2015-12-01

Facilitating Corpus Annotation by Improving Annotation Aggregation

Paul L. Felt
Brigham Young University - Provo

Facilitating Corpus Annotation by Improving Annotation Aggregation

Paul L. Felt

A dissertation submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Eric K. Ringger, Chair
Kevin Seppi
Christophe Giraud-Carrier
Deryle W. Lonsdale
Quinn Snell

Department of Computer Science

Brigham Young University

December 2015

ABSTRACT

Facilitating Corpus Annotation by Improving
Annotation Aggregation

Paul L. Felt
Department of Computer Science, BYU
Doctor of Philosophy

Annotated text corpora facilitate the linguistic investigation of language as well as the automation of natural language processing (NLP) tasks. NLP tasks include problems such as spam email detection, grammatical analysis, and identifying mentions of people, places, and events in text. However, constructing high quality annotated corpora can be expensive. Cost can be reduced by employing low-cost internet workers in a practice known as crowdsourcing, but the resulting annotations are often inaccurate, decreasing the usefulness of a corpus. This inaccuracy is typically mitigated by collecting multiple redundant judgments and aggregating them (e.g., via majority vote) to produce high quality consensus answers.

We improve the quality of consensus labels inferred from imperfect annotations in a number of ways. We show that transfer learning can be used to derive benefit from out-dated annotations which would typically be discarded. We show that, contrary to popular preference, annotation aggregation models that take a generative data modeling approach tend to outperform those that take a condition approach. We leverage this insight to develop csLDA, a novel annotation aggregation model that improves on the state of the art for a variety of annotation tasks. When data does not permit generative data modeling, we identify a conditional data modeling approach based on vector-space text representations that achieves state-of-the-art results on several unusual semantic annotation tasks. Finally, we identify a family of models capable of aggregating annotation data containing heterogenous annotation types such as label frequencies and labeled features. We present a multiannotator active learning algorithm for this model family that jointly selects an annotator, data items, and annotation type.

Keywords: crowdsourcing, corpus annotation, semantic embeddings, LDA, rich prior knowledge

ACKNOWLEDGMENTS

First and foremost, thank you, Stephanie. You have been tirelessly supportive from beginning to end. You have been a soundingboard for ideas, an emotional coach, a copy editor, and a friend through the long process of completing a dissertation. In many ways an acknowledgment seems like an insufficient way to describe your role—the reality is closer to joint authorship. Thanks also go to my children, Jane, Nathaniel, Gabriel, and Mackay who have grudgingly allowed me to work many late nights, but never without reminding me that time is precious. Similar thanks go to my parents, Doug and Shelley, as well as my parents-in-law, Scott and Jane. Without their constant support this work would not have been possible.

My advisor, Dr. Eric Ringger, has been instrumental in helping me develop the mental tools necessary to do this work. His remarkable attention to detail and rigor of thought were what originally attracted me to the field of natural language processing. I have benefited greatly from his passion for exploring the unknown and expanding the intellectual, geographical, and culinary horizons of himself and those around him. His role in this work cannot be overstated. Dr. Kevin Seppi has been similarly influential, consistently willing to drop whatever he was doing in order to discuss a new idea.

I would also like to thank Dr. Jordan Boyd-Graber for his close collaboration on the ideas that went into Chapter 5 related to the csLDA model. My committee also deserves a good deal of thanks for repeatedly providing valuable ideas and feedback, giving perspective, and helping my work stay focused in useful directions. Also, many thanks go to fellow students who have, surprisingly, been among my most effective teachers; in particular, Robbie Haertel, Dan Walker, Kevin Black, and Jeff Lund.

Thanks go to the Fulton Supercomputing Lab for providing the computational resources supporting a number of the experiments in this work.

Finally, this work was partly supported by the collaborative NSF Grant IIS-1409739 (BYU) and IIS-1409287 (UMD). Any opinions, findings, conclusions, or recommendations expressed here are those of the authors and do not necessarily reflect the view of the sponsor.

Table of Contents

Chapter 1

Introduction

1.1 Overview of Corpus Annotation

A number of disciplines use annotated corpora in order to accomplish their objectives. Linguists study corpora in order to investigate the usage and structure of language in a particular time period (synchronic linguistics), study the way that languages change over time (diachronic linguistics), and prove or disprove linguistic hypotheses [66]. Cultural historians use corpora to chart the movement of ideas and historical trends [86]. Pedagogical linguists use learner corpora, text and speech generated by language learners, in order to improve the effectiveness of language teaching and acquisition [95]. Computational linguists and computer scientists design corpus-based, or "data-driven," algorithms to automatically accomplish useful tasks such as translating text from one language into another, detecting spam email, analyzing the grammatical structure of text, and identifying mentions of people, places, and events in text [81]. Some of these uses require annotations only at the document level. Others require the annotation of sentences, phrases, words, and even letters.

One compelling use of annotated corpora is to document endangered and dying languages. Many estimates during the last couple of decades put the number of spoken languages in the world at about 6,000, and there is a consensus among linguists that many of those languages are disappearing rapidly [25, 50]. Documenting a language involves generating linguistic artifacts, including linguistically annotated corpora, so that the language can be meaningfully studied in the absence of living native speakers [6, 49].

The Standard Sample of Present-Day American English, more commonly known as the Brown Corpus, was published in 1964 and comprises roughly 1 million words of text published in the United States during the calendar year 1961. At the time of its publication, the Brown Corpus was far larger than any existing digital corpus. The original corpus creators annotated the corpus with word and sentence boundaries, and additionally annotated each word with one of roughly 80 labels encoding a combination of information about the word's form (morphology) and its role in the sentence (syntax) [42]. The LOB Corpus was created as a contemporary British English equivalent to the Brown Corpus, allowing the synchronic analysis of two major varieties of English. In the 1990s, the Brown and LOB corpora were updated with more recent language samples in the FROWN and FLOB corpora, respectively, allowing diachronic analyses of American and British English [126]. The Penn Treebank Corpus (PTB) re-annotated the Brown corpus as well as additional data with a revised annotation scheme [83]. Since then, many other groups have extended portions of these corpora with additional kinds of annotations, including word meanings, semantic frames, and deep and shallow grammatical parses [41, 71, 101, 114]. (Note that this redundant annotation of data with multiple competing annotation schemes will be important in later discussion.) Corpus linguists have used these corpora to make large-scale observations about English usage and characteristics, including estimating the entropy of the English language [11, 43, 64, 68]. Computational linguists have used the corpus to train data-driven models that automatically produce the same kinds of annotations as are attached to the corpus [10, 16, 112].

Despite its size, the Brown Corpus contains only a small sample of English text, limiting its usefulness in many domains. Accordingly, a large number of smaller specialized corpora have been developed in other domains. Organizations such as the Linguistic Data Consortium (LDC) and the European Language Resource Association (ELRA) collect and distribute annotated corpora in many languages. These corpora consist of news articles, spoken language transcripts, blog posts, meeting minutes, emails, internet chats, government and business documents [23, 34]. In addition, corpora have been developed by other groups using almost every kind of text imaginable, including movie reviews [103], memorable movie quotes [28], SMS text messages [2], and even suicide notes [108].

The kinds of annotations applied to text have been equally diverse, including information about the way text is uttered (phonetics and prosody [72]), the main subject of a document (document labeling for classification [81]), the grammar and syntax of the text (part-of-speech tagging, parsing, and morphological analysis [53, 72, 80]), the way the author feels about the subject being discussed (sentiment analysis [103]), important mentioned entities in the text (named entity recognition [130]), which pronouns refer to which entities (anaphora resolution [91]), uncertainty expressed in the text [129], and a host of area-specific tasks such as identifying biomedical entities and their interactions [3].

1.2 Opportunities to Improve Corpus Annotation

Hovy and Lavid [58] describe a number of unresolved issues affecting corpus annotation, including the process that goes into arriving at a satisfactory annotation scheme. An annotation scheme defines the possible structures and values of valid annotations and how they should be applied, and is usually recorded in artifacts such as annotation manuals, annotation GUIs, and meeting minutes. In abstract, the process of annotating a corpus starts with data and some idea about how to annotate it, often informed by linguistic theory, and results in two things: a set of annotations and the annotation scheme used to produce them. However, Hovy and Lavid [58] note that arriving at a satisfactory annotation scheme is actually an iterative process in which changes to the annotation scheme are interleaved with data annotation. As annotators work, they encounter language that challenges the existing annotation scheme, forcing them to revise it. Each time the annotation scheme changes, some cost is incurred as existing annotations are invalidated and must be updated before the corpus is complete. There is no previous work that we are aware of which attempts to re-use or update outdated annotations in a principled way.

Another central problem in corpus annotation is that of annotator inconsistency. Annotators are imperfect and will often disagree with one another or even with themselves when presented with the same data instance twice. A common solution to this problem is to pay various untrusted but low-cost internet workers to redundantly label the same data. Approaches such as majority

vote must then be used to aggregate the redundant annotations and infer a ground truth label. More modern annotation aggregation techniques model the annotation process probabilistically. Many of these aggregation techniques extend the item-response model proposed by Dawid and Skene [30] in which true data labels are modeled as unobserved random variables and annotations are observed as each human contributor corrupts the truth according to their own error characteristics. There is a growing body of variations and extensions to the item-response model that account for such things as correlation among annotator error and item difficulty [15, 57, 75, 109, 134, 136]. However, there is still much room for improvement. In particular, little previous work in annotation aggregation has jointly modeled the observed data and annotations, taking advantage of natural clusters in the data to inform inferred labels.

Finally, current annotation aggregation methods assume that all annotations have the same structure as the hidden label. However, this may not be the most effective way for annotators to convey their knowledge. Recent work in supervised learning shows that great cost savings can be acheived by allowing algorithms to be supervised by rich judgments such as expected label frequencies (e.g., "About 90% of the documents in my corpus should be labeled 'Sports'"), or labeled features (e.g., "Documents containing the word 'golf' should usually be labeled 'Sports'"). However, none of these methods are designed to accomodate redundant or untrusted judgments [46].

1.3 Thesis Statement

Judgments from multiple imperfect annotators can be augmented with information from outdated annotations, patterns in the data being annotated, and alternative types of judgments in order to improve annotation aggregation and reduce annotation cost for a variety of tasks.

1.4 Dissertation Organization

Many of the chapters of this dissertation have been published previously. Those which have not (Chapters 6 and 7) are currently in submission. In this section we list publications and briefly

summarize the contributions of each chapter. In addition, a brief introduction precedes each chapter in the text, citing publication information and describing the paper's place in the larger narrative.

- Chapter 2: P. Felt, E. Ringger, K. Seppi, and K. Heal. Using Transfer Learning to Assist Exploratory Corpus Annotation. In *Proceedings of LREC* 2014.
- Chapter 3: P. Felt, R. Haertel, E. Ringger, and K. Seppi. MomResp: A Bayesian Model for Multi-Annotator Document Labeling. In *Proceedings of LREC* 2014.
- Chapter 4: P. Felt, E. Ringger, K. Seppi, and R. Haertel. Early Gains Matter: A Case for Preferring Generative over Discriminative Crowdsourcing Models. In *Proceedings of NAACL* 2015.
- Chapter 5: P. Felt, E. Ringger, J. Boyd-Graber, and K. Seppi. Making the Most of Crowdsourced Document Annotations: Confused Supervised LDA. In *Proceedings of CoNLL* 2015. [Best paper award].

Chapter 2 presents two algorithms for using information from outdated annotations to improve the quality of pre-annotations and thus reduce annotation cost. Both algorithms significantly improve preannotations, but displaying complementary performance characteristics. One is fast to train but provides predictions slowly. The other trains slowly but provides predictions quickly.

In Chapter 3 we implement MOMRESP, an annotation aggregation model that seeks to jointly explain observed annotations and text documents. Generatively explaining the data allows this algorithm to benefit from even unannotated data and learn very quickly, outperforming alternatives when few annotations are available. At the time of writing this constituted the first implemented annotation aggretation model with a generative model of the data.

Chapter 4 presents an alternative inference algorithm for MOMRESP that is faster and yields higher quality answers than what came before. This paper compares annotation aggregation models with generative and those with conditional data models, and argues that generative data models (until now less explored than their conditional counterparts) should be preferred, when available.

Chapter 5 presents CSLDA, a novel annotation aggregation model. Like MOMRESP, CSLDA is a model with a generative data component, jointly explaining both the data as well as the observed annotations. However, CSLDA uses a more sophisticated data model based on topic modeling. This enhancement results in greatly improved annotation aggregation performance for classification tasks where a document's topical content is indicative of its true label.

Chapter 6 acknowledges that sometimes data does not lend itself to generative modeling. In these cases, we show that recent work in vector space document representations can be used to give conditional annotation aggregation models many of the same advantages as generative models.

In Chapter 7 we identify a family of models that can aggretate diverse types of annotations, including expected label frequencies and labeled features. We derive a training algorithm for a concrete member of this model family and use it to demonstrate that labeled features can greatly enhance the quality of inferred labels at relatively low cost. In addition, we present an active learning algorithm for this model and show that active learning can be used to further reduce annotation costs by judiciously selecting an annotator to query, a data instance that they should annotate as well as the type of annotation that they should provide.

Chapter 2

Using Transfer Learning to Assist Exploratory Corpus Annotation

Published in Proceedings of LREC 2014 [38]

This chapter treats the problem of iterative annotation scheme revision, affirming the hypothesis that outdated annotations may be useful in reducing annotation effort (by improving the quality of pre-annotations). This is the only chapter that deals explicitly with annotation scheme change.

Abstract

We describe an under-studied problem in language resource management: that of providing automatic assistance to annotators working in exploratory settings. When no satisfactory tagset already exists, such as in under-resourced or undocumented languages, it must be developed iteratively while annotating data. This process naturally gives rise to a sequence of datasets, each annotated differently. We argue that this problem is best regarded as a transfer learning problem with multiple source tasks. Using part-of-speech tagging data with simulated exploratory tagsets, we demonstrate that even simple transfer learning techniques can significantly improve the quality of pre-annotations in an exploratory annotation.

2.1 Exploratory Corpus Annotation (ECA)

Because corpora are useful for investigating the structure of language, studying the way that languages change over time, testing linguistic hypotheses, charting the movement of ideas and historical trends, and even improving the effectiveness of language teaching and acquisition, they

are an essential linguistic resource [66, 95]. One of the most urgent needs for annotated corpora is in the realm of under-resourced and endangered language documentation [6, 25, 49, 50].

In domains such as under-resourced language documentation, annotation is unavoidably exploratory and iterative in nature [58]. The annotator proposes an annotation scheme, annotates data, and then revises that annotation scheme in light of insights generated by applying the annotation scheme to real world data (Figure 2.1), a process which for brevity we refer to as ECA (exploratory corpus annotation). ECA results in a sequence of possibly disjoint annotation sets, or "versions", $V_1 \oplus \ldots \oplus V_K = \boldsymbol{V}$, where each V_v consists of data and annotations, $\{(x_i, y_i)\}_{i=1}^{N_v}$, produced according to V_v's annotation scheme.

Each time the annotation scheme changes, some cost is incurred as existing annotations are invalidated and must be updated before the corpus is complete. The cost associated with evolving annotation schemes is largely a hidden cost, since few annotation projects record or report internal changes. For example, the Natural Language Processing (NLP) Lab at BYU is collaborating with scholars of ancient languages at the Neal A. Maxwell Institute for Religious Scholarship to create a large corpus of annotated Classical Syriac.[1] Although significant time was spent at the outset defining the annotation scheme that would be used, as preliminary data has been annotated at least a dozen updates have already been made to the annotation scheme. Since we are starting with a sizable body of already annotated text, some of these changes have required considerable time and effort to implement (via re-annotation).

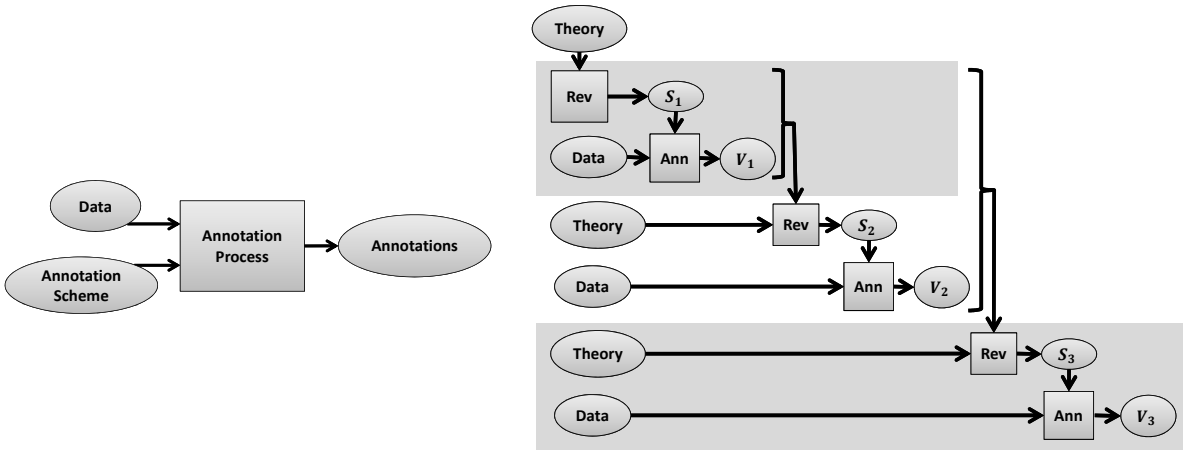
Annotation scheme revisions are especially likely in exploratory annotation scenarios dealing with languages or linguistic theories that have not previously been codified into annotation schemes. However, revisions can occur even in well established annotation tasks such as English part-of-speech tagging and parsing. When creating the Penn Treebank corpus, Marcus et al. [83] re-annotated the Brown corpus data with revised part-of-speech tags. Additionally, Marcus et al. report that after publishing the Penn Treebank, they identified a variety of limitations and

[1] `http://cpart.maxwellinstitute.byu.edu/home/sec/`

inconsistencies in their annotation scheme for English syntactic parsing and subsequently spent a good deal of effort repairing the parsing scheme and re-annotating data for future releases [82].

