are most effective at which learning stage? To evaluate these channels, future work can take use simulated human inputs as a starting point (the Cranfield paradigm experiments), but ultimately these evaluations should take place in real-world settings with real users.

Computer-assisted content analysis

Content analysis is a general research method for making sense of recorded communication. It is widely used by HCI researchers and health care practitioners to understand user-generated content in online social media, most of which consists of text. By reading text, researchers conceptualize the content in a set of codes, assign codes to individual documents, and analyze the distribution, correlation, and evolution of these codes. To gain insights into the ever-growing user-generated content, researchers need more powerful content analysis tools.

It will be very useful to have interactive systems to support large-scale content analysis. The system aims to help researchers quickly develop their codebook, efficiently code documents, ensure the generalizability of codes, and keep track of statistics such as inter-rater reliability and code distribution. The system will summarize the text collection as clusters/hierarchies of documents and words to assist open coding and axial coding. To train the text classifier, researchers can label informative words or use them to search for representative documents for each code category. The system will also suggest informative words and documents for further labeling. The classifier is updated with new labels, and evaluated on a validation set. The validation set needs to be gradually expanded to avoid overfitting, and revised if new codes are discovered

in the iterative learning process. Such a project is best carried out as a collaboration between HCI and data mining communities.

Heterogeneous health information analysis

Recent years have seen unprecedented growth of health-related text information. These text information comes in different genres: standardized knowledge bases such as the UMLS; biomedical literature such as MEDLINE; various types of clinical notes such as discharge summaries and radiologists' notes; and online health communities, such as MedHelp and WebMD. New knowledge and patterns often emerge when multiple sources of data are mined together. For example, by jointly mining biomedical literature and clinical notes, novel clinical findings could be discovered; jointly analyzing discussions in online health forums and records in hospitals may reveal different values and concerns of health care consumers and providers. A key challenge in the joint analysis of heterogeneous data is to represent different genres of information in a common space that facilitates data analysis. To achieve these goals, one develop techniques for learning unified text representation across different health data genres, leveraging existing medical knowledge bases and keeping the manual effort of domain experts at a minimum. With the learned data representation, we can explore many interesting heterogeneous data mining problems.

Interactive data visualization and annotation

Data scientists will gain enormous help if she or he can "see" large data sets from different perspectives and make informed decisions on where to explore and annotate next. In this research direction, we can project an unlabeled data set onto a 2D canvas using nonlinear dimensionality reduction methods. With different colored layers, we can visualize data density, true and predicted labels, uncertain regions, and promising areas for future annotation. These layers will change as a result of interactive learning.

Thus data annotation can be viewed as a "board game" between the human and the machine, leading to further research opportunities in HCI and deep reinforcement learning.

Learning to Teach

In the Big Data era, it is critical to optimally allocate resources (human attention, annotation, and knowledge) in teaching machine learning algorithms to achieve the best possible outcome. In classical active learning, the teacher obtains the next labeled example based on heuristic measures such as classifier uncertainty or data density, or lookahead search such as expected error reduction. We can unify these approaches by *learning the optimal teaching strategy* with reinforcement learning from existing labeled data sets. The goal is to learn an adaptive teaching policy that generalizes well to unseen supervised learning tasks. It will inform human teachers when to label representative examples, when to move towards boundary cases, and when to examine outliers. The action space can be further expanded, such as labeling words/phrases and synthetic examples. We can then learn more efficient ways of teaching machine learning algorithms, in addition to labeling examples.

Note that this is different from the machine teaching literature, where fully labeled data is needed to compute the training curriculum [164].

To summarize, interactive machine learning is a promising new paradigm towards data-efficient, user-friendly, and overall more intelligent machine learning methodology and applications.

APPENDIX A

Proofs of Theorems

Proof of Theorem 6.2.1

For any fixed distribution selected by G, the optimal strategy of D is to choose the Bayes classifier h^*.

Proof. For any h,

$$R(G_{XY}, h) = \mathbb{E}_{(x,y)\sim G_{XY}}\left[\mathbf{1}_{\{h(x)\neq y\}}\right] \tag{A.1}$$

$$= \mathbb{E}_{x\sim g(x)}\left[\mathbb{E}_{y\sim \eta(x)}\left[\mathbf{1}_{\{h(x)\neq y\}}\right]\right] \tag{A.2}$$

$$= \mathbb{E}_{x\sim g(x)}\left[\eta(x)\mathbf{1}_{\{h(x)=0\}} + (1-\eta(x))\mathbf{1}_{\{h(x)=1\}}\right] \tag{A.3}$$

To minimize $R(G_{XY}, h)$ for a fixed G_{XY}, it suffices for $h(x)$ to be such that $\forall x$,

$$\eta(x)\mathbf{1}_{\{h(x)=0\}} + (1-\eta(x))\mathbf{1}_{\{h(x)=1\}} \tag{A.4}$$

is minimized. Note that the indicators here are mutually exclusive, so it suffices for D to choose the Bayes classifier $h^*(x)$ defined in Eq. (6.3), concluding the proof. □

The above theorem holds because Bayes classifier h^* is agnostic to the X-marginal

distribution. The minimax game in Eq. (6.4) can then be reformulated as:

$$\max_G L(G) = \max_G R(G_{XY}, h^*) \tag{A.5}$$

$$= \max_g \mathbb{E}_{x\sim g(x)}\left[r(x)\right] \tag{A.6}$$

where we define the risk function $r(x) = \min(\eta(x), 1-\eta(x))$. The maximum value of $L(G)$ is $1/2$:

$$\max_G L(G) = \max_g \mathbb{E}_{x\sim g(x)}\left[r(x)\right] \leq \max_x r(x) = \frac{1}{2}\ . \tag{A.7}$$

The equality holds if and only if $\eta(x) = 1-\eta(x)$, i.e. $\eta(x) = 1/2$, assuming $\eta(x)$ is continuous. The next theorem shows that G can reach this maximum arbitrarily closely.

Proof of Theorem 6.2.2

For any $\epsilon > 0$, the generator G's can select a distribution g that achieves $R(G_{XY}, h^*) = L(G) = 1/2 - \epsilon$.

Proof. Let us partition the feature space $\mathcal{X}$ into two disjoint regions: $A = \{x : \eta(x) \neq 1/2\}$ and $B = \{x : \eta(x) = 1/2\}$. Let $g(x)$ assign δ probability mass on A and $1-\delta$ probability mass on B.

$$L(G) = \mathbb{E}_{x\sim g(x)}\left[r(x)\right] = \int_{x\in\mathcal{X}} g(x)r(x)dx \tag{A.8}$$

$$= \int_{x\in A} g(x)r(x)dx + \int_{x\in B} g(x)\min\left(\frac{1}{2}, 1-\frac{1}{2}\right)dx \tag{A.9}$$

$$= \left(\frac{1}{2}-\nu\right)\int_{x\in A} g(x)dx + \frac{1}{2}\int_{x\in B} g(x)dx \tag{A.10}$$

$$= \frac{1}{2} - \delta\nu\ . \tag{A.11}$$

In (A.10), $1/2-\nu = \mathbb{E}_{x\sim g(x)|A}\left[r(x)\right]$. As $\eta(x) \neq 1/2$ on A, $r(x) = \min\left(\eta(x), 1-\eta(x)\right) < 1/2$, therefore $\nu > 0$. G can select $g(x)$ with $\delta = \epsilon/\nu$ probability mass on A to achieve $L(G) = 1/2 - \epsilon$. $\square$

BIBLIOGRAPHY

[1] Apache lucene project. `http://lucene.apache.org/`.

[2] Icml 2016 workshop on data-efficient machine learning. `https://sites.google.com/site/dataefficientml/`.

[3] Charu C Aggarwal, Xiangnan Kong, Quanquan Gu, Jiawei Han, and Philip S Yu. Active learning: A survey. 2014.

[4] Saleema Amershi, Maya Cakmak, W Bradley Knox, and Todd Kulesza. Power to the people: The role of humans in interactive machine learning. American Association for Artificial Intelligence, 2014.

[5] Saleema Amershi, Max Chickering, Steven M Drucker, Bongshin Lee, Patrice Simard, and Jina Suh. Modeltracker: Redesigning performance analysis tools for machine learning. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 337–346. ACM, 2015.

[6] Saleema Amershi, James Fogarty, and Daniel Weld. Regroup: Interactive machine learning for on-demand group creation in social networks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 21–30. ACM, 2012.

[7] Josh Attenberg and Seyda Ertekin. Class imbalance and active learning. *H. He, & Y. Ma, Imbalanced Learning: Foundations, Algorithms, and Applications*, pages 101–149, 2013.

[8] Josh Attenberg, Panagiotis G Ipeirotis, and Foster J Provost. Beat the machine: Challenging workers to find the unknown unknowns. *Human Computation*, 11(11):2–7, 2011.

[9] Josh Attenberg, Prem Melville, and Foster Provost. A unified approach to active dual supervision for labeling features and examples. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 40–55. Springer, 2010.

[10] Josh Attenberg and Foster Provost. Why label when you can search?: alternatives to active learning for applying human resources to build classification models

under extreme class imbalance. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 423–432. ACM, 2010.

[11] Josh Attenberg and Foster Provost. Inactive learning?: difficulties employing active learning in practice. *ACM SIGKDD Explorations Newsletter*, 12(2):36–41, 2011.

[12] Jason Baldridge and Miles Osborne. Active learning and the total cost of annotation. In *EMNLP*, pages 9–16, 2004.