A more extreme case comes from the SUSANNE corpus, another derivative of the Brown Corpus, annotated with very detailed parsing information. The 512-page book describing the SUSANNE annotation scheme required twelve years of work to finish, and the author describes the accompanying 130,000-word corpus as a "by-product of the work of creating the SUSANNE annotation scheme" [115]. These examples underscore the effort involved in developing a satisfactory annotation scheme, even for mainstream languages and linguistic annotation tasks.

The costs involved in iteratively improving an annotation scheme mean that budget-constrained corpus developers often must choose between developing a linguistically optimal annotation scheme and generating useful amounts of annotated data. To make matters worse, statistical pre-annotation—the traditional method of reducing annotation overhead—is hampered by the lack of a self-consistent training set.



(a) Corpus annotation with a pre-defined, unchanging annotation scheme

(b) Exploratory corpus annotation (ECA)
Ann = annotation generation
Rev = revising the annotation scheme

Figure 2.1: Two Kinds of Corpus Annotation

2.2 Previous Work

Numerous annotation projects have shown that assisting annotators with good pre-annotations is essential to annotator efficiency and accuracy. Studies in English part-of-speech tagging, Chinese parsing, information extraction, named entity recognition, and Semitic morphological analysis all demonstrate that high accuracy pre-annotations correlate strongly with low annotation cost [21, 26, 35, 45, 83]. This point is critical to our future decision (see Section 2.4) to focus on increasing model accuracy as a stand-in for reduced cost.

Because accurate pre-annotation models are so effective in reducing annotation effort, much work has been done to train high quality models with as little data as possible. The active learning literature aims to reduce the cost required to train a model by selecting instances for annotation that are likely to be most informative [117]. Weakly supervised techniques attempt to speed model training by learning from unlabeled instances, or by allowing annotators to communicate their knowledge to the model by specifying labels or constraints that are applicable to large classes of data instances [33, 46, 74, 113].

We know of no previous work that explicitly addresses the problem of providing automatic assistance for ECA; however, corpus developers have naturally gravitated towards the solution of adapting knowledge from the data in out-of-date versions. For example, the creators of the Penn Treebank corpus used an altered version of the Brown Corpus's annotation scheme, which can be seen as an example of a single large step in the iterative process of ECA [83]. Although the existing Brown Corpus annotations were unsuitable for direct use, the creators of the Penn Treebank used an automatic tagging model trained on heuristically modified Brown Corpus data to automatically pre-annotate Penn Treebank data. Although imperfect, these pre-annotations effectively doubled annotation speed, greatly reducing annotation cost [22, 42].

2.2.1 Transfer Learning

Using knowledge from one or more source tasks to improve performance on a target task, as the Penn Treebank developers did, is known as transfer learning, and is an area of active research

within machine learning. Providing pre-annotations for ECA fits naturally into the transfer learning framework. The following definition of transfer learning borrows notation and ideas from Pan and Yang [102], but with minor changes to highlight connections between transfer learning and the motivation presented in Section 2.1.

Definition 1 (Transfer Learning) *Let $\mathcal{D}$ denote a **domain** comprising the feature space $\mathcal{X}$ and a distribution $p(x)$ over data $x \in \mathcal{X}$. Let $\mathcal{T}$ denote a **task**, or annotation scheme, comprising a feature space $\mathcal{X}$, a label space $\mathcal{Y}$, and a labeling function $f : x \rightarrow y$ where $x \in \mathcal{X}$ and $y \in \mathcal{Y}$.[2] Finally, let **version** V_t be the set of annotations produced according to the annotation scheme of task $\mathcal{T}_t$. Then the goal of transfer learning is to use data from all source versions $\boldsymbol{V}_{1..t-1}$ to improve our ability to model the target annotation scheme $\mathcal{T}_t$.*

Definition 1 encompasses a large number of scenarios. There may be one or many source versions. Different quantities of data and annotations may be available in any given version. Furthermore, each version is associated with an annotator, domain, and task, and therefore may differ from other versions in terms of $\mathcal{X}$, $p(x)$, $\mathcal{Y}$, or f. Each of these differences can be understood via simple examples. Text and images come from domains $\mathcal{D}_1$, $\mathcal{D}_2$ where $\mathcal{X}_1 \neq \mathcal{X}_2$. Poetry and newswire text come from domains where $p_1(x) \neq p_2(x)$. When two part-of-speech tagging tasks use different tagsets, then $\mathcal{Y}_1 \neq \mathcal{Y}_2$; when they assign different meanings to the same tag, $f_1 \neq f_2$. Each setting of these variables in a versioned dataset $\boldsymbol{V}_{1...t}$ corresponds to a different transfer learning scenario.

Much work on transfer learning for NLP is currently in domain adaptation, the transfer setting in which $\mathcal{T}_s = \mathcal{T}_t$, $\mathcal{D}_s \neq \mathcal{D}_t$, and the domains differ only in the marginal distribution of the data, $p_s(X)$ and $p_t(X)$. An example of domain adaptation would be using Wikipedia text in which named entities (people, places, events, etc.) have been labeled in order to improve named entity recognition in movie reviews.

[2] Although f may be approximately described in annotation manuals, the true f is generally unseen. In probabilistic approaches, f is often modeled as $p(y|x)$.

Other notable related work includes multi-task learning, a transfer setting in which $\mathcal{D}_s = \mathcal{D}_t$ and there are multiple tasks that all differ from one another. Multi-task learning is unusual in that no source/destination distinctions made among tasks [19]. For example, Collobert et al. [24] construct a system that simultaneously learns part-of-speech tagging, named entity recognition, chunking, and semantic role labeling. Each task helps to inform the others, leading to higher performance on all tasks learned jointly than was possible for any individual task when learned individually.

2.3 ECA as Transfer Learning

We formally define the problem of providing machine assistance in the setting of exploratory corpus annotation as a transfer learning problem and introduce some simple solutions adapted from previous work. The solutions described below are appealing since they are conceptually simple and easy to implement using existing models as building blocks.

Definition 2 (Exploratory Corpus Annotation) *This is a transfer learning setting in which the following are true. There are multiple source tasks $\mathcal{T}_{1..t-1}$. For each pair i, j of source tasks, $\mathcal{D}_i = \mathcal{D}_j$ and $\mathcal{T}_i \neq \mathcal{T}_j$. Finally, each source version has at least some labeled data. Little or no labeled data is available for the target version V_t.*

Only a few of the possible transfer learning settings are commonly studied, and none of those match Definition 2. ECA is unusual and interesting from a transfer learning point of view because it has multiple source tasks and there is often a sequential relationship among the tasks.

2.3.1 Baselines: TGTTRAIN and ALLTRAIN

Let TGTTRAIN be the approach that ignores all data from source tasks and trains a traditional supervised classification model only on the current target data $\boldsymbol{V}_t$. We can expect TGTTRAIN to do well when $\boldsymbol{V}_t$ is large and badly when $\boldsymbol{V}_t$ is small, such as at the beginning of a project or just after a change is made to the annotation scheme. TGTTRAIN corresponds to annotation projects that

discard out-dated annotations when a new version is introduced. In practice, this tends to happen during the initial stages of a project, when the perceived value of the information being lost is low.

Let ALLTRAIN be the algorithm that trains a traditional supervised classification model on all available data $V_{1...t}$, ignoring version boundaries. We would expect ALLTRAIN to do well when there are few differences between the source and target datasets, and badly when there are large differences.

2.3.2 STACK

STACK refers to an adaptation of stacked generalization, in which traditional supervised models are trained on each of the datasets, and a higher level model is trained to accomplish the target task using the predictions of the lower level models as features [137]. The higher level model can potentially discover patterns in the errors of the underlying models in order to know which are trustworthy in which contexts, and whether their guesses are wrong in ways that can be predictably mapped to the correct answer.

2.3.3 AUGMENT

AUGMENT is a simple and effective domain adaptation technique proposed by Daumé III [29]. AUGMENT moves the data into a feature space that allows traditional supervised learning techniques to find commonalities and differences among data from different domains. It is assumed that there are two datasets: the source X_s and the target X_t. Each source feature vector is mapped into the new feature space by the kernel function $\Phi^s(\boldsymbol{x}) = \langle \boldsymbol{x}, \boldsymbol{x}, 0 \rangle$, and each target feature vector is mapped by the function $\Phi^t(\boldsymbol{x}) = \langle \boldsymbol{x}, 0, \boldsymbol{x} \rangle$. Thus each feature has a source-specific version, a target-specific version, and a general version in the new feature space. This can be generalized to the context of multiple source domains by defining $\Phi^{s1}(\boldsymbol{x}) = \langle \boldsymbol{x}, 0, 0, ..., \boldsymbol{x} \rangle$, $\Phi^{s2}(\boldsymbol{x}) = \langle 0, \boldsymbol{x}, 0, ..., \boldsymbol{x} \rangle$,...,$\Phi^t(\boldsymbol{x}) = \langle 0, 0, ..., \boldsymbol{x}, \boldsymbol{x} \rangle$.

2.4 Experiments

We would like to test the hypothesis that transfer learning can improve pre-annotations for ECA. However, evaluating a model in this setting requires access to corpora that recorded every version of the data $\boldsymbol{V}$ since the beginning of the project. We are aware of no such datasets. However, we can simulate such corpora by starting with an existing annotated corpus and probabilistically generating sequences of intermediate versions that explain how that corpus's annotation scheme might have come to be. For example, to start in familiar territory, we use the following process to create versioned datasets explaining the derivation of the Penn Treebank's part-of-speech (POS) tagged data.

Algorithm 1 Simulate Versioned POS Datasets

Given: $GoldData$ is the reference dataset
Given: $GoldTags$ is the reference tagset
1: $Tags \leftarrow CompositeTag(GoldTags)$
2: dataset $\leftarrow \{\}$
3: **while** $Tags \neq$ GoldTags **do**
4: $\quad op \leftarrow sample(\{\text{SPLIT,MOVE,MERGE}\})$
5: $\quad Tags \leftarrow apply(op, Tags)$
6: $\quad \kappa \leftarrow sampleVersionSize()$
7: $\quad dataset \leftarrow annotate(\kappa, Tags, GoldData)$
8: **return** $dataset$

Algorithm 1 starts by grouping all the reference tags into a single composite tagset, then iterates between altering the tagset and annotating data until the original tagset is reached. A SPLIT represents an annotator deciding that the largest composite tag in the tagset is too broad and dividing it. A MERGE represents an annotator deciding that the distinctions between two tags are too fine and lumping them together. A MOVE represents an annotator moving one of the reference tags out of one composite tag and into another; in other words, deciding that a set of words that was previously labeled as something would be better labeled something else.

In order to be linguistically reasonable, splits are determined by finding a min-cut in the graph of reference tags, where reference tags are connected with strong ad-hoc weights if they are in the same family of tags (e.g., nouns, verbs, punctuation, etc), and weak weights otherwise. Merges

and moves are chosen by sampling from a distribution over reference tag pairs where pairs that are identified by the Penn Treebank tagging guidelines as confusable (25 of these) or very confusable (9 of these) are more probable [116].

Finally, $sampleVersionSize()$ is implemented by hypothesizing that annotation projects alternate between small annotation batches for development and large batches for production approaching the size of the desired corpus, N, as the tagset converges on the reference set. The final mix favors development mode, since production mode involves heavy costs in the later stages (Figure 2.2).

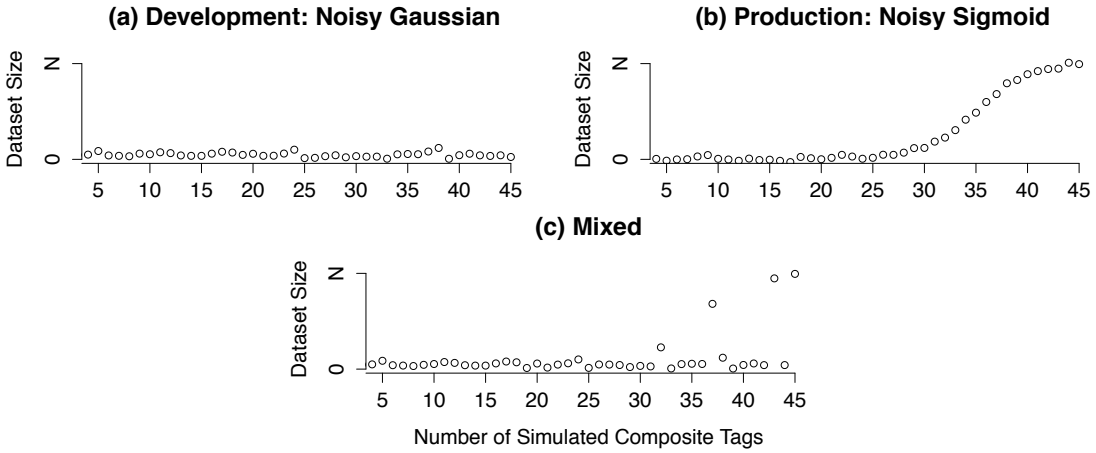


Figure 2.2: Example draws from $sampleVersionSize()$

This simulated data is clearly not ideal, and we and are actively working on developing real-world ECA data based on annotation projects we are involved in [37]. However, in the meantime, simulated data allows us to make cautious observations about the characteristics of the problem and projections about the potential of transfer learning models to improve pre-annotation for ECA.

We used Algorithm 1 to generate 30 diverse datasets, choosing values for the simulation parameters at random. We used maximum-entropy Markov models ("maxent taggers") with standard features [131] to implement the transfer algorithms described in Section 2.3. Figure 2.3 shows the learning curve of each algorithm on a single dataset. TGTTRAIN's learning curve shows deep valleys at each version transition, because it is equivalent to beginning an entirely new learning curve at the beginning of each version. ALLTRAIN, on the other hand, shows a much smoother

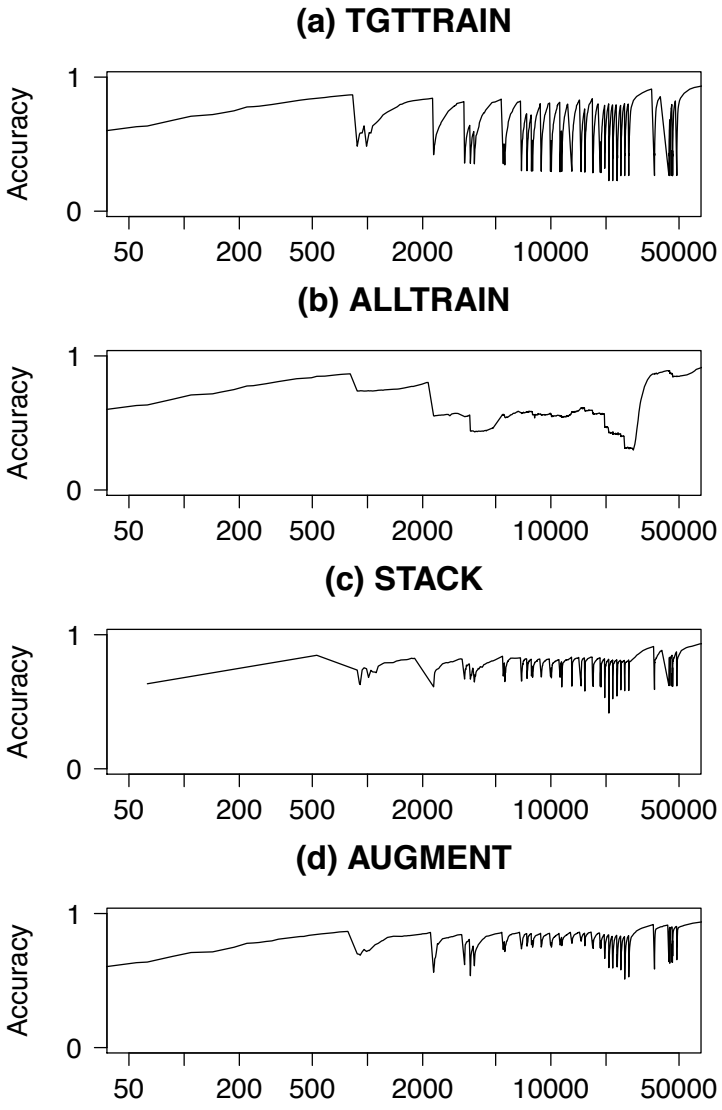


Figure 2.3: An example learning curve for each algorithm on the same dataset.

pattern. Using old data allows it to avoid the valleys seen in TGTTRAIN, but hurts its ability to reach high peaks quickly. STACK and AUGMENT both have peaks similar to TGTTRAIN, but manage to recover more quickly from version changes and avoid the low valleys.

	Heldout AAUC	Train Secs	Eval Secs
TGTTRAIN	225.4	0.2	0.003
ALLTRAIN	218.0	7.7	0.004
STACK	**240.3**	6.5	<u>0.046</u>
AUGMENT	**246.4**	<u>60.9</u>	0.006

Table 2.1: Bolded accuracies are significantly (p-val¡0.01) better than non-bolded competitors. Underlined times are significantly (p-val¡0.01) worse than non-underlined competitors. AAUC is averaged over 30 datasets. "Train Secs" means model training time averaged over all datasets. "Eval Secs" means average seconds to infer the labeling of a single sentence.

Because we are interested in a model that performs well at all stages of the ECA process, we need to compare entire learning curves rather than just final accuracy. A natural summary statistic for the quality of a learning curve is average area under the curve (AAUC). An accurate estimate of AAUC requires good resolution on the learning curve, so we evaluate between 500 and 1000 points on each learning curve, sampling more densely around version transitions. In Table 2.1 we report the average AAUC of each algorithm over all the simulated datasets, along with model timing statistics.

STACK and AUGMENT significantly outperform both TGTTRAIN and ALLTRAIN. Notice that AUGMENT is particularly slow to train, since its feature space has been expanded linearly in K, the number of versions. STACK is unusually slow at inference time, since it must solicit predictions from K subordinate models as features. The fact that TGTTRAIN was relatively easy to beat is encouraging, suggesting that there is room for more sophisticated transfer learning to make significant improvements over the baseline approach.

2.5 Conclusions and Future Work

We have described the problem of providing automatic assistance to annotators working in exploratory settings. We have argued that this problem should be regarded as a transfer learning problem, and shown that existing transfer learning techniques can be adapted to significantly improve the quality of pre-annotations in simulated exploratory part-of-speech tagging. Corpus annotators working in novel annotation domains should be encouraged by these results and by the existence of a rich body of transfer learning work to draw on.