[13] Eric B Baum and Kenneth Lang. Query learning can work poorly when a human oracle is used. In *International joint conference on neural networks*, volume 8, page 8, 1992.

[14] Alina Beygelzimer, Daniel J Hsu, John Langford, and Chicheng Zhang. Search improves label for active learning. In *Advances in Neural Information Processing Systems*, pages 3342–3350, 2016.

[15] Mustafa Bilgic. Career: Active learning through rich and transparent interactions. `http://grantome.com/grant/NSF/IIS-1350337`. Accessed: 2016-04-10.

[16] Mustafa Bilgic and Paul N Bennett. Active query selection for learning rankers. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 1033–1034. ACM, 2012.

[17] Michael Brooks, Saleema Amershi, Bongshin Lee, Steven M Drucker, Ashish Kapoor, and Patrice Simard. Featureinsight: Visual support for error-driven feature ideation in text classification. In *Visual Analytics Science and Technology (VAST), 2015 IEEE Conference on*, pages 105–112. IEEE, 2015.

[18] Chris Buckley, Amit Singhal, Mandar Mitra, and Gerard Salton. New retrieval approaches using smart: Trec 4. In *Proceedings of the Fourth Text REtrieval Conference (TREC-4)*, pages 25–48, 1995.

[19] Chris Callison-Burch and Mark Dredze. Creating speech and language data with amazon's mechanical turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 1–12. Association for Computational Linguistics, 2010.

[20] Rui M Castro and Robert D Nowak. Minimax bounds for active learning. *IEEE Transactions on Information Theory*, 54(5):2339–2353, 2008.

[21] Shuo Chang, Peng Dai, Lichan Hong, Cheng Sheng, Tianjiao Zhang, and Ed H Chi. Appgrouper: Knowledge-based interactive clustering tool for app search results. In *Proceedings of the 21st International Conference on Intelligent User Interfaces*, pages 348–358. ACM, 2016.

[22] Xi Chen, Paul N Bennett, Kevyn Collins-Thompson, and Eric Horvitz. Pairwise ranking aggregation in a crowdsourced setting. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 193–202. ACM, 2013.

[23] Yukun Chen, Hongxin Cao, Qiaozhu Mei, Kai Zheng, and Hua Xu. Applying active learning to supervised word sense disambiguation in medline. *Journal of the American Medical Informatics Association*, 20(5):1001–1006, 2013.

[24] Zhe Chen, Sasha Dadiomov, Richard Wesley, Gang Xiao, Daniel Cory, Michael Cafarella, and Jock Mackinlay. Spreadsheet property detection with rule-assisted active learning. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 999–1008. ACM, 2017.

[25] Zhiyuan Chen and Bing Liu. *Lifelong Machine Learning*. Morgan & Claypool Publishers, 2016.

[26] Jason Chuang, Daniel Ramage, Christopher Manning, and Jeffrey Heer. Interpretation and trust: Designing model-driven visualizations for text analysis. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 443–452. ACM, 2012.

[27] Gordon V Cormack and Maura R Grossman. Scalability of continuous active learning for reliable high-recall text classification. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 1039–1048. ACM, 2016.

[28] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[29] Ingemar J Cox, Matt L Miller, Stephen M Omohundro, and Peter N Yianilos. Pichunter: Bayesian relevance feedback for image retrieval. In *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, volume 3, pages 361–369. IEEE, 1996.

[30] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[31] Jia Deng, Olga Russakovsky, Jonathan Krause, Michael S Bernstein, Alex Berg, and Li Fei-Fei. Scalable multi-label annotation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 3099–3102. ACM, 2014.

[32] Pinar Donmez and Jaime G Carbonell. Proactive learning: cost-sensitive active learning with multiple imperfect oracles. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 619–628. ACM, 2008.

[33] Pinar Donmez, Jaime G. Carbonell, and Paul N. Bennett. Dual strategy active learning. In *Proceedings of the 18th European Conference on Machine Learning*, ECML '07, pages 116–127. Springer-Verlag, 2007.

[34] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

[35] Gregory Druck, Gideon Mann, and Andrew McCallum. Learning from labeled features using generalized expectation criteria. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 595–602. ACM, 2008.

[36] Gregory Druck, Burr Settles, and Andrew McCallum. Active learning by labeling features. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 81–90. Association for Computational Linguistics, 2009.

[37] Harris Drucker, Behzad Shahrary, and David C Gibbon. Support vector machines: relevance feedback and information retrieval. *Information Processing and Management*, 38(3):305–323, 2002.

[38] Alex Endert, Patrick Fiaux, and Chris North. Semantic interaction for visual text analytics. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 473–482. ACM, 2012.

[39] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[40] Hui Fang and ChengXiang Zhai. Semantic term matching in axiomatic approaches to information retrieval. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 115–122. ACM, 2006.

[41] Li Fei-Fei, Rob Fergus, and Pietro Perona. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence*, 28(4):594–611, 2006.

[42] Gregory P Finley, Serguei VS Pakhomov, Reed McEwan, and Genevieve B Melton. Towards comprehensive clinical abbreviation disambiguation using machine-labeled training data. In *AMIA Annual Symposium Proceedings*, volume 2016, page 560. American Medical Informatics Association, 2016.

[43] Carol Friedman, Lyudmila Shagina, Yves Lussier, and George Hripcsak. Automated encoding of clinical documents based on natural language processing. *Journal of the American Medical Informatics Association*, 11(5):392–402, 2004.

[44] Yifan Fu, Bin Li, Xingquan Zhu, and Chengqi Zhang. Do they belong to the same class: active learning by querying pairwise label homogeneity. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 2161–2164. ACM, 2011.

[45] Ravi Ganti and Alexander G Gray. Building bridges: viewing active learning from the multi-armed bandit lens. *arXiv preprint arXiv:1309.6830*, 2013.

[46] Marco Gillies, Andrea Kleinsmith, and Harry Brenton. Applying the cassm framework to improving end user debugging of interactive machine learning. In *Proceedings of the 20th International Conference on Intelligent User Interfaces*, pages 181–185. ACM, 2015.

[47] Daniel Golovin and Andreas Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, 42:427–486, 2011.

[48] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[49] Sai R. Gouravajhala, Jinyeong Yim, Karthik Desingh, Yanda Huang, Odest Chadwicke Jenkins, and Walter S. Lasecki. EURECA: enhanced understanding of real environments via crowd assistance. In *Proceedings of the Sixth AAAI Conference on Human Computation and Crowdsourcing, HCOMP 2018, Zürich, Switzerland, July 5-8, 2018.*, pages 31–40, 2018.

[50] Mark S Granovetter. The strength of weak ties. *American journal of sociology*, pages 1360–1380, 1973.

[51] Isabelle Guyon, Gavin Cawley, Gideon Dror, and Vincent Lemaire. Results of the active learning challenge. In *Active Learning and Experimental Design workshop In conjunction with AISTATS 2010*, pages 19–45, 2011.

[52] David A Hanauer. Emerse: the electronic medical record search engine. In *AMIA Annual Symposium Proceedings*, volume 2006, page 941. American Medical Informatics Association, 2006.

[53] David A Hanauer, Qiaozhu Mei, James Law, Ritu Khanna, and Kai Zheng. Supporting information retrieval from electronic health records: A report of university of michigans nine-year experience in developing and using the electronic medical record search engine (emerse). *Journal of biomedical informatics*, 55:290–300, 2015.

[54] Steve Hanneke and Liu Yang. Minimax analysis of active learning. *Journal of Machine Learning Research*, 16(12):3487–3602, 2015.

[55] Donna Harman. Relevance feedback revisited. In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '92, pages 1–10, New York, NY, USA, 1992. ACM.

[56] Abhay S Harpale and Yiming Yang. Personalized active learning for collaborative filtering. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 91–98. ACM, 2008.

[57] Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9):1263–1284, 2009.

[58] Yulan He. Learning sentiment classification model from labeled features. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 1685–1688. ACM, 2010.

[59] Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 541–550. Association for Computational Linguistics, 2011.

[60] Steven CH Hoi, Rong Jin, Jianke Zhu, and Michael R Lyu. Batch mode active learning and its application to medical image classification. In *Proceedings of the 23rd international conference on Machine learning*, pages 417–424. ACM, 2006.

[61] Steven CH Hoi, Rong Jin, Jianke Zhu, and Michael R Lyu. Semisupervised svm batch mode active learning with applications to image retrieval. *ACM Transactions on Information Systems (TOIS)*, 27(3):16, 2009.

[62] Eric Horvitz. Principles of mixed-initiative user interfaces. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 159–166. ACM, 1999.

[63] Neil Houlsby, José Miguel Hernández-Lobato, and Zoubin Ghahramani. Cold-start active learning with robust ordinal matrix factorization. In *International Conference on Machine Learning*, pages 766–774, 2014.

[64] Nancy Ide and Jean Véronis. Introduction to the special issue on word sense disambiguation: the state of the art. *Computational linguistics*, 24(1):2–40, 1998.

[65] Jagadeesh Jagarlamudi, Hal Daumé III, and Raghavendra Udupa. Incorporating lexical priors into topic models. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 204–213. Association for Computational Linguistics, 2012.

[66] Kevin G Jamieson and Robert Nowak. Active ranking using pairwise comparisons. In *Advances in Neural Information Processing Systems*, pages 2240–2248, 2011.

[67] Shali Jiang, Gustavo Malkomes, Geoff Converse, Alyssa Shofner, Benjamin Moseley, and Roman Garnett. Efficient nonmyopic active search. In *International Conference on Machine Learning*, pages 1714–1723, 2017.