We plan to develop models that leverage the sequential nature of the versions. We also plan to apply the insights developed in this paper to improve pre-annotations for annotators engaged in real-world annotation projects. Finally, in order to apply these techniques seamlessly in annotation projects, it would be beneficial to discover a way of learning to automatically identify the boundaries between versions so that annotators need not manually identify annotation scheme changes.

Chapter 3

MOMRESP: A Bayesian Model for Multi-Annotator Document Labeling

Published in Proceedings of LREC 2014 [36]

This chapter shifts gear from the previous chapter, moving away from transfer learning and toward the crowdsourcing objective of inferring labels from untrusted annotations. This chapter begins a thread of work continued in Chapters 4, 5, and 6 that investigates how best to use evidence from the data itself (the words in a document) to improve the quality of inferred labels.

Abstract

Data annotation in modern practice often involves multiple, imperfect human annotators. Multiple annotations can be used to infer estimates of the ground-truth labels and to estimate individual annotator error characteristics (or reliability). We introduce MOMRESP, a model that improves upon item response models to incorporate information from both natural data clusters as well as annotations from multiple annotators to infer ground-truth labels for the document classification task. We implement this model and show that MOMRESP can use unlabeled data to improve estimates of the ground-truth labels over a majority vote baseline dramatically in situations where both annotations are scarce and annotation quality is low as well as in situations where annotators disagree consistently. Correspondingly, in those same situations, estimates of annotator reliability are also stronger than the majority vote baseline. Because MOMRESP predictions are subject to label switching, we introduce a solution that finds nearly optimal predicted class reassignments in a variety of settings using only information available to the model at inference time. Although

MOMRESP does not perform well in annotation-rich situations, we show evidence suggesting how this shortcoming may be overcome in future work.

3.1 Introduction

To build labeled corpora and to train NLP models using supervised learning methods we rely heavily on imperfect annotators, ranging from nearly perfect experts to very imperfect workers in crowd-sourcing settings. Often we rely on multiple annotators providing redundant annotations to improve the quality of the resulting labeled data. Although annotations produced by fallible annotators are not individually trustworthy, they can be used collectively to infer the correct labels for data and also to estimate annotator error characteristics. To be clear, in this paper, we use the term *annotation* to refer to human input and *label* to refer either to gold-standard reference labels or to model-predicted labels.

In the machine learning literature, annotations are useful first and foremost as a means of inducing a model that can be used to predict the labels of new data. In other fields, corpus labels are interesting in their own right because they facilitate meaningful data analysis. For example, corpus linguists employ corpora labeled with linguistic categories to aid in the analysis of diachronic trends and patterns in language [47, 68]. Estimating ground-truth labels is closely related to the task of learning individual annotator error characteristics, since these can be used to upweight trustworthy annotations and downweight others. Annotator error profiles can also be used to elucidate opportunities to retain, train, and advise annotators.

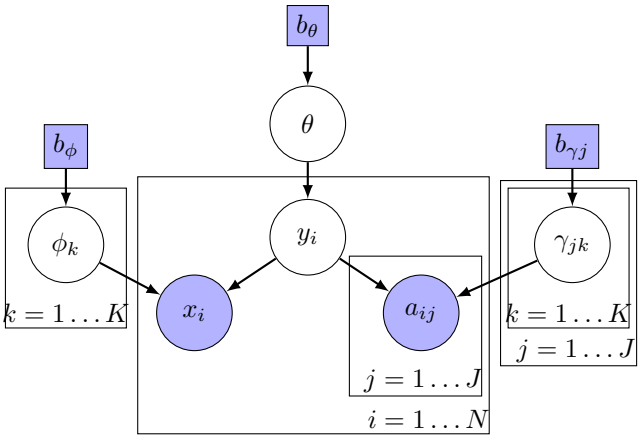
In this paper we introduce, implement, and evaluate a model for the inference of ground-truth labels and annotator reliability that utilizes features of the data as well as annotations from multiple annotators; the two sources of information reinforce one another and make possible ground truth inference that is superior to the available baselines.

3.2 Previous Work

Dawid and Skene [30] laid the groundwork for inferring ground-truth labels and estimating individual annotator accuracy by proposing a statistical model known as the "item-response" model. Carpenter [15] and Pasternack and Roth [105] describe models that are essentially Bayesian versions of the same model. There is a growing body of variations and extensions to this simple item-response model to account for correlations among annotators or item difficulty [15, 57, 75, 109, 122, 134, 136, 142]. Only human annotations are leveraged in the previously described approaches. However, Lam and Stork [69], in effect, extend the item-response model such that the inferred label of a data item depends not only on annotations, but also on the features of the data instance itself. Carroll et al. [18] propose a model that similarly takes advantage of data features. However, neither of these proposed models is implemented or evaluated in previous work.

Inferring labels and estimating annotator trustworthiness can be seen as special cases of the fact-finding task [55, 106] in which the annotators are sources of information and their annotations are claims. However, the general task of fact-finding is far more complex than the special case of data annotation.

Furthermore, there is a good deal of work focused on the machine learning problem of training a useful model from multiple error-prone annotations, especially in light of recent interest in crowd-sourcing. However, much of this work relies more on heavy redundancy than sophisticated aggregation techniques [123]. The *de facto* standard approach is to infer ground-truth corpus labels using a simple majority vote, assess inter-annotator agreement to gain some confidence in the quality of the labels, and then pass the resulting labeled corpus to a machine learning algorithm in order to train the desired model in the traditional batch manner. Sheng et al. [118] construct training corpora in which each instance, annotated n times, is replicated with each annotation and weighted by $\frac{1}{n}$. Models trained from this 'soft' labeling are shown to always be at least as good as, and usually better than, those trained by majority vote. Jurgens [61] experiments with constructing similar weighted datasets using explicit annotator input.



$J :=$ number of annotators
$K :=$ number of labels
$N :=$ number of instances
$F :=$ number of features(word types)

$$\theta \sim SymDir(b_\theta), \qquad dim(\theta) = K$$
$$\forall k: \ \phi_k \sim SymDir(b_\phi), \qquad dim(\phi_k) = F$$
$$\forall j,k: \ \gamma_{jk} \sim Dir(b_{\gamma_{jk}}), \qquad dim(\gamma_{jk}) = K$$
$$dim(\gamma_j) = K \times K$$
$$\forall i: \ y_i|\theta \sim Cat(\theta), \qquad y_i \in \{1 \dots K\}$$
$$\forall i: \ x_i|y_i, \phi \sim Multinom(|x_i|_1, \phi_{y_i}), \qquad dim(x_i) = F$$
$$\forall i,j: \ a_{ij}|y_i, \gamma_j \sim Multinom(|a_{ij}|_1, \gamma_{j,y_i}), \qquad dim(a_{ij}) = K$$

Figure 3.1: MOMRESP: a generative Bayesian model for inferring ground-truth labels y from the annotations a of multiple annotators and from data x while modeling individual annotator accuracies γ. ($|\cdot|_1$ denotes the L_1-norm.)

3.3 Methods

To take advantage of the opportunities presented by natural data clusters and multiple, potentially sparse annotations, we now present MOMRESP, a generative Bayesian model that can infer ground-truth labels from multiple, noisy annotators and simultaneously estimate the error characteristics of each annotator. We also present an inference algorithm for the model.

3.3.1 Model

MOMRESP is inspired by Bayesian models described, but not implemented or evaluated, in previous work [17, 18, 51]. It is called MOMRESP because it adds a mixture-of-multinomials (MOM) data component to a Bayesian item-response model. The model is based on three main principles:

1. Ground-truth labels y are unobservable.

2. All annotations a may carry useful information, even when incorrect.

3. A document's words x can help in determining a document's label y.

Figure 3.1 presents the model as a directed graphical model. The model assigns probability to variables as though they were generated according to the following process. First, label class proportions θ and word proportions ϕ_k for each label class $k \in \{1 \dots K\}$ are drawn. (K is the number of label classes.) For each annotator $j \in \{1 \dots J\}$, a probability vector γ_{jk} is drawn specifying the probabilities of the annotations annotator j is likely to produce in the presence of the label class k. (J is the number of annotators.) Thus, γ_j can be seen as an annotator j-specific confusion matrix (alternatively: a contingency table or error matrix), where each row sums to 1. For each of the N documents, the ith annotated document is generated by drawing a document label y_i from the categorical distribution $Cat(\theta)$ and then drawing words x_i from a multinomial distribution with parameters ϕ_{y_i} and drawing annotations a_{ij} from a multinomial distribution with parameters γ_{jy_i}.

Notice that in the absence of annotations a, this model reduces to a mixture-of-multinomials document clustering model. In the absence of any data x, this model becomes a multinomial item-response model—a Bayesian version of the approach of Dawid and Skene [30].

3.3.2 Inference

Having formulated a generative model and specified a distribution over every variable in the model means that we can apply standard Bayesian statistical machinery such as Markov Chain Monte Carlo (MCMC) inference to the task of inferring the values of hidden labels y and annotator error characteristics γ given some observed set of documents x and annotations a, however incomplete the annotations may be. We derive an efficient collapsed Gibbs sampler for y. Then the per-annotator confusion matrix γ is constructed from these samples after-the-fact, as described below.

Liu [78] provides empirical and theoretical evidence that analytically integrating out parameters where possible (i.e., 'collapsing' the model) improves Gibbs sampling. Accordingly, we derive a collapsed sampler by analytically integrating out the parameters of the model (θ, ϕ, γ).

Let $p(y_i = c|y_{i' \neq i})$ be the full conditional distribution for y_i. It represents the conditional distribution over possible values c for y_i given the data and annotations and sample values for all other latent class labels $y_{i'}$ where $i' \neq i$. We omit its derivation due to space constraints, merely noting that it is very similar to the derivation for a mixture-of-multinomials model [133]. For notational simplicity, we define the following count variables, which are disambiguated by the letters of their superscripts. For the purpose of the full conditionals, each count variable excludes the counts associated with the instance i being sampled (indicated by sums over $i' \neq i$):

$$\begin{aligned}
\forall k: \; n_k^{(\theta)} &= \sum_{i' \neq i}^{N} \mathbb{1}(y_{i'} = k) \\
\forall j, k, k': \; n_{jkk'}^{(\gamma)} &= \sum_{i' \neq i}^{N} a_{i'jk'}^{\mathbb{1}(y_{i'}=k)} \\
\forall k, f: \; n_{kf}^{(\phi)} &= \sum_{i' \neq i}^{N} x_{i'f}^{\mathbb{1}(y_{i'}=k)}.
\end{aligned}$$

That is, $n_k^{(\theta)}$ is the number of instances currently labeled k; $n_{jkk'}^{(\gamma)}$ is the number of times that annotator j chose annotation k' on instances labeled k; and $n_{kf}^{(\phi)}$ is the number of times word (or feature) f occurs with instances having an inferred label value of k.

Using these count variables, the full conditional distribution for the MOMRESP model has the following form:

$$\begin{aligned}
&p(y_i = c|y_{i' \neq i}) \propto (b_\theta + n_c^{(\theta)}) \\
&\cdot \prod_{j=1}^{J} \left(\sum_{k'=1}^{K} \left(b_{\gamma_j} + n_{jck'}^{(\gamma)} \right) \right)^{-\overline{\sum_{k'=1}^{K} a_{ijk'}}} \prod_{k'=1}^{K} \left(b_{\gamma_j} + n_{jck'}^{(\gamma)} \right)^{\overline{a_{ijk'}}} \\
&\cdot \left(\sum_{f=1}^{F} \left(b_\phi + n_{cf}^{(\phi)} \right) \right)^{-\overline{\sum_{f=1}^{F} x_{if}}} \prod_{f=1}^{F} \left(b_\phi + n_{cf}^{(\phi)} \right)^{\overline{x_{if}}}
\end{aligned} \tag{3.1}$$

where the notation $x^{\bar{n}}$ represents the rising factorial defined as $x^{\bar{n}} := x(x+1)(x+2)\dots(x+n-1)$ and $x^{-\bar{n}} := \frac{1}{x^{\bar{n}}}$.

Gibbs sampling yields samples that consist of label values for each y_i. In our sampling experiments, we use the label values y in the single last sample of the Markov chain as the predicted corpus labels. The experiments of Walker [133] with a sampler for the mixture-of-multinomials model indicate that taking the last sample is an efficient way to summarize the Markov chain without sacrificing the quality of the samples. In order to estimate the accuracy vector γ_{jk} for annotator j on label k, we compute the mean of the Dirichlet distribution for γ_{jk} using the sufficient statistics $n^{(\gamma)}_{jkk'}$. The mean values of θ and ϕ are similarly estimated using $n^{(\theta)}_{k}$ and $n^{(\phi)}_{kf}$, respectively. In order encourage our sampler to explore effectively, we anneal our sampler with 250 samples at each temperature of the following schedule: $1000, 500, 200, 100, 50, 20, 10, 5, 2, 1$. This is followed by an additional 500 samples without annealing.

3.3.3 Class Correspondence Correction

The problem of class correspondence, or label switching, arises in unsupervised and semi-supervised mixture models [127]. The model described in Section 3.3.1 assigns the same probability to any permutation of the inferred class label types assignable to y. That is, we could relabel every y whose value is class A to class B and vice versa as long as we also swapped the rows in γ and ϕ that correspond to class A and class B. The model has no reason (aside from weak prior preferences) to prefer one of these solutions over the other. However, inferred labels y are only useful when they align with true gold-standard labels; therefore, we must solve the class correspondence problem.

Stephens [127] points out that the problem of finding an optimal (with respect to some loss function) reassignment of inferred label classes can be posed and efficiently solved as an instance of the well-known assignment problem. In this formulation, we have K 'source' classes as a result of sampling, and each source class will be assigned to one of K 'destination' classes. Let $L(c', c'')$ be some loss function that defines the cost of (re-)assigning source class c' to destination class c''. Each possible assignment from c' to c'' is represented with a boolean variable $\pi_{c'c''}$, where $\pi_{c'c''} = 1$

indicates the presence of an assignment in the final solution, and $\pi_{c'c''} = 0$ indicates absence. Our objective is to discover the solution π that minimizes

$$\sum_{c'=1}^{K} \sum_{c''=1}^{K} \pi_{c'c''} L(c', c'')$$

subject to the following constraints

$$\forall c' \in \{1 \dots K\} : \sum_{c''=1}^{K} \pi_{c'c''} = 1$$

$$\forall c'' \in \{1 \dots K\} : \sum_{c'=1}^{K} \pi_{c'c''} = 1$$

$$\forall c', c'' : \pi_{c'c''} \geq 0$$

The first set of constraints ensures that each source class is assigned once. The second set ensures that each destination class receives a single assignment. The final constraints ensure non-negativity. Although this formulation allows for fractional assignments, the optimal solution is guaranteed to have integer values because the constraint matrix is totally unimodular [12].

3.3.4 Loss Functions

The problem remains of choosing a loss function L. Intuitively, L's purpose is to penalize decisions that assign inferred label classes to the wrong gold-standard label classes. We experiment with three loss functions. The first relies on gold-standard labels; the second two do not.

CorrectCount

Confronted with a similar semi-supervised class correspondence problem, Nigam et al. [100] note that it would be a relatively simple matter to align a small number of latent classes *manually* with true classes by inspecting a few documents assigned to each class. The CORRECTCOUNT loss function automates this insight by assuming that some number of gold-standard labels are

available for the purposes of empirical class alignment. Predicted labels y are compared with gold-standard labels to compute an error matrix E where $E_{kk'}$ is the number of times a document with the gold-standard label k was predicted by the model to have label k'. The ideal E is diagonal because a diagonal error matrix represents no errors in the predicted labels. Transposed columns in E indicate that the model is "calling things by the wrong name," a symptom of label switching in the inference process. Correcting transposed columns involves changing the predicted label class at index c' to some better position c''. CORRECTCOUNT measures the magnitude of the diagonal entry of the moved column in its destination position:

$$L_{\text{CC}}(c', c'') = -E_{c''c'}$$

This loss function favors column assignments with strong diagonal entries (i.e., good label accuracy) in a given error matrix E. It is implicitly parameterized by the number of gold-standard labels used to generate E. Were this number to equal the size of the corpus (an untenable situation), using the resulting CORRECTCOUNT loss function in conjunction with the LP solver would yield the highest possible label accuracy achievable with any solution to the class correspondence problem.

BestAnnotator

Assembling gold-standard labels is cumbersome, and it begs the question which MOMRESP aims to address in the first place. The BESTANNOTATOR loss function instead only uses parameter estimates found in the model. Recall that inference yields estimates of error characteristics γ_j for each annotator j, and each γ_j matrix can be viewed as a normalized confusion matrix. When inferred label classes are permuted, the rows of matrix γ_j are permuted. Suppose there is an annotator $\hat{j}$ whose annotations we believe to be mostly in accord with the truth; i.e., for no class k is she more likely to choose some other class k'. Unfortunately, the model's predicted $\gamma_{\hat{j}}$ is subject to label switching, manifested as permutations of the rows of $\hat{j}$'s true confusion matrix. Then our belief is that with the correct row ordering, $\gamma_{\hat{j}}$ is strongly diagonal. Thus, the intuition underlying the

BESTANNOTATOR loss function is that the overall divergence of a given γ_j from the identity matrix — assuming γ_j's rows are in the proper order — should be small. Consequently, we define this loss function using KL Divergence from the indicated row of the identity matrix I:

$$L_{\text{BA}}(c', c'') = KL(I_{c''} || \gamma_{\hat{j}c'})$$

AggregateAnnotator

Rather than relying on a single annotator, we might try aggregating across annotators. AGGREGATEANN assumes that for no class k are all annotators more likely to choose some other class k'. Thus, we aggregate the error characteristics γ of all annotators. Summed rows would not constitute probability vectors, so rather than normalizing the summed probability vectors and employing KL divergence, this loss function measures the aggregated diagonal entry of the row in its proposed destination c'':

$$L_{\text{AA}}(c', c'') = -\sum_{j=1}^{J} \gamma_{jc'c''}$$

3.4 Experiments

Models dealing with multiple error-prone annotations can be challenging to evaluate in a controlled way because multiply-annotated benchmark datasets for classification have not been established. Sheng et al. [118] simulate annotations for an annotator by corrupting the ground-truth labels according to an accuracy parameter associated with that annotator. They use a strategy they term Generalized Round Robin (GRR) for determining which data instances are annotated and with how many annotations. In GRR, an instance is selected at random (without replacement) to be annotated d separate times by annotators selected randomly with replacement. After all instances have been annotated, the process is repeated. Thus, an instance can be annotated more than d times if revisited. GRR simulation has two parameters of interest: the number d of annotations per instance per round and annotator quality.