[68] Antonio J Jimeno-Yepes, Bridget T McInnes, and Alan R Aronson. Exploiting mesh indexing in medline to generate a data set for word sense disambiguation. *BMC bioinformatics*, 12(1):223, 2011.

[69] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 154–161. ACM, 2005.

[70] Ashish Kapoor, Bongshin Lee, Desney Tan, and Eric Horvitz. Interactive optimization for steering machine classification. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1343–1352. ACM, 2010.

[71] Xiangnan Kong, Wei Fan, and Philip S Yu. Dual active feature and sample selection for graph classification. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 654–662. ACM, 2011.

[72] Andreas Krause and Daniel Golovin. Submodular function maximization. *Tractability: Practical Approaches to Hard Problems*, 3(19):8, 2012.

[73] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123(1):32–73, 2017.

[74] Robert Krovetz. Viewing morphology as an inference process. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 191–202. ACM, 1993.

[75] Alex Kulesza, Ben Taskar, et al. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2–3):123–286, 2012.

[76] Todd Kulesza, Saleema Amershi, Rich Caruana, Danyel Fisher, and Denis Charles. Structured labeling to facilitate concept evolution in machine learning. In *Proceedings of CHI*, 2014.

[77] Todd Kulesza, Margaret Burnett, Weng-Keen Wong, and Simone Stumpf. Principles of explanatory debugging to personalize interactive machine learning. In *Proceedings of the 20th International Conference on Intelligent User Interfaces*, pages 126–137. ACM, 2015.

[78] Todd Kulesza, Simone Stumpf, Margaret Burnett, and Irwin Kwan. Tell me more?: the effects of mental model soundness on personalizing an intelligent

agent. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1–10. ACM, 2012.

[79] Himabindu Lakkaraju, Ece Kamar, Rich Caruana, and Eric Horvitz. Identifying unknown unknowns in the open world: Representations and policies for guided exploration. In *AAAI*, volume 1, page 2, 2017.

[80] Ken Lang. Newsweeder: Learning to filter netnews. In *In Proceedings of the Twelfth International Conference on Machine Learning*, pages 331–339, 1995.

[81] Walter S Lasecki, Christopher D Miller, Iftekhar Naim, Raja Kushalnagar, Adam Sadilek, Daniel Gildea, and Jeffrey P Bigham. Scribe: deep integration of human and machine intelligence to caption speech in real time. *Communications of the ACM*, 60(9):93–100, 2017.

[82] Matthew Lease. On quality control and machine learning in crowdsourcing. *Human Computation*, 11(11), 2011.

[83] Gondy Leroy and Thomas C Rindflesch. Using symbolic knowledge in the umls to disambiguate words in small datasets with a naïve bayes classifier. In *Medinfo*, pages 381–385, 2004.

[84] Cheng Li, Yue Wang, and Qiaozhu Mei. A user-in-the-loop process for investigational search: Foreseer in trec 2013 microblog track. In *TREC*, 2013.

[85] Cheng Li, Yue Wang, Paul Resnick, and Qiaozhu Mei. Req-rec: High recall retrieval with query pooling and interactive classification. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 163–172. ACM, 2014.

[86] Greg Little, Lydia B Chilton, Max Goldman, and Robert C Miller. Turkit: human computation algorithms on mechanical turk. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, pages 57–66. ACM, 2010.

[87] Bing Liu, Xiaoli Li, Wee Sun Lee, and Philip S Yu. Text classification by labeling words. In *AAAI*, volume 4, pages 425–430, 2004.

[88] Hongfang Liu, Stephen B Johnson, and Carol Friedman. Automatic resolution of ambiguous terms based on machine learning and conceptual relations in the umls. *Journal of the American Medical Informatics Association*, 9(6):621–636, 2002.

[89] Hongfang Liu, Yves A Lussier, and Carol Friedman. Disambiguating ambiguous biomedical terms in biomedical narrative text: an unsupervised method. *Journal of biomedical informatics*, 34(4):249–261, 2001.

[90] Hongfang Liu, Virginia Teller, and Carol Friedman. A multi-aspect comparison study of supervised word sense disambiguation. *Journal of the American Medical Informatics Association*, 11(4):320–331, 2004.

[91] Weiyang Liu, Bo Dai, Ahmad Humayun, Charlene Tay, Chen Yu, Linda B Smith, James M Rehg, and Le Song. Iterative machine teaching. *arXiv preprint arXiv:1705.10470*, 2017.

[92] Bo Long, Jiang Bian, Olivier Chapelle, Ya Zhang, Yoshiyuki Inagaki, and Yi Chang. Active learning for ranking through expected loss optimization. *IEEE Transactions on Knowledge and Data Engineering*, 27(5):1180–1191, 2015.

[93] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to information retrieval*, volume 1. Cambridge University Press Cambridge, 2008.

[94] Prem Melville and Vikas Sindhwani. Active dual supervision: Reducing the cost of annotating examples and features. In *Proceedings of the NAACL HLT 2009 Workshop on Active Learning for Natural Language Processing*, pages 49–57. Association for Computational Linguistics, 2009.

[95] Sungrim Moon, Serguei Pakhomov, Nathan Liu, James O Ryan, and Genevieve B Melton. A sense inventory for clinical abbreviations and acronyms created using clinical notes and medical dictionary resources. *Journal of the American Medical Informatics Association*, 21(2):299–307, 2013.

[96] Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V Vazirani. *Algorithmic game theory*, volume 1. Cambridge University Press Cambridge, 2007.

[97] Douglas W Oard and William Webber. *Information Retrieval for E-Discovery*. Foundations and Trends in Information Retrieval. Now Publishers, 2013.

[98] Serguei Pakhomov, Ted Pedersen, and Christopher G Chute. Abbreviation and acronym disambiguation in clinical discourse. In *AMIA Annual Symposium Proceedings*, volume 2005, page 589. American Medical Informatics Association, 2005.

[99] Sinno Jialin Pan, Qiang Yang, et al. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

[100] Adam Perer and Fei Wang. Frequence: interactive mining and visualization of temporal frequent event sequences. In *Proceedings of the 19th international conference on Intelligent User Interfaces*, pages 153–162. ACM, 2014.

[101] Yashoteja Prabhu, Anil Kag, Shilpa Gopinath, Kunal Dahiya, Shrutendra Harsola, Rahul Agrawal, and Manik Varma. Extreme multi-label learning with label features for warm-start tagging, ranking & recommendation. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 441–449. ACM, 2018.

[102] Jay Pujara, Ben London, and Lise Getoor. Reducing label cost by combining feature labels and crowdsourcing. In *ICML workshop on Combining learning strategies to reduce label cost*, 2011.

[103] James Pustejovsky and Amber Stubbs. *Natural Language Annotation for Machine Learning: A guide to corpus-building for applications. "* O'Reilly Media, Inc.", 2012.

[104] Hema Raghavan and James Allan. An interactive algorithm for asking and incorporating feature feedback into support vector machines. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 79–86. ACM, 2007.

[105] Hema Raghavan, Omid Madani, and Rosie Jones. Interactive feature selection. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, IJCAI'05, pages 841–846, 2005.

[106] Hema Raghavan, Omid Madani, and Rosie Jones. Active learning with feedback on features and instances. *The Journal of Machine Learning Research*, 7:1655–1686, 2006.

[107] Suju Rajan, Dragomir Yankov, Scott J Gaffney, and Adwait Ratnaparkhi. A large-scale active learning system for topical categorization on the web. In *Proceedings of the 19th international conference on World wide web*, pages 791–800. ACM, 2010.

[108] Karthik Raman, Paul N. Bennett, and Kevyn Collins-Thompson. Toward whole-session relevance: Exploring intrinsic diversity in web search. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '13, pages 463–472. ACM, 2013.

[109] Parisa Rashidi and Diane J Cook. Ask me better questions: active learning queries based on rule induction. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 904–912. ACM, 2011.

[110] J.J. Rocchio. Relevance feedback in information retrieval. In *The SMART retrieval system experiments in automatic document processing*, pages 313–323. Prentice Hall, 1971.

[111] Nicholas Roy and Andrew McCallum. Toward optimal active learning through monte carlo estimation of error reduction. *ICML, Williamstown*, pages 441–448, 2001.

[112] Yong Rui, Thomas S Huang, Michael Ortega, and Sharad Mehrotra. Relevance feedback: a power tool for interactive content-based image retrieval. *Circ. and Sys. for Video Tech., IEEE Transactions on*, 8(5):644–655, 1998.

[113] Olga Russakovsky, Li-Jia Li, and Li Fei-Fei. Best of both worlds: human-machine collaboration for object annotation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2121–2131, 2015.

[114] Gerard Salton and Chris Buckley. Improving retrieval performance by relevance feedback. In *Journal of the American Society for Information Science (1986-1998)*, volume 41, pages 288–297. ACM, 1990.

[115] Guergana K Savova, Anni R Coden, Igor L Sominsky, Rie Johnson, Philip V Ogren, Piet C De Groen, and Christopher G Chute. Word sense disambiguation across two domains: biomedical literature and clinical notes. *Journal of biomedical informatics*, 41(6):1088–1100, 2008.

[116] Martijn J Schuemie, Jan A Kors, and Barend Mons. Word sense disambiguation in the biomedical domain: an overview. *Journal of Computational Biology*, 12(5):554–565, 2005.

[117] Burr Settles. Active learning literature survey. *University of Wisconsin, Madison*, 2010.