	A1	A2	A3	A4	A5
HIGH	90	85	80	75	70
MED	70	65	60	55	50
LOW	50	40	30	20	10
CONFLICT	50†	40†	30†	20†	10†

Table 3.1: Annotator (A1-A5) accuracies for each quality level (HIGH, MED, LOW, CONFLICT). † indicates that errors are systematic (see text for details).

We use GRR with annotators from the quality pools (one named pool per row) in Table 3.1. Each pool lists the accuracy of five annotators, A1-A5 (five is an arbitrary choice for the experiments). In the quality settings HIGH, MED, and LOW, annotator errors are distributed uniformly across the incorrect classes. Because there are no patterns among errors, these settings approximate situations in which annotators are ultimately in agreement about the task they are doing, although some are better at it than others.

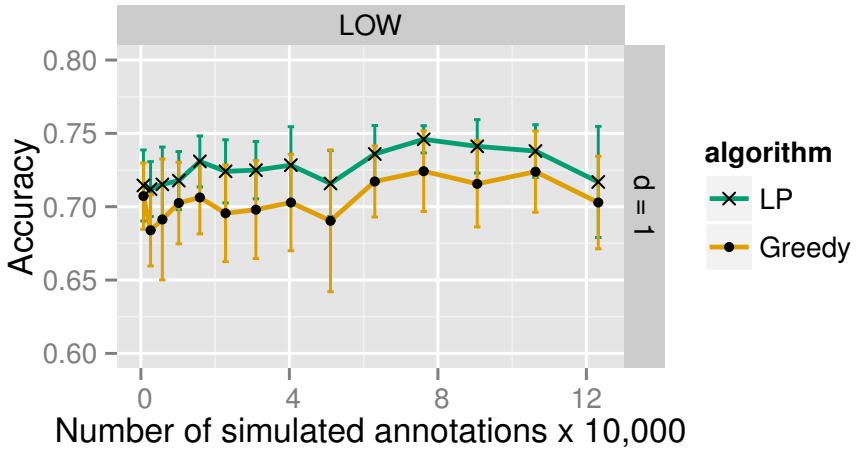


Figure 3.2: Linear Programming (LP) versus Greedy search (Greedy) for class correspondence correction. Both approaches use the same CorrectCount loss function.

The CONFLICT quality setting in Table 3.1 is special in that annotator errors are systematic rather than uniform random. Systematic errors are produced at simulation time by constructing a confusion matrix (similar to 'γ') for each simulated annotator with diagonals set to the desired accuracy, and with off-diagonals sampled from a symmetric Dirichlet distribution with parameter 0.1 for sparsity and then scaled so that each row sums to 1. These draws from a sparse Dirichlet

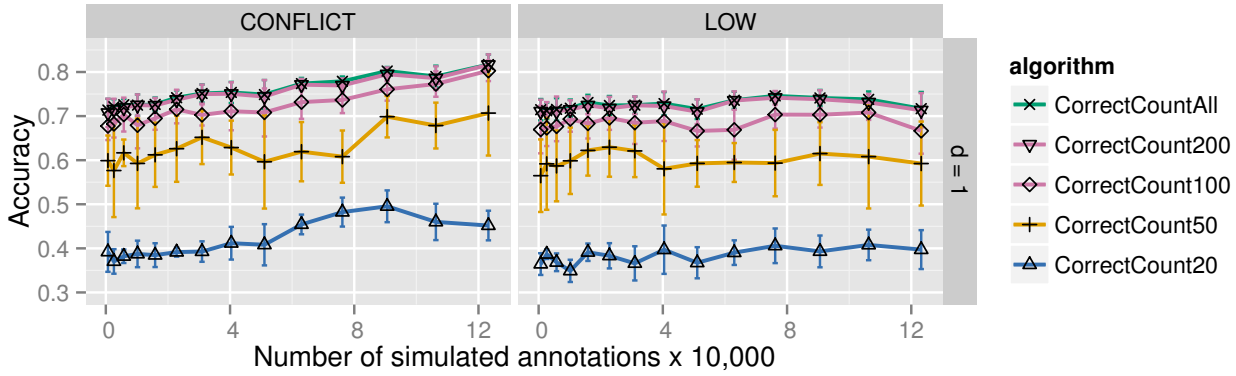


Figure 3.3: Correcting the class correspondence problem using the CORRECTCOUNT loss function with various amounts of manually labeled data to create its error matrix. Notice that performance does not plateau until about 200 items have been manually labeled, or 10 per class.

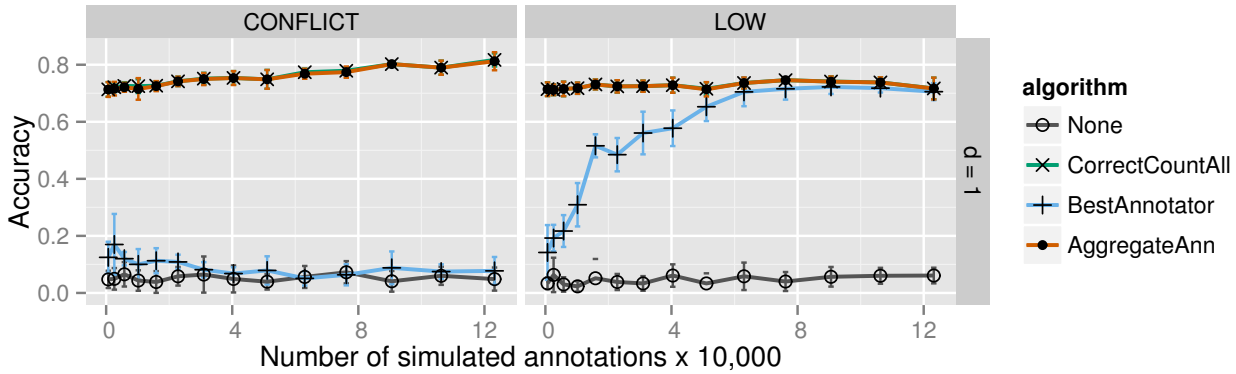


Figure 3.4: Class correspondence correction strategies. NONE takes the raw sampling results in which classes are essentially randomly permuted. CORRECTCOUNT uses information from gold-standard labels for all the data and is an upper bound on what is possible. BESTANNOTATOR uses the γ_j of an arbitrarily selected annotator j. AGGREGATEANN uses the aggregated γ matrices of all annotators, and performs at nearly the level of the upper bound in all settings tested (including those not shown).

yield error patterns that are quite self-consistent. For example, annotator A5 in the CONFLICT setting will label class B as B only 10% of the time, but might label B as C 85% of the time. CONFLICT approximates an annotation project where annotators understand the annotation guidelines differently from one another.

We use GRR to simulate the annotators in Table 3.1 annotating the 20 Newsgroups data set [60], which consists of approximately 20,000 documents evenly divided among 20 classes. For each experiment, we randomly select a subset of 17,000 documents to serve as our corpus. 20 Newsgroups is an appropriate dataset because it is a well-known text classification benchmark and has been used by previous work in evaluating related semi-supervised mixture-of-multinomials models [100]. In all plots, all graphed lines represent at least 20 randomized runs on different simulations (the corpus and annotations change in every simulation), with error bars—too small to see in most cases—representing one standard deviation.

3.4.1 Class Correspondence Correction

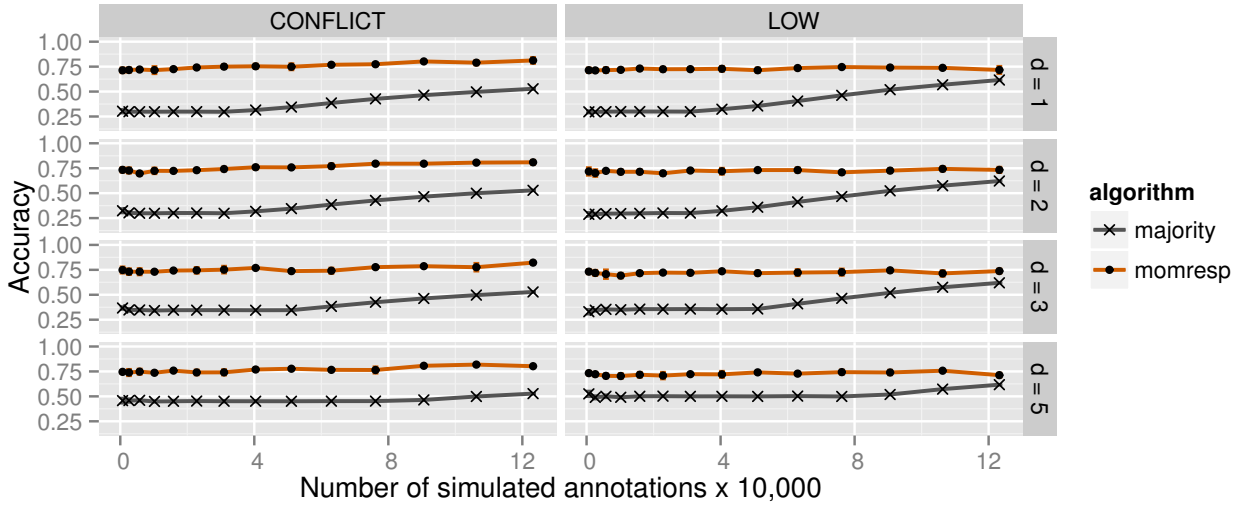


Figure 3.5: Inferred label accuracy on the 20 Newsgroups dataset

In Section 3.3.3 we explained that label switching among predicted label classes could be rectified by defining a loss function over potential label class reassignments and using linear programming to find the label class permutation incurring the smallest possible loss overall. To

confirm that linear programming is worth the time required to implement, Figure 3.2 demonstrates that in practice, using linear programming to find the optimal alignment solution yields gains over a simple greedy solution. Figure 3.2 graphs the model's inferred label accuracy after sampling and *post hoc* class correspondence correction. Greedy and LP both use the CORRECTCOUNT loss function with access to a full confusion matrix, and both are given the same set of sampled model parameters to correct. Greedy assembles a solution by iteratively selecting the source class k corresponding to the error matrix column with the largest value, and then assigning it to the unclaimed destination class k' that will result in the largest value along the diagonal of the error matrix. The optimal linear programming search, although slower in practice, yields answers with a higher mean and less variance than the greedy search.

In Figure 3.3 we explore how much manually labeled data the CORRECTCOUNT loss function requires in order to be effective. CorrectCount20 has access to 20 randomly selected labeled instances, CorrectCount50 has access to 50, and so on. CorrectCountALL has access to the full error matrix and should therefore be regarded as an upper bound on improvements to be gained from realigning inferred label classes. Notice that once approximately 200 gold-standard labels are available then CORRECTCOUNT's performance reaches its peak. That is roughly 10 labels per class, and represents a non-negligible amount of work.

So as not to be dependent on any gold-standard labels, the loss functions BESTANNOTATOR and AGGREGATEANN use only inferred model parameters γ. Figure 3.4 compares their behavior with the upper bound CORRECTCOUNT and 'None,' which indicates the performance of samples whose latent class correspondence has not been corrected. BESTANNOTATOR struggles badly in the CONFLICT setting in which annotators make systematic errors. This is unsurprising, since CONFLICT violates the assumption made by BESTANNOTATOR that an annotator exists who is basically aligned with the truth. BESTANNOTATOR does better in LOW where errors are made uniformly randomly, however it still takes some time to overcome data sparsity. AGGREGATEANN is robust to the systematic errors in CONFLICT because the patterns in the errors are washed out in aggregate. AGGREGATEANN is also more robust to data sparsity since it does not limit itself to

drawing on information from a single individual's annotations. AGGREGATEANN follows the upper bound CORRECTCOUNT so closely that we use it for all subsequent experiments.

Although because of space constraints we have focused on select experimental conditions, the patterns seen in Figures 3.2, 3.3, and 3.4 hold for all other unshown experimental settings, including for the MED and HIGH annotation settings.

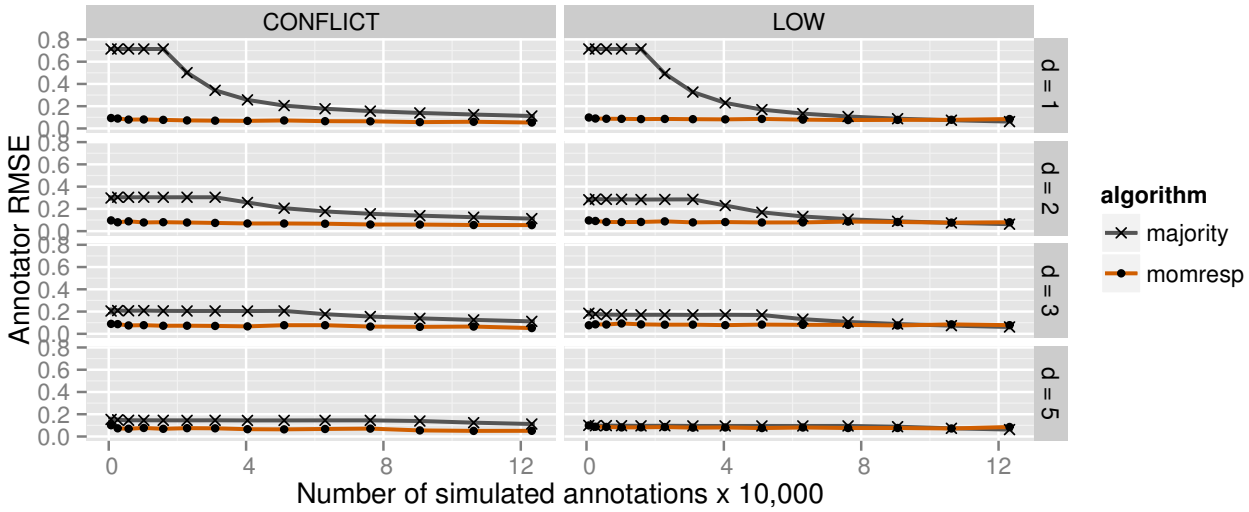


Figure 3.6: Root mean squared error (RMSE) between true and estimated annotator accuracy.

3.4.2 Inferred Label Accuracy

We now compare MOMRESP with majority vote, dubbed MAJORITY. We run simulations that sweep the annotator quality pools from Table 3.1 and annotations $d \in \{1, 2, 3, 5, 10\}$ per instance per round. Accuracy is computed by comparing the withheld gold-standard labels to the label inferred by the model with corrected class correspondences. In order to examine the question of inferred label quality, this section examines accuracy over all items in the dataset with one or more annotations. In Figure 3.5 we chart accuracy as a function of the number of simulated annotations. Because there may be multiple annotations per document, these learning curves extend beyond the length of the corpus. Curves end when every document has roughly 7 annotations.

Majority vote (MAJORITY) suffers from a number of shortcomings. Because it shares no information among instances, accuracy is constant until every item in the dataset has been selected

for annotation at least once (at x-coordinate $17,000 \cdot d$) and instances begin to be re-annotated. Not surprisingly, when annotation quality is high and there are enough annotations per instance, majority vote is sufficient to always select the correct label; e.g., when $d = 10$ and annotator quality is HIGH (not shown). However, when annotations are sparse or of low quality, there is significant room for improvement above the baseline. Also notice that because majority vote implicitly assumes that annotation errors are uniform random, MAJORITY particularly struggles to deal with the systematic errors encountered in CONFLICT.

MOMRESP is superior to MAJORITY in the annotation settings considered in Figures 3.5 and 3.6. MOMRESP allows information from the features (word counts) to be used when selecting a label; in effect, the features get a vote alongside the annotations. Furthermore, class-conditional word probabilities ϕ are shared among both annotated and unannotated instances. Thus, MOMRESP is able to leverage information from all instances when inferring labels for instances with annotations. This gives MOMRESP a tremendous advantage in the early stages of corpus annotation when annotations are too sparse or uncertain to trust, such as where $d = 1$ and annotator quality is LOW.

3.4.3 Annotator Error Estimation

We now measure the effectiveness of the model in learning annotator accuracy. Because we simulated annotator error characteristics, we can compare model predictions with the truth. We compute expected annotator accuracy according to each model and compare those predictions with the simulation parameters used to generate the dataset. These differences are aggregated across all annotators using root mean-squared error. When RMSE is high, estimates of annotator accuracy are poor; when it is low, estimates are good. For MOMRESP, expected annotator accuracy is computed by inspecting the diagonals of the γ matrices. We compare with a baseline approach (MAJORITY) defined to be the percentage of times that each annotator agreed with the majority vote label.

Figure 3.6 shows that a model's ability to accurately learn annotator error characteristics is closely related to its ability to infer correct corpus labels. MOMRESP enjoys an advantage in

settings stages when it can use data evidence to assess the likely quality of sparse annotations. MAJORITY overtakes MOMRESP as redundant annotations accumulate. MAJORITY provides poor estimates when faced with systematic annotator error in CONFLICT.

3.4.4 Failure Cases

The modeling assumptions made in MOMRESP cause it to perform very poorly in settings where annotations are far more informative than the natural data clusters. MOMRESP assumes that documents were generated by selecting a class y and subsequently selecting words x and annotations a based on class y. Accordingly, during inference words and annotations are given equal consideration. However, when annotations are highly accurate and sufficiently abundant, they contain far more information than the average word. For example, in Figure 3.7 annotators are highly accurate, and each document has at least 3 annotations at all times.

To test our hypothesized explanation of MOMRESP's bad behavior in these situations, we tried removing the model's data component, effectively turning it into a Bayesian item-response model. This data-insensitive model is called ITEMRESP in Figure 3.7. Notice that ITEMRESP performs as well as MAJORITY in annotation-rich settings, but lacks the advantages of MOMRESP in annotation-poor settings such as at the beginning of the MED column. These complementary strengths and weaknesses suggest that were a model able to give appropriate weight to the evidence coming from document words, it could enjoy the complementary strengths of MOMRESP and ITEMRESP. Preliminary tests using ad-hoc weighting methods are promising. A better solution will build the data component weighting into the model in a principled way.