[118] Burr Settles. Closing the loop: Fast, interactive semi-supervised annotation with queries on features and instances. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1467–1478. Association for Computational Linguistics, 2011.

[119] Burr Settles. From theories to queries: Active learning in practice. *Active Learning and Experimental Design W*, pages 1–18, 2011.

[120] Burr Settles and Mark Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the conference on empirical methods in natural language processing*, pages 1070–1079. Association for Computational Linguistics, 2008.

[121] Burr Settles, Mark Craven, and Lewis Friedland. Active learning with real annotation costs. In *Proceedings of the NIPS workshop on cost-sensitive learning*, pages 1–10, 2008.

[122] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016.

[123] Manali Sharma, Di Zhuang, and Mustafa Bilgic. Active learning with rationales for text classification. In *Proceedings of the NAACL HLT 2015 Conference (Long Posters)*, pages 49–57. Association for Computational Linguistics, 2015.

[124] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

[125] Xuehua Shen, Bin Tan, and ChengXiang Zhai. Context-sensitive information retrieval using implicit feedback. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '05, pages 43–50. ACM, 2005.

[126] Xuehua Shen and ChengXiang Zhai. Active feedback in ad hoc information retrieval. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 59–66. ACM, 2005.

[127] Victor S Sheng, Foster Provost, and Panagiotis G Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 614–622. ACM, 2008.

[128] Vikas Sindhwani, Prem Melville, and Richard D Lawrence. Uncertainty sampling and transductive experimental design for active dual supervision. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 953–960. ACM, 2009.

[129] Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y Ng. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of the conference on empirical methods in natural language processing*, pages 254–263. Association for Computational Linguistics, 2008.

[130] I Soboroff, I Ounis, C Macdonald, and J Lin. Overview of the trec-2012 microblog track. In *Proceedings of the Twenty-First Text REtrieval Conference*, 2012.

[131] Richard Socher, Milind Ganjoo, Christopher D Manning, and Andrew Ng. Zero-shot learning through cross-modal transfer. In *Advances in neural information processing systems*, pages 935–943, 2013.

[132] Amber Stubbs and Özlem Uzuner. Annotating longitudinal clinical narratives for de-identification: The 2014 i2b2/uthealth corpus. *Journal of biomedical informatics*, 58:S20–S29, 2015.

[133] Jina Suh, Xiaojin Zhu, and Saleema Amershi. The label complexity of mixed-initiative classifier training. In *International Conference on Machine Learning*, pages 2800–2809, 2016.

[134] Buzhou Tang, Hongxin Cao, Yonghui Wu, Min Jiang, and Hua Xu. Clinical entity recognition using structural support vector machines with rich features. In *Proceedings of the ACM sixth international workshop on Data and text mining in biomedical informatics*, pages 13–20. ACM, 2012.

[135] Aibo Tian and Matthew Lease. Active learning to maximize accuracy vs. effort in interactive information retrieval. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 145–154. ACM, 2011.

[136] Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *J. Mach. Learn. Res.*, 2:45–66, March 2002.

[137] Özlem Uzuner, Brett R South, Shuying Shen, and Scott L DuVall. 2010 i2b2/va challenge on concepts, assertions, and relations in clinical text. *Journal of the American Medical Informatics Association*, 18(5):552–556, 2011.

[138] Vladimir Naumovich Vapnik. An overview of statistical learning theory. *IEEE transactions on neural networks*, 10(5):988–999, 1999.

[139] Ellen M Voorhees. Query expansion using lexical-semantic relations. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 61–69. ACM, 1994.

[140] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. Irgan: A minimax game for unifying generative and discriminative information retrieval models. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 515–524. ACM, 2017.

[141] Xuanhui Wang, Hui Fang, and ChengXiang Zhai. A study of methods for negative relevance feedback. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 219–226. ACM, 2008.

[142] Xuezhi Wang, Roman Garnett, and Jeff Schneider. Active search on graphs. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 731–738. ACM, 2013.

[143] Yue Wang, Kai Zheng, Hua Xu, and Qiaozhu Mei. Clinical word sense disambiguation with interactive search and classification. In *AMIA Annual Symposium Proceedings*, volume 2016, page 2062. American Medical Informatics Association, 2016.

[144] Kai Wei, Rishabh Iyer, and Jeff Bilmes. Submodularity in data subset selection and active learning. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 1954–1963, 2015.

[145] Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics bulletin*, 1(6):80–83, 1945.

[146] Ian H Witten, Craig G Nevill-Manning, and David Maulsby. Interacting with learning agents: implications for ml from hci. In *Workshop on Machine Learning meets Human-Computer Interaction, ML*, volume 96, pages 51–58, 1996.