3.5 Conclusions and Future Work

We have presented MOMRESP, a model that incorporates information from both natural data clusters as well as annotations from multiple annotators to infer ground-truth estimates for the document classification task. We have demonstrated that MOMRESP can use unlabeled data to improve estimates of the ground-truth labels over a majority vote baseline dramatically in situations

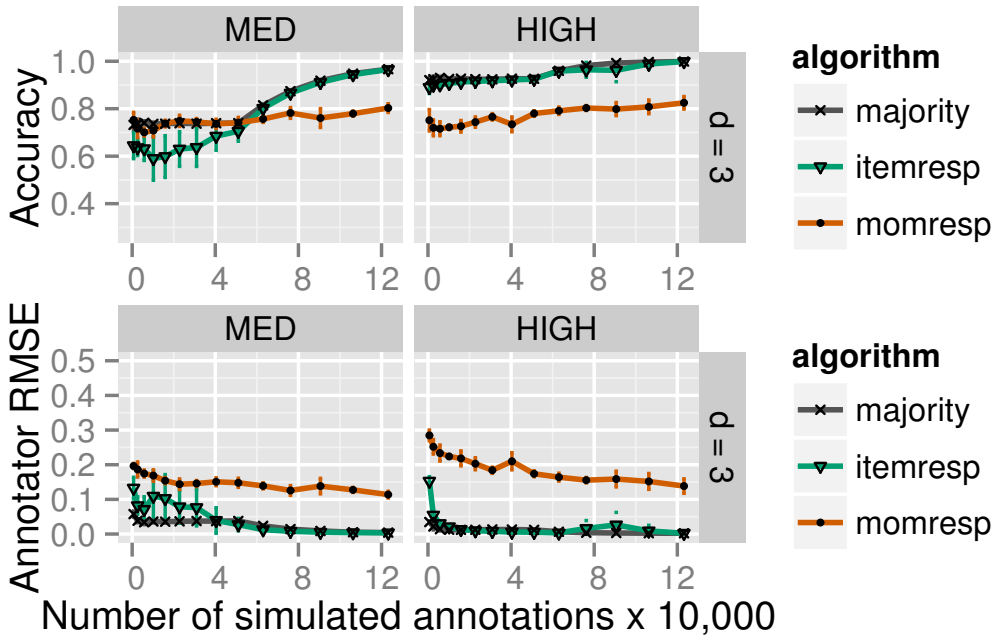


Figure 3.7: Label accuracy (top) and RMSE between true and estimated annotator accuracy (bottom). d=3 indicates that each document is labeled 3 times before moving to the next document, and MED and HIGH indicate that annotations are highly reliable. MOMRESP's data component becomes a liability in these situations.

where both annotations are scarce and annotation quality is low as well as in situations where annotators disagree consistently. Correspondingly, in those same situations, estimates of annotator reliability are also stronger than the baseline. Because MOMRESP predictions are subject to label switching, we identified a solution that found nearly optimal predicted class reassignments in a variety of settings using only information available to the model at inference time. MOMRESP does not perform well in cases where annotations are more informative than natural data clusters. We showed that when the data component is removed from the model—turning it into a Bayesian item-response model—it performs well in annotation-rich settings at the expense of performance in annotation-poor settings. Future work will focus on combining the complementary strengths of MOMRESP and data-ignorant item-response models by refining the structure of the model to weight the information from the model's data component in a principled way.

Acknowledgments

We express our appreciation to the BYU Fulton Supercomputing Lab for computing resources that made this work possible, and also to Kristian Heal, Deryle Lonsdale, and Kevin Black for valuable input and feedback.

Chapter 4

Early Gains Matter: A Case for Preferring Generative over Discriminative Crowdsourcing Models

Published in Proceedings of NAACL 2015 [40]

This chapter introduces an improved training method for the MOMRESP model and challenges a widely-held preference for conditional data modeling in the crowdsourcing literature. It also identifies shortcomings in the MOMRESP model that motivate the improved model in the subsequent chapter.

Abstract

In modern practice, labeling a dataset often involves aggregating annotator judgments obtained from crowdsourcing. State-of-the-art aggregation is performed via inference on probabilistic models, some of which are data-aware, meaning that they leverage features of the data (e.g., words in a document) in addition to annotator judgments. Previous work largely prefers discriminatively trained conditional models. This paper demonstrates that a data-aware crowdsourcing model incorporating a generative multinomial data model enjoys a strong competitive advantage over its discriminative log-linear counterpart in the typical crowdsourcing setting. That is, the generative approach is better except when the annotators are highly accurate in which case simple majority vote is often sufficient. Additionally, we present a novel mean-field variational inference algorithm for the generative model that significantly improves on the previously reported state-of-the-art for that model. We validate our conclusions on six text classification datasets with both human-generated and synthetic annotations.

4.1 Introduction

The success of supervised machine learning has created an urgent need for manually-labeled training datasets. Crowdsourcing allows human label judgments to be obtained rapidly and at relatively low cost. Micro-task markets such as Amazon's Mechanical Turk and CrowdFlower have popularized crowdsourcing by reducing the overhead required to distribute a job to a community of annotators (the "crowd"). However, crowdsourced judgments often suffer from high error rates. A common solution to this problem is to obtain multiple redundant human judgments, or annotations,[1] relying on the observation that, in aggregate, the ability of non-experts often rivals or exceeds that of experts by averaging over individual error patterns [61, 123, 128].

For the purposes of this paper a *crowdsourcing model* is a model that infers, at a minimum, class labels y based on the evidence of one or more imperfect annotations a. A common baseline method aggregates annotations by *majority vote* but by so doing ignores important information. For example, some annotators are more reliable than others, and their judgments ought to be weighted accordingly. State-of-the-art crowdsourcing methods formulate probabilistic models that account for such side information and then apply standard inference techniques to the task of inferring ground truth labels from imperfect annotations.

Data-aware crowdsourcing models additionally account for the features x comprising each data instance (e.g., words in a document). The data can be modeled generatively by proposing a joint distribution $p(y, x, a)$. However, because of the challenge of accurately modeling complex data x, most previous work uses a discriminatively trained conditional model $p(y, a|x)$, hereafter referred to as a discriminative model. As Ng and Jordan [96] explain, maximizing conditional log likelihood is a computationally convenient approximation to minimizing a discriminative 0-1 loss objective, giving rise to the common practice of referring to conditional models as discriminative.

Contributions. This paper challenges the popular preference for discriminative data models in the crowdsourcing literature by demonstrating that in typical crowdsourcing scenarios a generative model enjoys a strong advantage over its discriminative counterpart. We conduct, on both real

[1] We use the term *annotation* to identify human judgments and distinguish them from gold standard class *labels*.

and synthetic annotations, the first empirical comparison of structurally comparable generative and discriminative crowdsourcing models. The comparison is made fair by developing similar mean-field variational inference algorithms for both models. The generative model is considerably improved by our variational algorithm compared with the previously reported state-of-the-art for that model.

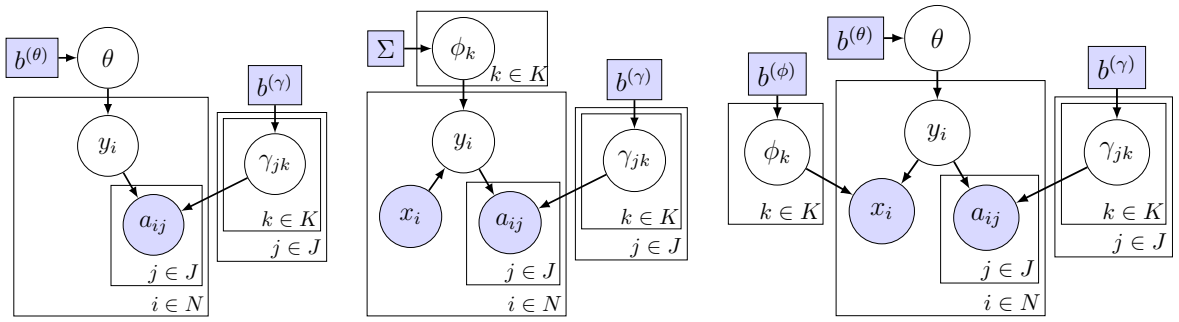
4.2 Previous Work

Dawid and Skene [30] laid the groundwork for modern annotation aggregation by proposing the *item-response* model: a probabilistic crowdsourcing model $p(y, a|\gamma)$ over document labels y and annotations a parameterized by confusion matrices γ for each annotator. A growing body of work extends this model to account for such things as correlation among annotators, annotator trustworthiness, item difficulty, and so forth [9, 57, 104, 106, 122, 135, 136, 142].

Of the crowdsourcing models that are data-aware, most model the data discriminatively [18, 79, 110, 140]. A smaller line of work models the data generatively [69, 120]. We are aware of no papers that compare a generative crowdsourcing model with a similar discriminative model. In the larger context of supervised machine learning, Ng and Jordan [96] observe that generative models parameters tend to converge with fewer training examples than their discriminatively trained counterparts, but to lower asymptotic performance levels. This paper explores those insights in the context of crowdsourcing models.

4.3 Models

At a minimum, a probabilistic crowdsourcing model predicts ground truth labels y from imperfect annotations a (i.e., $\operatorname{argmax}_y p(y|a)$). In this section we review the specifics of two previously-proposed data-aware crowdsourcing models. These models are best understood as extensions to a Bayesian formulation of the item-response model that we will refer to as ITEMRESP. ITEMRESP,



(a) ITEMRESP (b) LOGRESP (c) MOMRESP

Figure 4.1: Directed graphical model depictions of the models discussed in this paper. Round nodes are variables with distributions. Rectangular nodes are hyperparameters (without distributions). Shaded nodes have known values (although some a values may be unobserved).

illustrated in Figure 4.1a, is defined by the joint distribution

$$p(\theta, \gamma, y, a) \tag{4.1}$$
$$= p(\theta)\Big[\prod_{j\in J}\prod_{k\in K} p(\gamma_{jk})\Big]\prod_{i\in N} p(y_i|\theta)\prod_{j\in J} p(a_{ij}|\gamma_j, y_i)$$

where J is the set of annotators, K is the set of class labels, N is the set of data instances in the corpus, θ is a stochastic vector in which θ_k is the probability of label class k, γ_j is a matrix of stochastic vector rows in which $\gamma_{jkk'}$ is the probability that annotator j annotates with k' items whose true label is k, y_i is the class label associated with the ith instance in the corpus, and a_{ijk} is the number of times that instance i was annotated by annotator j with label k. The fact that a_{ij} is a count vector allows for the general case where annotators express their uncertainty over multiple class values. Also, $\theta \sim Dirichlet(b^{(\theta)})$, $\gamma_{jk} \sim Dirichlet(b^{(\gamma)}_{jk})$, $y_i|\theta \sim Categorical(\theta)$, and $a_{ij}|y_i, \gamma_j \sim Multinomial(\gamma_{jy_i}, M_i)$ where M_i is the number of times annotator j annotated instance i. We need not define a distribution over M_i because in practice $M_i = |a_{ij}|_1$ is fixed and known during posterior inference. A special case of this model formulates a_{ij} as a categorical distribution assuming that annotators will provide at most one annotation per item. All hyperparameters are designated b and are disambiguated with a superscript (e.g., the hyperparameters for $p(\theta)$ are $b^{(\theta)}$).

When ITEMRESP parameters are set with uniform θ values and diagonal confusion matrices γ, majority vote is obtained.

Inference in a crowdsourcing model involves a corpus with an annotated portion $N^A = \{i : |a_i|_1 > 0\}$ and also potentially an unannotated portion $N^U = \{i : |a_i|_1 = 0\}$. ITEMRESP can be written as $p(\gamma, y, a) = p(\gamma, y^A, y^U, a)$ where $y^A = \{y_i : i \in N^A\}$ and $y^U = \{y_i : i \in N^U\}$. However, because ITEMRESP has no model of the data x, it receives no benefit from unannotated data N^U.

4.3.1 Log-linear data model (LOGRESP)

One way to make ITEMRESP data-aware is by adding a discriminative log-linear data component [79, 110]. For short, we refer to this model as LOGRESP, illustrated in Figure 4.1b. Concretely,

$$p(\gamma, \phi, y, a|x) = \Big[\prod_{j \in J} \prod_{k \in K} p(\gamma_{jk})\Big] \prod_{k \in K} p(\phi_k) \prod_{i \in N} p(y_i|x_i, \phi) \prod_{j \in J} p(a_{ij}|\gamma_j, y_i) \tag{4.2}$$

where x_{if} is the value of feature f in data instance i (e.g., a word count in a text classification problem), ϕ_{kf} is the probability of feature f occurring in an instance of class k, $\phi_k \sim Normal(0, \Sigma)$, and $y_i|x_i, \phi \sim LogLinear(x_i, \phi)$. That is, $p(y_i|x_i, \phi) = \exp[\phi_{y_i}^T x_i] / \sum_k \exp[\phi_k^T x_i]$.

In the special case that each γ_j is the identity matrix (each annotator is perfectly accurate), LOGRESP reduces to a multinomial logistic regression model. Because it is a conditional model, LOGRESP lacks any built-in capacity for semi-supervised learning.

4.3.2 Multinomial data model (MOMRESP)

An alternative way to make ITEMRESP data-aware is by adding a generative multinomial data component [36, 69]. We refer to the model as MOMRESP, shown in Figure 4.1c.

$$p(\theta,\gamma,\phi,y,x,a) = p(\theta)\Big[\prod_{j\in J}\prod_{k\in K} p(\gamma_{jk})\Big] \tag{4.3}$$
$$\prod_{k\in K} p(\phi_k)\prod_{i\in N} p(y_i|\theta)p(x_i|y_i,\phi)\prod_{j\in J} p(a_{ij}|\gamma_j,y_i)$$

where ϕ_{kf} is the probability of feature f occurring in an instance of class k, $\phi_k \sim Dirichlet(b_k^{(\phi)})$, $x_i \sim Multinomial(\phi_{y_i}, T_i)$, and T_i is a number-of-trials parameter (e.g., for text classification T_i is the number of words in document i). $T_i = |x_i|_1$ is observed during posterior inference $p(\theta,\gamma,\phi,y|x,a)$.

Because MOMRESP is fully generative over the data features x, it naturally performs semi-supervised learning as data from unannotated instances N^U inform inferred class labels y^A of annotated instances via ϕ. This can be seen by observing that $p(x)$ terms prevent terms involving y^U from summing out of the marginal distribution $p(\theta,\gamma,\phi,y^A,x,a) = \sum_{y^U} p(\theta,\gamma,\phi,y^A,y^U,x,a) = p(\theta,\gamma,\phi,y^A,x^A,a)\sum_{y^U} p(y^U|\theta)p(x^U|y^U)$.

When $N = N^U$ (the unsupervised setting) the posterior distribution $p(\theta,\gamma,\phi,y^U|x,a) = p(\theta,\phi,y^U|x)$ is a mixture of multinomials clustering model. Otherwise, the model resembles a semi-supervised naïve Bayes classifier [100]. However, naïve Bayes is supervised by trustworthy labels whereas MOMRESP is supervised by imperfect annotations mediated by inferred annotator error characteristic γ. In the special case that γ is the identity matrix (each annotator is perfectly accurate), MOMRESP reduces to a possibly semi-supervised naïve Bayes classifier where each annotation is a fully trusted label.

4.3.3 A Generative-Discriminative Pair

MOMRESP and LOGRESP are a generative-discriminative pair, meaning that they belong to the same parametric model family but with parameters fit to optimize joint likelihood and conditional likelihood, respectively. This relationship is seen via the equivalence of the conditional probability of LOGRESP $p_L(y, a|x)$ and the same expression according to MOMRESP $p_M(y, a|x)$. For simplicity in this derivation we omit priors and consider ϕ, θ, and γ to be known values. Then

$$p_M(y, a|x) = \frac{p(y)p(x|y)p(a|y)}{\sum_{y'}\sum_{a'} p(y')p(x|y')p(a'|y')} \tag{4.4}$$

$$= \frac{p(y)p(x|y)}{\sum_{y'} p(y')p(x|y')} \cdot p(a|y) \tag{4.5}$$

$$= \frac{\exp[e^{w_y^T x+z}]}{\sum_k \exp[e^{w_k^T x+z}]} \cdot p(a|y) \tag{4.6}$$

$$= p_L(y, a|x) \tag{4.7}$$

Equation 4.4 follows from Bayes Rule and conditional independence in the model. In Equation 4.5 $p(a'|y)$ sums to 1. The first term of Equation 4.6 is the posterior $p(y|x)$ of a naïve Bayes classifier, known to have the same form as a logistic regression classifier where parameters w and z are constructed from ϕ and θ.[2]

4.4 Mean-field Variational Inference (MF)

In this section we present novel mean-field (MF) variational algorithms for LOGRESP and MOMRESP. Note that Liu et al. [79] present (in an appendix) variational inference for LOGRESP based on belief propagation (BP). They do not test their algorithm for LOGRESP; however, their comparison of MF and BP variational inference for the ITEMRESP model indicates that the two flavors of variational inference perform very similarly. Our MF algorithm for LOGRESP has not been designed with the idea of outperforming its BP analogue, but rather with the goal of ensuring that the

[2] `http://cs.cmu.edu/~tom/mlbook/NBayesLogReg.pdf` gives a proof of this property in the continuous case and hints about the discrete case proof.

generative and discriminative model use the same inference algorithm. We expect that we would achieve the same results if our comparison used variational BP algorithms for both MOMRESP and LOGRESP, although such an additional comparison is beyond the scope of this work.

Broadly speaking, variational approaches to posterior inference transform inference into an optimization problem by searching within some family of tractable approximate distributions Q for the distribution $q \in Q$ that minimizes distributional divergence from an intractable target posterior p^*. In particular, under the mean-field assumption we confine our search to distributions Q that are fully factorized.

4.4.1 LOGRESP Inference

We approximate LOGRESP's posterior $p^*(\gamma, \phi, y|x, a)$ using the fully factorized approximation $q(\gamma, \phi, y) = \left[\prod_j \prod_k q(\gamma_{jk})\right] \prod_k q(\phi_k) \prod_i q(y_i)$. Approximate marginal posteriors q are disambiguated by their arguments.

Algorithm. Initialize each $q(y_i)$ to the empirical distribution observed in the annotations a_i. The Kullback-Leibler divergence $KL(q||p^*)$ is minimized by iteratively updating each variational distribution in the model as follows:

$$q(\gamma_{jk}) \propto \prod_{k' \in K} \gamma_{jkk'}^{b_{jkk'}^{(\gamma)} + \sum_{i \in N} a_{ijk'} q(y_i = k) - 1} = Dirichlet(\alpha_{jk}^{(\gamma)})$$

$$q(\phi_k) \propto \exp\left[\phi_k^T \Sigma^{-1} \phi_k + \sum_{i \in N} q(y_i = k) \phi_k^T x_i\right]$$

$$\begin{aligned} q(y_i) &\propto \prod_{k \in K} \exp\Big[\sum_{j \in J} \sum_{k' \in K} a_{ijk'} E_{q(\gamma_{jk})}[\log \gamma_{jkk'}] + \\ &\qquad \sum_{f \in F} x_{if} E_{q(\phi_k)}[\phi_{kf}]\Big]^{\mathbb{1}(y_i = k)} \\ &\propto \prod_{k \in K} \alpha_{ik}^{(y) \mathbb{1}(y_i = k)} = Categorical(\alpha_i^{(y)}) \end{aligned}$$

Approximate distributions are updated by calculating variational parameters $\alpha^{(\cdot)}$, disambiguated by a superscript. Because $q(\gamma_{jk})$ is a Dirichlet distribution the term $E_{q(\gamma_{jk})}[\log \gamma_{jkk'}]$ appearing in $q(y_i)$ is computed analytically as $\psi(\alpha^{(\gamma)}_{jkk'}) - \psi(\sum_{k'} \alpha^{(\gamma)}_{jkk'})$ where ψ is the digamma function.

The distribution $q(\phi_k)$ is a logistic normal distribution. This means that the expectations $E_{q(\phi_k)}[\phi_{kf}]$ that appear in $q(y_i)$ cannot be computed analytically. Following Liu et al. [79], we approximate the distribution $q(\phi_k)$ with the point estimate $\hat{\phi}_k = \text{argmax}_{\phi_k} q(\phi_k)$ which can be calculated using existing numerical optimization methods for log-linear models. Such maximization can be understood as embedding the variational algorithm inside of an outer EM loop such as might be used to tune hyperparameters in an empirical Bayesian approach (where ϕ are treated as hyperparameters).

4.4.2 MOMRESP Inference

MOMRESP's posterior $p^*(y, \theta, \gamma, \phi|x, a)$ is approximated with the fully factorized distribution $q(y, \theta, \gamma, \phi) = q(\theta)\left[\prod_j \prod_k q(\gamma_{jk})\right] \prod_k q(\phi_k) \prod_i q(y_i)$.

Algorithm. Initialize each $q(y_i)$ to the empirical distribution observed in the annotations a_i. The Kullback-Leibler divergence $KL(q||p^*)$ is minimized by iteratively updating each variational distribution in the model as follows:

$$q(\theta) \propto \prod_{k \in K} \theta_k^{b_k^{(\theta)} + \sum_{i \in N} q(y_i = k) - 1} = Dirichlet(\alpha^{(\theta)})$$

$$q(\gamma_{jk}) \propto \prod_{k' \in K} \gamma_{jkk'}^{b_{jkk'}^{(\gamma)} + \sum_{i \in N} a_{ijk'} q(y_i = k) - 1} = Dirichlet(\alpha_{jk}^{(\gamma)})$$

$$q(\phi_k) \propto \prod_{f \in F} \phi_{kf}^{b_{kf}^{(\phi)} + \sum_{i \in N} x_{if} q(y_i = k) - 1} = Dirichlet(\alpha_k^{(\phi)})$$

$$q(y_i) \propto \prod_{k \in K} \exp\Big[\sum_{j \in J} \sum_{k' \in K} a_{ijk'} E_{q(\gamma_{jk})}[\log \gamma_{jkk'}] + E_{q(\theta_k)}[\log \theta_k] + \sum_{f \in F} x_{if} E_{q(\phi_k)}[\log \phi_{kf}]\Big]^{\mathbb{1}(y_i=k)}$$
$$\propto \prod_{k \in K} \alpha_{ik}^{(y)\mathbb{1}(y_i=k)} = Categorical(\alpha_i^{(y)})$$

Approximate distributions are updated by calculating the values of variational parameters $\alpha^{(\cdot)}$, disambiguated by a superscript. The expectations of log terms in the $q(y_i)$ update are all with respect to Dirichlet distributions and so can be computed analytically as explained previously.

4.4.3 Model priors and implementation details

Computing a lower bound on the log likelihood shows that in practice the variational algorithms presented above converge after only a dozen or so updates. We compute $\text{argmax}_{\phi_k}\, q(\phi_k)$ for LOGRESP using the L-BFGS algorithm as implemented in MALLET [85]. We choose uninformed priors $b_k^{(\theta)} = 1$ for MOMRESP and identity matrix $\Sigma = \mathbb{1}$ for LOGRESP. We set $b_{kf}^{(\phi)} = 0.1$ for MOMRESP to encourage sparsity in per-class word distributions. Liu et al. [79] argue that a uniform prior over the entries of each confusion matrix γ_j can lead to degenerate performance. Accordingly, we set the diagonal entries of each $b_j^{(\gamma)}$ to a higher value $b_{jkk}^{(\gamma)} = \frac{1+\delta}{K+\delta}$ and off-diagonal entries to a lower value $b_{jkk'}^{(\gamma)} = \frac{1}{K+\delta}$ with $\delta = 2$.

Both MOMRESP and LOGRESP are given full access to all instances in the dataset, annotated and unannotated. However, as explained in Section 4.3.1, LOGRESP is conditioned on the data and thus is structurally unable to make use of unannotated data. We experimented briefly with self-training for LOGRESP but it had little effect. With additional effort one could likely settle on a heuristic scheme that allowed LOGRESP to benefit from unannotated data. However, since such an extension is external to the model itself, it is beyond the scope of this work.

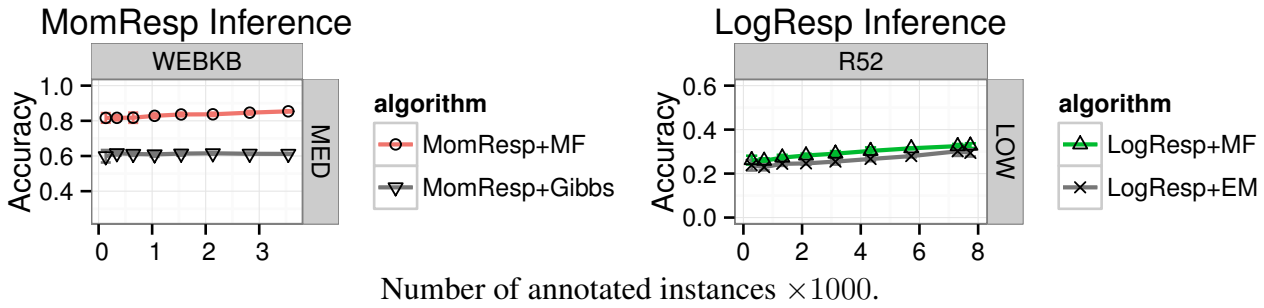


Figure 4.2: Mean field (MF) variational inference outperforms previous inference methods for both models. *Left*: MOMRESP with MF (MOMRESP+MF) versus with Gibbs sampling (MOMRESP+Gibbs) on the WebKB dataset using annotators from the MED pool. *Right*: LOGRESP with MF (LOGRESP+MF) versus with EM (LOGRESP+EM) on the Reuters52 dataset using annotators from the LOW pool.

4.5 Experiments with Simulated Annotators

Models which learn from error-prone annotations can be challenging to evaluate in a systematic way. Simulated annotations allow us to systematically control annotator behavior and measure the performance of our models in each configuration.

4.5.1 Simulating Annotators

We simulate an annotator by corrupting ground truth labels according to that annotator's accuracy parameters. Simulated annotators are drawn from the annotator quality pools listed in Table 4.1. Each row is a named pool and contains five annotators A1–A5, each with a corresponding accuracy parameter (the number five is chosen arbitrarily). In the pools HIGH, MED, and LOW, annotator errors are distributed uniformly across the incorrect classes. Because there are no patterns among errors, these settings approximate situations in which annotators are ultimately in agreement about the task they are doing, although some are better at it than others. The HIGH pool represents a corpus annotation project with high quality annotators. In the MED and LOW pools annotators are progressively less reliable.

The CONFLICT annotator pool in Table 4.1 is special in that annotator errors are made systematically rather than uniformly. Systematic errors are produced at simulation time by construct-

ing a per-annotator confusion matrix (similar to γ_j) whose diagonal is set to the desired accuracy setting, and whose off-diagonal row entries are sampled from a symmetric Dirichlet distribution with parameter 0.1 to encourage sparsity and then scaled so that each row properly sums to 1. These draws from a sparse Dirichlet yield consistent error patterns. The CONFLICT pool approximates an annotation project where annotators understand the annotation guidelines differently from one another. For the sake of example, annotator A5 in the CONFLICT setting will annotate documents with the true class B as B exactly 10% of the time but might annotate B as C 85% of the time. On the other hand, annotator A4 might annotate B as D most of the time. We choose low agreement rates for CONFLICT to highlight a case that violates majority vote's assumption that annotators are basically in agreement.

	A1	A2	A3	A4	A5
HIGH	90	85	80	75	70
MED	70	65	60	55	50
LOW	50	40	30	20	10
CONFLICT	50†	40†	30†	20†	10†

Table 4.1: For each simulated annotator quality pool (HIGH, MED, LOW, CONFLICT), annotators A1-A5 are assigned an accuracy. † indicates that errors are systematically in conflict as described in the text.

4.5.2 Datasets and Features

We simulate the annotator pools from Table 4.1 on each of six text classification datasets. The datasets 20 Newsgroups, WebKB, Cade12, Reuters8, and Reuters52 are described by Cardoso-Cachopo [14]. The LDC-labeled Enron emails dataset is described by Berry et al. [4]. Each dataset is preprocessed via Porter stemming and by removal of the stopwords from MALLET's stopword list. Features occurring fewer than 5 times in the corpus are discarded. Features are fractionally scaled so that $|x_i|_1$ is equal to the average document length since document scaling has been shown to be beneficial for multinomial document models [100].

Each dataset is annotated according to the following process: an instance is selected at random (without replacement) and annotated by three annotators selected at random (without replacement). Because annotation simulation is a stochastic process, each simulation is repeated five times.

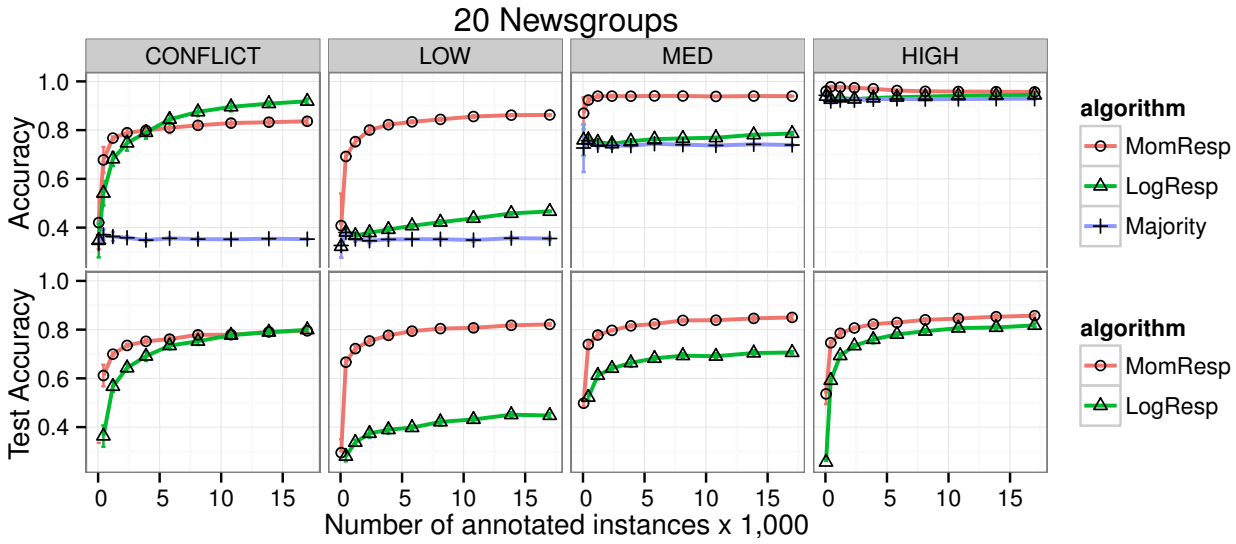


Figure 4.3: *Top row*: Inferred label accuracy on three-deep annotations. A majority vote baseline is shown for reference. *Bottom row*: Generalization accuracy on a test set. Majority vote is not shown since it does not generate test set predictions. Each column uses the indicated simulated annotator pool.

4.5.3 Validating Mean-field Variational Inference

Figure 4.2 compares mean-field variational inference (MF) with alternative inference algorithms from previous work. For variety, the left and right plots are calculated over arbitrarily chosen datasets and annotator pools, but these trends are representative of other settings. MOMRESP using MF is compared with MOMRESP using Gibbs sampling estimating $p(y|x, a)$ from several hundred samples (an improvement to the method used by Felt et al. [36]). MOMRESP benefits significantly from MF. We suspect that this disparity could be reduced via hyperparameter optimization as indicated by Asuncion et al. [1]. However, that investigation is beyond the scope of the current work. LOGRESP using MF is compared with LOGRESP using expectation maximization (EM) as in [110].

LOGRESP with MF displays minor improvements over LOGRESP with EM. This is consistent with the modest gains that Liu et al. [79] reported when comparing variational and EM inference for the ITEMRESP model.

4.5.4 Discriminative (LOGRESP) versus Generative (MOMRESP)

We run MOMRESP and LOGRESP with MF inference on the cross product of datasets and annotator pools. Inferred label accuracy on items that have been annotated is the primary task of crowdsourcing; we track this measure accordingly. However, the ability of these models to generalize on unannotated data is also of interest and allows better comparison with traditional non-crowdsourcing models. Figure 4.3 plots learning curves for each annotator pool on the 20 Newsgroups dataset; results on other datasets are summarized in Table 4.2. The first row of Figure 4.3 plots the accuracy of labels inferred from annotations. The second row of Figure 4.3 plots generalization accuracy using the inferred model parameters ϕ (and θ in the case of MOMRESP) on held-out test sets with no annotations. The generalization accuracy curves of MOMRESP and LOGRESP may be compared with those of naïve Bayes and logistic regression, respectively. Recall that in the special case where annotations are both flawless and trusted (via diagonal confusion matrices γ) then MOMRESP and LOGRESP simplify to semi-supervised naïve Bayes and logistic regression classifiers, respectively.

Notice that MOMRESP climbs more steeply than LOGRESP in all cases. This observation is in keeping with previous work in supervised learning. Ng and Jordan [96] argue that generative and discriminative models have complementary strengths: generative models tend to have steeper learning curves and converge in terms of parameter values after only $\log n$ training examples, whereas discriminative models tend to achieve higher asymptotic levels but converge more slowly after n training examples. The second row of Figure 4.3 shows that even after training on three-deep annotations over the entire 20 newsgroups dataset, LOGRESP's data model does not approach its asymptotic level of performance. The early steep slope of the generative model is more desirable in this setting than the eventually superior performance of the discriminative model given large numbers of annotations. Figure 4.4 additionally plots MOMRESPA, a variant of MOMRESP deprived

of all unannotated documents, showing that the early generative advantage is not attributable entirely to semi-supervision.

The generative model is more robust to annotation noise than the discriminative model, seen by comparing the LOW, MED, and HIGH columns in Figure 4.3. This robustness is significant because crowdsourcing tends to yield noisy annotations, making the LOW and MED annotator pools of greatest practical interest. This assertion is borne out by an experiment with CrowdFlower, reported in Section 4.6.

To validate that LOGRESP does, indeed, asymptotically surpass MOMRESP we ran inference on datasets with increasing annotation depths. Crossover does not occur until 20 Newsgroups is annotated nearly 12-deep for LOW, 5-deep for MED, and 3.5-deep (on average) for HIGH. Additionally, for each combination of dataset and annotator pool except those involving CONFLICT, by the time LOGRESP surpasses MOMRESP, the majority vote baseline is extremely competitive with LOGRESP. The CONFLICT setting is the exception to this rule: CONFLICT annotators are particularly challenging for majority vote since they violate the implicit assumption that annotators are basically aligned with the truth. The CONFLICT setting is of practical interest only when annotators have dramatic deep-seated differences of opinion about what various labels should mean. For most crowdsourcing projects this issue may be avoided with sufficient up-front orientation of the annotators. For reference, in Figure 4.4 we show that a less extreme variant of CONFLICT behaves more similarly to LOW.

Table 4.2 reports the percent of the dataset that must be annotated three-deep before LOGRESP's inferred label accuracy surpasses that of MOMRESP. Crossover tends to happen later when annotation quality is low and earlier when annotator quality is high. Cases reported as *NA* were too close to call; that is, the dominating algorithm changed depending on the random run.