[147] Weng-Keen Wong, Ian Oberst, Shubhomoy Das, Travis Moore, Simone Stumpf, Kevin McIntosh, and Margaret Burnett. End-user feature labeling: A locally-weighted regression approach. In *Proceedings of the 16th international conference on Intelligent user interfaces*, pages 115–124. ACM, 2011.

[148] Yonghui Wu, Joshua Denny, S Trent Rosenbloom, Randolph A Miller, Dario A Giuse, Min Song, and Hua Xu. A prototype application for real-time recognition and disambiguation of clinical abbreviations. In *Proceedings of the 7th international workshop on Data and text mining in biomedical informatics*, pages 7–8. ACM, 2013.

[149] Wayne Xiong, Jasha Droppo, Xuedong Huang, Frank Seide, Michael L Seltzer, Andreas Stolcke, Dong Yu, and Geoffrey Zweig. Toward human parity in conversational speech recognition. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 25(12):2410–2423, 2017.

[150] Hua Xu, Marianthi Markatou, Rositsa Dimova, Hongfang Liu, and Carol Friedman. Machine learning and word sense disambiguation in the biomedical domain: design and evaluation issues. *BMC bioinformatics*, 7(1):334, 2006.

[151] Hua Xu, Peter D Stetson, and Carol Friedman. Combining corpus-derived sense profiles with estimated frequency information to disambiguate clinical abbreviations. In *AMIA Annual Symposium Proceedings*, volume 2012, page 1004. American Medical Informatics Association, 2012.

[152] Jun Xu, Yaoyun Zhang, Hua Xu, et al. Clinical abbreviation disambiguation using neural word embeddings. *Proceedings of BioNLP 15*, pages 171–176, 2015.

[153] Zhao Xu, Xiaowei Xu, Kai Yu, and Volker Tresp. A hybrid relevance feedback approach to text retrieval. In *European Conference on Information Retrieval*, Lecture Notes in Computer Science, pages 281–293. Springer, 2003.

[154] Zuobing Xu, Ram Akella, and Yi Zhang. Incorporating diversity and density in active learning for relevance feedback. In *Advances in Information Retrieval*, pages 246–257. Springer, 2007.

[155] Yi Yang, Shimei Pan, Yangqiu Song, Jie Lu, and Mercan Topkara. User-directed non-disruptive topic model update for effective exploration of dynamic content. In *Proceedings of the 20th International Conference on Intelligent User Interfaces*, pages 158–168. ACM, 2015.

[156] Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pages 412–420, San Francisco, CA, USA, 1997.

[157] Dawei Yin, Yuening Hu, Jiliang Tang, Tim Daly, Mianwei Zhou, Hua Ouyang, Jianhui Chen, Changsung Kang, Hongbo Deng, Chikashi Nobata, et al. Ranking relevance in yahoo search. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 323–332. ACM, 2016.

[158] Hong Yu, Won Kim, Vasileios Hatzivassiloglou, and W John Wilbur. Using medline as a knowledge source for disambiguating abbreviations and acronyms in full-text biomedical journal articles. *Journal of biomedical informatics*, 40(2):150–159, 2007.

[159] Omar Zaidan, Jason Eisner, and Christine Piatko. Using annotator rationales to improve machine learning for text categorization. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 260–267, 2007.

[160] Chengxiang Zhai and John Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of the tenth international conference on Information and knowledge management*, pages 403–410. ACM, 2001.

[161] Chengxiang Zhai and John Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 334–342. ACM, 2001.

[162] Ye Zhang, Matthew Lease, and Byron C Wallace. Active discriminative text representation learning. In *AAAI*, pages 3386–3392, 2017.

[163] Jianlong Zhou and Fang Chen. Making machine learning useable. *International Journal of Intelligent Systems Technologies and Applications*, 14(2):91–109, 2015.

[164] Tianyi Zhou and Jeff Bilmes. Minimax curriculum learning: Machine teaching with desirable difficulties and scheduled diversity. *International Conference on Learning Representations (ICLR)*, 2018.

[165] Xiang Sean Zhou and Thomas S Huang. Relevance feedback in image retrieval: A comprehensive review. *Multimedia systems*, 8(6):536–544, 2003.

[166] Zhi-Hua Zhou. A brief introduction to weakly supervised learning. *National Science Review*, 5(1):44–53, 2017.

[167] Xiaojin Zhu. Semi-supervised learning literature survey. *Computer Science, University of Wisconsin-Madison*, 2(3):4, 2006.

[168] Xiaojin Zhu. Machine teaching: An inverse problem to machine learning and an approach toward optimal education. In *AAAI*, pages 4083–4087, 2015.

[169] Xiaojin Zhu, John Lafferty, and Zoubin Ghahramani. Combining active learning and semi-supervised learning using gaussian fields and harmonic functions. In *ICML 2003 workshop on the continuum from labeled to unlabeled data in machine learning and data mining*, volume 3, 2003.