Unsurprisingly, MOMRESP is not well suited to all classification datasets. The 0% entries in Table 4.2 mean that LOGRESP dominates the learning curve for that annotator pool and dataset. These cases are likely the result of the MOMRESP model making the same strict inter-feature independence assumptions as naïve Bayes, rendering it tractable and effective for many classification

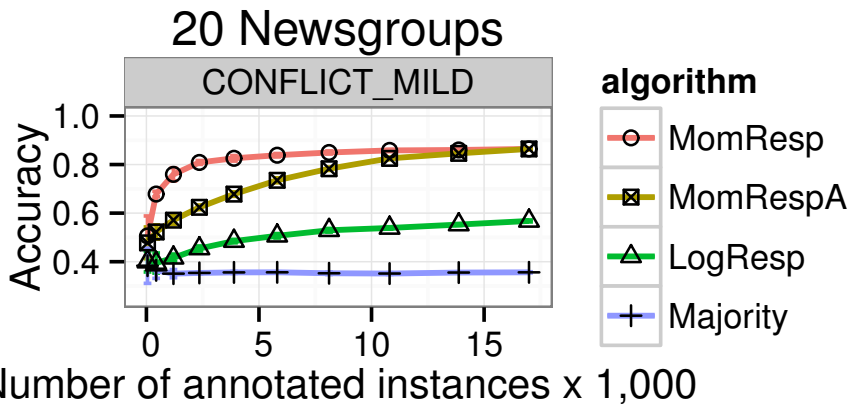


Figure 4.4: Inferred label accuracy for a variant of the CONFLICT annotator pool in which the off-diagonals of each annotator confusion matrix are drawn from a Dirichlet parameterized by 1 rather than 0.1. Also adds the algorithm MOMRESPA to show the effect of removing MOMRESP's access to unannotated documents.

tasks but ill-suited for datasets where features are highly correlated or for tasks in which class identity is not informed by document vocabulary. The CADE12 dataset, in particular, is known to be challenging. A supervised naïve Bayes classifier achieves only 57% accuracy on this dataset [14]. We would expect MOMRESP to perform similarly poorly on sentiment classification data. Although we assert that generative models are inherently better suited to crowdsourcing than discriminative models, a sufficiently strong mismatch between model assumptions and data can negate this advantage.

	CONFLICT	LOW	MED	HIGH
20 News	21%	✓	✓	✓
WebKB	*NA*	✓	✓	0%
Reuters8	*NA*	✓	✓	✓
Reuters52	✓	✓	✓	✓
CADE12	0%	✓	0%	0%
Enron	✓	✓	✓	18%

Table 4.2: The percentage of the dataset that must be annotated (three-deep) before the generative model MOMRESP is surpassed by LOGRESP. ✓indicates that MOMRESP dominates the entire learning curve; 0% indicates that LOGRESP dominates. *NA* indicates high variance cases that were too close to call.

4.6 Experiments with Human Annotators

In the previous section we used simulations to control annotator error. In this section we relax that control. To assess the effect of real-world annotation error on MOMRESP and LOGRESP, we selected 1000 instances at random from 20 Newsgroups and paid annotators on CrowdFlower to annotate them with the 20 Newsgroups categories, presented as human-readable names (e.g., "Atheism" for alt.atheism). Annotators were allowed to express uncertainty by selecting up to three unique categories per document. During the course of a single day we gathered 7,265 annotations, with each document having a minimum of 3 and a mean of 7.3 annotations.[3] Figure 4.5 shows learning curves for the CrowdFlower annotations. The trends observed previously are unchanged. MOMRESP enjoys a significant advantage when relatively few annotations are available. Presumably LOGRESP would still dominate if we were able to explore later portions of the curve or curves with greater annotation depth.

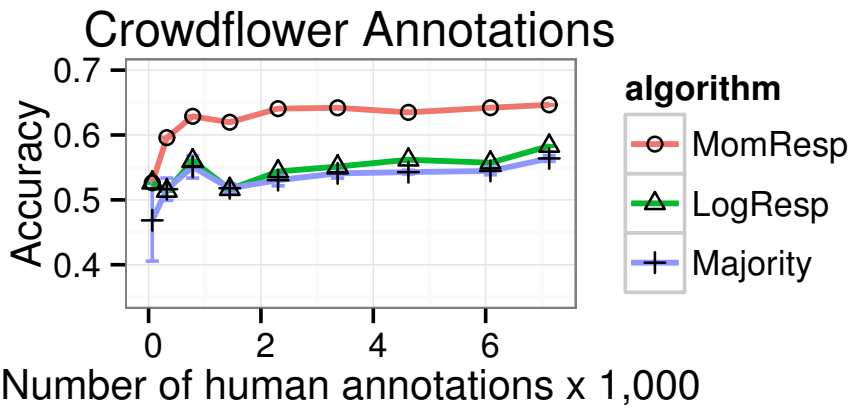


Figure 4.5: Inferred label accuracy on annotations gathered from CrowdFlower over a subset of 1000 instances of the 20 Newsgroups dataset. At the last plotted point there are $7,265/1,000 \approx 7.3$ annotations per instance.

[3] This dataset and the scripts that produced it are available via git at git://nlp.cs.byu.edu/plf1/crowdflower-newsgroups.git

4.7 Conclusions and Future Work

We have argued that generative models are better suited than discriminative models to the task of annotation aggregation since they tend to be more robust to annotation noise and to approach their asymptotic performance levels with fewer annotations. Also, in settings where a discriminative model would usually shine, there are often enough annotations that a simple baseline of majority vote is sufficient.

In support of this argument, we developed comparable mean-field variational inference for a generative-discriminative pair of crowdsourcing models and compared them on both crowdsourced and synthetic annotations on six text classification datasets. In practice we found that on classification tasks for which generative models of the data work reasonably well, the generative model greatly outperforms its discriminative log-linear counterpart.

The generative multinomial model we employed makes inter-feature independence assumptions ill suited to some classification tasks. Document topic models [7] could be used as the basis of a more sophisticated generative crowdsourcing model. One might also transform the data to make it more amenable to a simple model using documents assembled from distributed word representations [88]. Finally, although we expect these results to generalize, we have only experimented with text classification. Similar experiments could be performed on other commonly crowdsourced tasks such as sequence labeling.

Acknowledgments

We thank Alex Smola and the anonymous reviewers for their insightful comments. This work was supported by the collaborative NSF Grant IIS-1409739 (BYU) and IIS-1409287 (UMD).

Chapter 5

Making the Most of Crowdsourced Document Annotations: Confused Supervised LDA

Published in Proceedings of CoNLL 2015 [39]

Best paper award

This chapter builds on the previous chapter by proposing a novel generative crowdsourcing model. This model overcomes many of the deficiencies associated with MOMRESP and achieves state-of-the-art performance on a number of annotation aggregation datasets.

Abstract

Corpus labeling projects frequently use low-cost workers from microtask marketplaces; however, these workers are often inexperienced or have misaligned incentives. Crowdsourcing models must be robust to the resulting systematic and non-systematic inaccuracies. We introduce a novel crowdsourcing model that adapts the discrete supervised topic model sLDA to handle multiple corrupt, usually conflicting (hence "confused") supervision signals. Our model achieves significant gains over previous work in the accuracy of deduced ground truth.

5.1 Modeling Annotators and Abilities

Supervised machine learning requires labeled training corpora, historically produced by laborious and costly annotation projects. Microtask markets such as Amazon's Mechanical Turk and CrowdFlower have turned crowd labor into a commodity that can be purchased with relatively little overhead. However, crowdsourced judgments can suffer from high error rates. A common

solution to this problem is to obtain multiple redundant human judgments, or annotations,[1] relying on the observation that, in aggregate, non-experts often rival or exceed experts by averaging over individual error patterns [61, 123, 128].

A *crowdsourcing model* harnesses the wisdom of the crowd and infers labels based on the evidence of the available annotations, imperfect though they be. A common baseline crowdsourcing method aggregates annotations by *majority vote*, but this approach ignores important information. For example, some annotators are more reliable than others and their judgments ought to be up-weighted accordingly. State-of-the-art crowdsourcing methods account for annotator expertise, often through a probabilistic formalism. Compounding the challenge, assessing unobserved annotator expertise is tangled with estimating ground truth from imperfect annotations; thus, joint inference of these interrelated quantities is necessary. State-of-the-art models also take the data into account, because data features can help ratify or veto human annotators.

We introduce a model that improves on state-of-the-art crowdsourcing algorithms by modeling not only the annotations but also the features of the data (e.g., words in a document). Section 5.2 identifies modeling deficiencies affecting previous work and proposes a solution based on topic modeling; Section 5.2.4 presents inference for the new model. Experiments that contrast the proposed model with select previous work on several text classification datasets are presented in Section 5.3. In Section 5.4 we highlight additional related work.

5.2 Latent Representations that Reflect Labels and Confusion

Most crowdsourcing models extend the item-response model of Dawid and Skene [30]. The Bayesian version of this model, referred to here as ITEMRESP, is depicted in Figure 5.1. In the generative story for this model, a confusion matrix γ_j is drawn for each human annotator j. Each row γ_{jc} of the confusion matrix γ_j is drawn from a symmetric Dirichlet distribution $Dir(b_{jc}^{(\gamma)})$ and encodes a categorical probability distribution over label classes that annotator j is apt to choose when presented with a document whose true label is c. Then for each document d an unobserved

[1] In this paper, we call human judgments *annotations* to distinguish them from gold standard class *labels*.

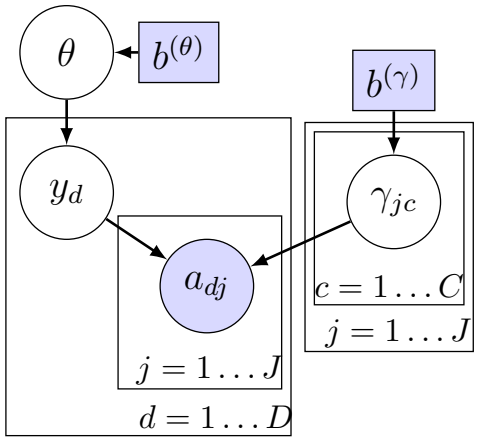


Figure 5.1: **ITEMRESP** as a plate diagram. Round nodes are random variables. Rectangular nodes are free parameters. Shaded nodes are observed. D, J, C are the number of documents, annotators, and classes, respectively.

document label y_d is drawn. Annotations are generated as annotator j corrupts the true label y_d according to the categorical distribution $Cat(\gamma_{jy_d})$.

5.2.1 Leveraging Data

Some extensions to ITEMRESP model the features of the data (e.g., words in a document). Many data-aware crowdsourcing models condition the labels on the data [59, 79, 110, 140], possibly because discriminative classifiers dominate supervised machine learning. Others model the data generatively [9, 36, 69, 120]. Felt et al. [40] argue that generative models are better suited than conditional models to crowdsourcing scenarios because generative models often learn faster than their conditional counterparts [96]—especially early in the learning curve. This advantage is amplified by the annotation noise typical of crowdsourcing scenarios.

Extensions to ITEMRESP that model document features generatively tend to share a common high-level architecture. After the document class label y_d is drawn for each document d, features are drawn from class-conditional distributions. Felt et al. [40] identify the MOMRESP model, reproduced in Figure 5.2, as a strong representative of generative crowdsourcing models. In MOMRESP, the feature vector x_d for document d is drawn from the multinomial distribution with parameter vector ϕ_{y_d}. This class-conditional multinomial model of the data inherits many of the

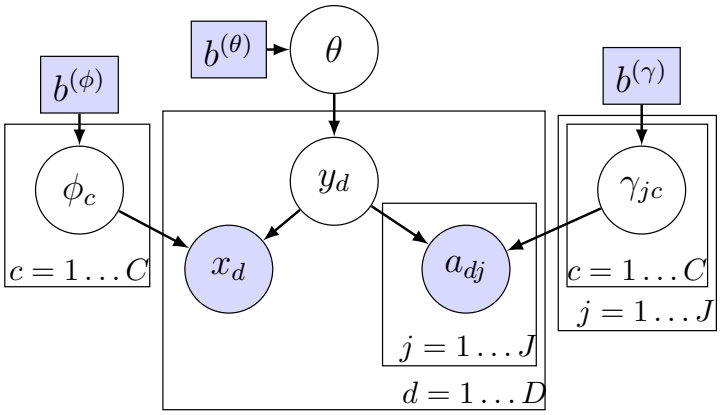


Figure 5.2: **MOMRESP** as a plate diagram. $|x_d| = V$, the size of the vocabulary. Documents with similar feature vectors x tend to share a common label y. Reduces to mixture-of-multinomials clustering when no annotations a are observed.

strengths and weaknesses of the naïve Bayes model that it resembles. Strengths include easy inference and a strong inductive bias which helps the model be robust to annotation noise and scarcity. Weaknesses include overly strict conditional independence assumptions among features, leading to overconfidence in the document model and thereby causing the model to overweight feature evidence and underweight annotation evidence. This imbalance can result in degraded performance in the presence of high quality or many annotations.

5.2.2 Confused Supervised LDA (CSLDA)

We solve the problem of imbalanced feature and annotation evidence observed in MOMRESP by replacing the class-conditional structure of previous generative crowdsourcing models with a richer generative story where documents are drawn first and class labels y_d are obtained afterwards via a log-linear mapping. This move towards conditioning classes on document content is sensible because in many situations document content is authored first, whereas label structure is not imposed until afterwards. It is plausible to assume that there will exist some mapping from a latent document structure to the desired document label distinctions. Moreover, by jointly modeling topics and the mapping to labels, we can learn the latent document representations that best explain how best to predict and correct annotator errors.

Term	Definition
N_d	Size of document d
N_{dt}	$\sum_n \mathbb{1}(z_{dn} = t)$
N_t	$\sum_{d,n} \mathbb{1}(z_{dn} = t)$
$N_{jcc'}$	$\sum_d a_{djc'} \mathbb{1}(y_d = c)$
N_{jc}	$\langle N_{jc1} \cdots N_{jcC} \rangle$
N_{vt}	$\sum_{dn} \mathbb{1}(w_{dn} = v \wedge z_{dn} = t)$
N_t	$\sum_{dn} \mathbb{1}(z_{dn} = t)$
$\hat{N}$	Count excludes variable being sampled
$\bar{z}_d$	Vector where $\bar{z}_{dt} = \frac{1}{N_d} \sum_n \mathbb{1}(z_{dn} = t)$
$\hat{\bar{z}}_d$	Excludes the z_{dn} being sampled

Table 5.1: Definition of counts and select notation. $\mathbb{1}(\cdot)$ is the indicator function.

We call our model **confused supervised LDA** (**CSLDA**, Figure 5.3), based on supervised topic modeling. Latent Dirichlet Allocation [8, LDA] models text documents as admixtures of word distributions, or topics. Although pre-calculated LDA topics as features can inform a crowdsourcing model [73], supervised LDA (sLDA) provides a principled way of incorporating document class labels and topics into a single model, allowing topic variables and response variables to co-inform one another in joint inference. For example, when sLDA is given movie reviews labeled with sentiment, inferred topics cluster around sentiment-heavy words [84], which may be quite different from the topics inferred by unsupervised LDA. One way to view CSLDA is as a discrete sLDA in settings with noisy supervision from multiple, imprecise annotators.

The generative story for CSLDA is:

1. Draw per-topic word distributions ϕ_t from $Dir(b^{(\theta)})$.
2. Draw per-class regression parameters η_c from $Gauss(\mu, \Sigma)$.
3. Draw per-annotator confusion matrices γ_j with row γ_{jc} drawn from $Dir(b^{(\gamma)}_{jc})$.
4. For each document d,
   (a) Draw topic vector θ_d from $Dir(b^{(\theta)})$.
   (b) For each token position n, draw topic z_{dn} from $Cat(\theta_d)$ and word w_{dn} from $Cat(\phi_{z_{dn}})$.

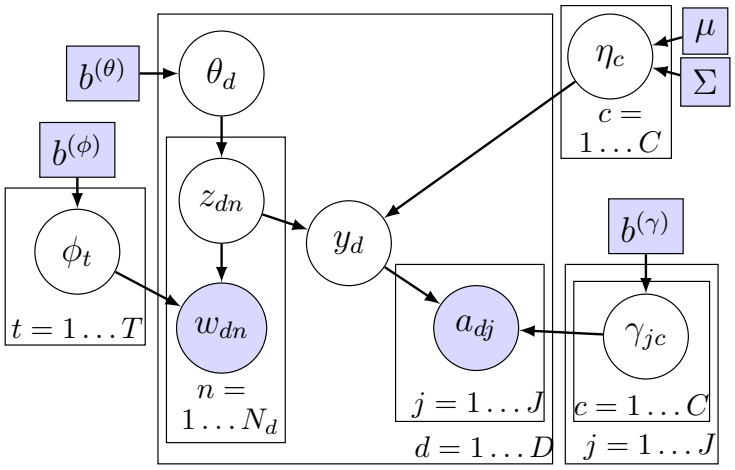


Figure 5.3: **csLDA** as a plate diagram. D, J, C, T are the number of documents, annotators, classes, and topics, respectively. N_d is the size of document d. $|\phi_t| = V$, the size of the vocabulary. η_c is a vector of regression parameters. Reduces to LDA when no annotations a are observed.

(c) Draw class label y_d with probability proportional to $\exp[\eta_{y_d}^\intercal \bar{z}_d]$.

(d) For each annotator j draw annotation vector a_{dj} from γ_{jy_d}.

5.2.3 Stochastic EM

We use stochastic expectation maximization (EM) for posterior inference in csLDA, alternating between sampling values for topics z and document class labels y (the E-step) and optimizing values of regression parameters η (the M-step). To sample z and y efficiently, we derive the full conditional distributions of z and y in a collapsed model where θ, ϕ, and γ have been analytically integrated out. Omitting multiplicative constants, the collapsed model joint probability is

$$
\begin{aligned}
p(z, w, y, a|\eta) &= p(z)p(w|z)p(y|z,\eta)p(a|y) \qquad (5.1)\\
&\propto M(a)\cdot\left(\prod_d B(N_d+b^{(\theta)})\right)\cdot\left(\prod_t B(N_t+b_t^{(\phi)})\right)\\
&\cdot\left(\prod_d \frac{\exp(\eta_{y_d}^\intercal \bar{z}_d)}{\sum_c \exp(\eta_c^\intercal \bar{z}_d)}\right)\cdot\left(\prod_j\prod_c B(N_{jc}+b_{jc}^{(\gamma)})\right)
\end{aligned}
$$

where $B(\cdot)$ is the Beta function (multivariate as necessary), counts N and related symbols are defined in Table 5.1, and $M(a) = \prod_{d,j} M(a_{dj})$ where $M(a_{dj})$ is the multinomial coefficient.

Simplifying Equation 5.1 yields full conditionals for each word z_{dn},

$$p(z_{dn} = t|\hat{z}, w, y, a, \eta) \propto \left(\hat{N}_{dt} + b_t^{(\theta)}\right) \cdot \frac{\hat{N}_{w_{dn}t} + b_{w_{dn}}^{(\phi)}}{\hat{N}_t + |b^{(\phi)}|_1} \cdot \frac{\exp(\frac{\eta_{y_d t}}{N_d})}{\sum_c \exp(\frac{\eta_{ct}}{N_d} + \eta_c^\intercal \hat{\bar{z}}_d)}, \tag{5.2}$$

and similarly for document label y_d:

$$p(y_d = c|z, w, y, a, \eta) \propto \frac{\exp(\eta_c^\intercal \bar{z}_d)}{\sum_{c'} \exp(\eta_{c'}^\intercal \bar{z}_d)} \cdot \prod_j \frac{\prod_{c'} \left(\hat{N}_{jcc'} + b^{(\gamma)}\right)^{\overline{a_{djc'}}}}{\left(\sum_{c'} \hat{N}_{jcc'} + b_{jcc'}^{(\gamma)}\right)^{\overline{\sum_{c'} a_{djc'}}}}, \tag{5.3}$$

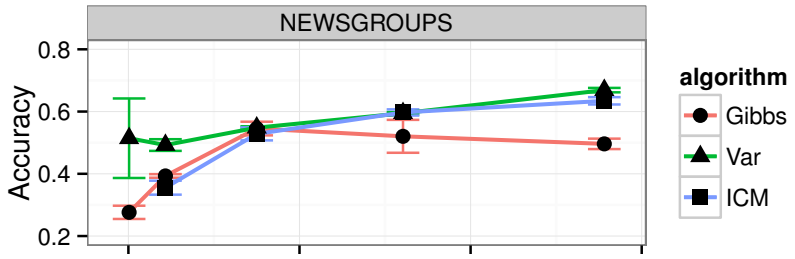
where $x^{\overline{k}} \triangleq x(x+1)\cdots(x+k-1)$ is the rising factorial. In Equation 5.2 the first and third terms are independent of word n and can be cached at the document level for efficiency.

For the M-step, we train the regression parameters η (containing one vector per class) by optimizing the same objective function as for training a logistic regression classifier, assuming that class y is given:

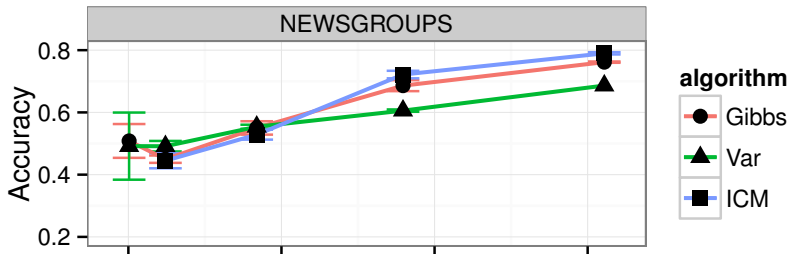
$$p(y = c|z, \eta) = \prod_d \frac{\exp(\eta_c^\intercal \bar{z}_d)}{\sum_{c'} \exp(\eta_{c'}^\intercal \bar{z}_d)}. \tag{5.4}$$

We optimize the objective (Equation 5.4) using L-BFGS and a regularizing Gaussian prior with $\mu = 0, \sigma^2 = 1$.

While EM is sensitive to initialization, CSLDA is straightforward to initialize. Majority vote is used to set initial y values $\tilde{y}$. Corresponding initial values for z and η are obtained by clamping y to $\tilde{y}$ and running stochastic EM on z and η.



(a) Hyperparameters fixed



(b) Hyperparameters optimized via grid search on validation data

Figure 5.4: Differences among the inferred label accuracy learning curves of three ITEMRESP inference algorithms are reduced when hyperparameters are optimized.

5.2.4 Hyperparameter Optimization

Ideally, we would test CSLDA performance under all of the many algorithms available for inference in such a model. Although that is not feasible, Asuncion et al. [1] demonstrate that hyperaparameter optimization in LDA topic models helps to bring the performance of alternative inference algorithms into approximate agreement. Accordingly, in Section 5.2.4 we implement hyperparameter optimization for CSLDA to make our results as general as possible.

Before moving on, however, we take a moment to validate that the observation of Asuncion et al. generalizes from LDA to the ITEMRESP model, which, together with LDA, comprises CSLDA. Figure 5.4 demonstrates that three ITEMRESP inference algorithms, Gibbs sampling (Gibbs), mean-field variational inference (Var), and the iterated conditional modes algorithm (ICM) [5], are brought into better agreement after optimizing their hyperparameters via grid search. That is, the algorithms in Figure 5.4b are in better agreement, particularly near the extremes, than the algorithms in Figure 5.4a. This difference is subtle, but it holds to an equal and greater extent in other simulation conditions we tested (experiment details are similar to those reported in Section 5.3).

Fixed-point Hyperparameter Updates

Although a grid search is effective, it is not practical for a model with many hyperparameters such as CSLDA. For efficiency, therefore, we use the fixed-point updates of Minka [89]. Our updates differ slightly from Minka's since we tie hyperparameters, allowing them to be learned more quickly from less data. In our implementation the matrices of hyperparameters $b^{(\phi)}$ and $b^{(\theta)}$ over the Dirichlet-multinomial distributions are completely tied such that $b^{(\phi)}_{tv} = b^{(\phi)} \forall t, v$ and $b^{(\theta)}_{t} = b^{(\theta)} \forall t$. This leads to

$$b^{(\phi)} \leftarrow b^{(\phi)} . \frac{\sum_{t,v}[\Psi(N_{tv}+b^{(\phi)})]-TV\Psi(b^{(\phi)})}{V[\Psi(N_t+Vb^{(\phi)})-\Psi(Vb^{(\phi)})]} \tag{5.5}$$

and

$$b^{(\theta)} \leftarrow b^{(\theta)} . \frac{\sum_{d,t}[\Psi(N_{dt}+b^{(\theta)})]-NT\Psi(b^{(\theta)})}{T[\Psi(N_d+Tb^{(\theta)})-\Psi(Tb^{(\theta)})]}. \tag{5.6}$$

The updates for $b^{(\gamma)}$ are slightly more involved since we choose to tie the diagonal entries $b^{(\gamma)}_d$ and separately the off-diagonal entries $b^{(\gamma)}_o$, updating each separately:

$$b^{(\gamma)}_d \leftarrow b^{(\gamma)}_d . \frac{\sum_{j,c}[\Psi(N_{jcc}+b^{(\gamma)}_d)]-JC\Psi(b^{(\gamma)}_d)}{Z(b^{(\gamma)})} \tag{5.7}$$

and $b^{(\gamma)}_o \leftarrow$

$$b^{(\gamma)}_o . \frac{\sum_{j,c,c' \neq c}[\Psi(N_{jcc'}+b^{(\gamma)}_o)]-JC(C-1)\Psi(b^{(\gamma)}_o)]}{(C-1)Z(b^{(\gamma)})} \tag{5.8}$$

where

$$Z(b^{(\gamma)}) = \sum_{j,c}[\Psi(N_{jc} + b^{(\gamma)}_d + (C-1)b^{(\gamma)}_o)] - JC\Psi(b^{(\gamma)}_d + (C-1)b^{(\gamma)}_o).$$

As in the work of Asuncion et al. [1], we add an algorithmic gamma prior ($b^{(\cdot)} \sim G(\alpha, \beta)$) for smoothing by adding $\frac{\alpha-1}{b^{(\cdot)}}$ to the numerator and β to the denominator of Equations 5.5-5.8. Note that these algorithmic gamma "priors" should not be understood as first-class members of the CSLDA model (Figure 5.3). Rather, they are regularization terms that keep our hyperparameter search algorithm from straying towards problematic values such as 0 or ∞.

5.3 Experiments

For all experiments we set CSLDA's number of topics T to 1.5 times the number of classes in each dataset. We found that model performance was reasonably robust to this parameter. Only when T drops below the number of label classes does performance suffer. As per Section 5.2.3, z and η values are initialized with 500 rounds of stochastic EM, after which the full model is updated with 1000 additional rounds. Predictions are generated by aggregating samples from the last 100 rounds (the mode of the approximate marginal posterior).

We compare CSLDA with (1) a majority vote baseline, (2) the ITEMRESP model, and representatives of the two main classes of data-aware crowdsourcing models, namely (3) data-generative and (4) data-conditional. MOMRESP represents a typical data-generative model [9, 36, 69, 120]. Data-conditional approaches typically model data features conditionally using a log-linear model [59, 79, 110, 140]. For the purposes of this paper, we refer to this model as LOGRESP. For ITEMRESP, MOMRESP, and LOGRESP we use the variational inference methods presented by Felt et al. [40]. Unlike that paper, in this work we have augmented inference with the in-line hyperparameter updates described in Section 5.2.4.

5.3.1 Human-generated Annotations

To gauge the effectiveness of data-aware crowdsourcing models, we use the sentiment-annotated tweet dataset distributed by CrowdFlower as a part of its "data for everyone" initiative.[2] In the "Weather Sentiment" task, 20 annotators judged the sentiment of 1000 tweets as either positive,

[2] http://www.crowdflower.com/data-for-everyone

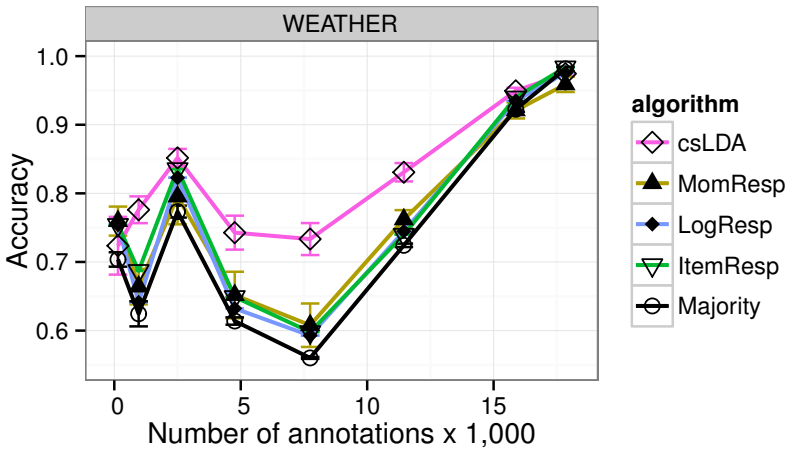


Figure 5.5: Inferred label accuracy of models on sentiment-annotated weather tweets.

negative, neutral, or unrelated to the weather. In the secondary "Weather Sentiment Evaluated" task, 10 additional annotators judged the correctness of each consensus label. We construct a gold standard from the consensus labels that were judged to be correct by 9 of the 10 annotators in the secondary task.

Figure 5.5 plots learning curves of the accuracy of model-inferred labels as annotations are added (ordered by timestamp). All methods, including majority vote, converge to roughly the same accuracy when all 20,000 annotations are added. When fewer annotations are available, statistical models beat majority vote, and CSLDA is considerably more accurate than other approaches. Learning curves are bumpy because annotation order is not random and because inferred label accuracy is calculated only over documents with at least one annotation. Learning curves collectively increase when average annotation depth (the number of annotations per item) increases and decrease when new documents are annotated and average annotation depth decreases. CSLDA stands out by being more robust to these changes than other algorithms, and also by maintaining a higher level of accuracy across the board. This is important because high accuracy using fewer annotations translates to decreased annotations costs.

	D	C	V	$\frac{1}{N}\sum_d N_d$
20 News	16,995	20	22,851	111
WebKB	3,543	4	5,781	131
Reuters8	6,523	8	6,776	53
Reuters52	7,735	52	5,579	58
CADE12	34,801	12	41,628	110
Enron	3,854	32	14,069	431

Table 5.2: Dataset statistics. D is number of documents, C is number of classes, V is number of features, and $\frac{1}{N}\sum_d N_d$ is average document size. Values are calculated after setting aside 15% as validation data and doing feature selection.

5.3.2 Synthetic Annotations

Datasets including both annotations and gold standard labels are in short supply. Although plenty of text categorization datasets have been annotated, common practice reflects that initial noisy annotations be discarded and only consensus labels be published. Consequently, we follow previous work in achieving broad validation by constructing synthetic annotators that corrupt known gold standard labels. We base our experimental setup on the annotations gathered by Felt et al. [40],[3] who paid CrowdFlower annotators to relabel 1000 documents from the well-known 20 Newsgroups classification dataset. In that experiment, 136 annotators contributed, each instance was labeled an average of 6.9 times, and annotator accuracies were distributed approximately according to a $Beta(3.6, 5.1)$ distribution. Accordingly we construct 100 synthetic annotators, each parametrized by an accuracy drawn from $Beta(3.6, 5.1)$ and with errors drawn from a symmetric Dirichlet $Dir(1)$. Datasets are annotated by selecting an instance (at random without replacement) and then selecting K annotators (at random without replacement) to annotate it before moving on. We choose $K = 7$ to mirror the empirical average in the CrowdFlower annotation set.

We evaluate on six text classification datasets, summarized in Table 5.2. The 20 Newsgroups, WebKB, Cade12, Reuters8, and Reuters52 datasets are described in more detail by Cardoso-Cachopo [14]. The LDC-labeled Enron emails dataset is described by Berry et al. [4]. Each dataset is

[3]The dataset is available via git at `git://nlp.cs.byu.edu/plf1/crowdflower-newsgroups.git`

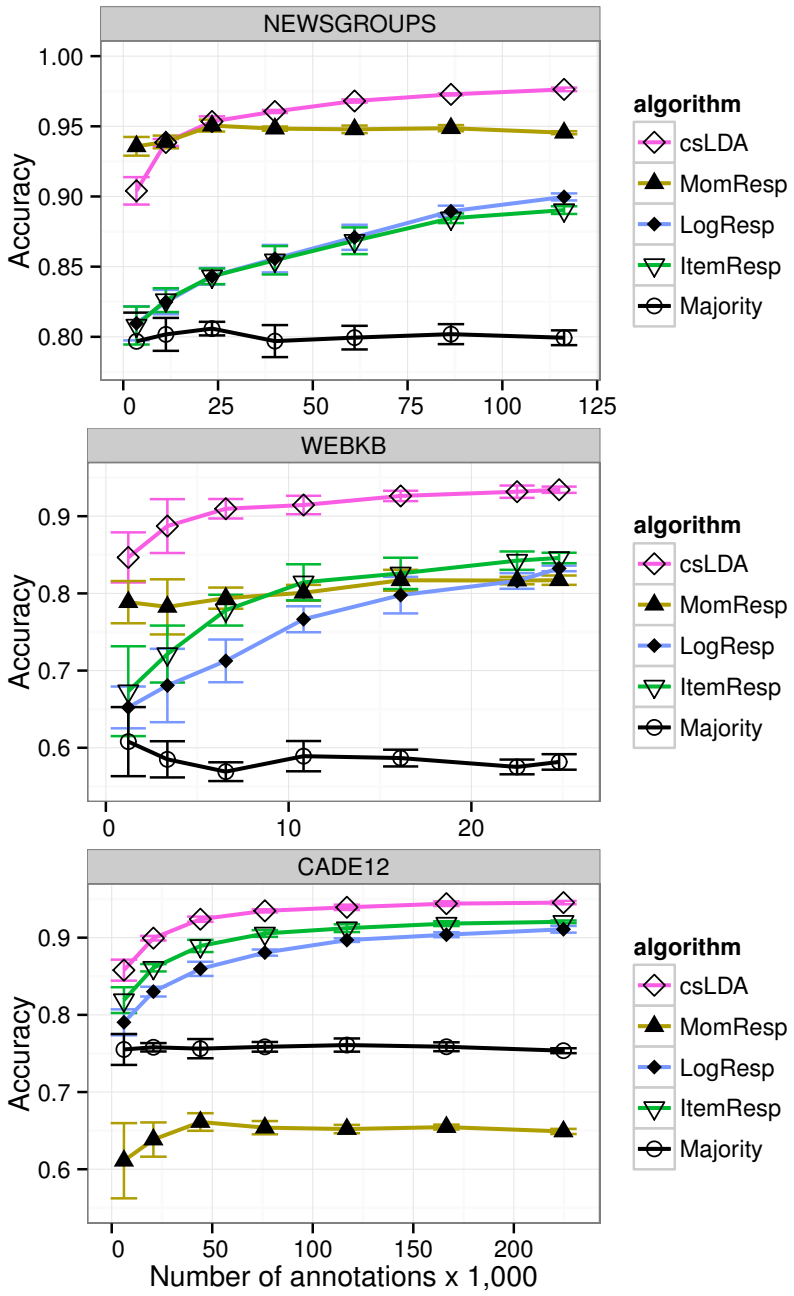


Figure 5.6: Inferred label accuracy of models on synthetic annotations. The first instance is annotated 7 times, then the second, and so on.

preprocessed via Porter stemming and by removal of the stopwords from MALLET's stopword list [85]. Features occurring fewer than 5 times in the corpus are discarded. In the case of MOMRESP, features are fractionally scaled so that each document is the same length, in keeping with previous work in multinomial document models [100].

95% Accuracy	csLDA	MomResp	LogResp	ItemResp	Majority
20 News	**85 (5.0x)**	150 (8.8x)	152 (8.9x)	168 (9.9x)	233 (13.7x)
WebKB	**31 (8.8x)**	-	46 (13.0x)	46 (13.0x)	-
Reuters8	**25 (3.8x)**	-	73 (11.2x)	62 (9.5x)	-
Reuters52	**33 (4.3x)**	73 (9.4x)	67.5 (8.7x)	60 (7.8x)	87 (11.2x)
CADE12	**250 (7.2x)**	-	295 (8.5x)	290 (8.3x)	570 (16.4x)
Enron	**31 (8.0x)**	-	40 (10.4x)	38 (9.9x)	47 (12.2x)

Table 5.3: The number of annotations $\times 1000$ at which the algorithm reaches 95% inferred label accuracy on the indicated dataset (average annotations per instance are in parenthesis). All instances are annotated once, then twice, and so on. Empty entries ('-') do not reach 95% even with 20 annotations per instance.

Figure 5.6 plots learning curves on three representative datasets (Enron resembles Cade12, and the Reuters datasets resemble WebKB). csLDA consistently outperforms LogResp, ItemResp, and majority vote. The generative models (csLDA and MomResp) tend to excel in low-annotation portions of the learning curve, partially because generative models tend to converge quickly and partially because generative models naturally learn from unlabeled documents (i.e., semi-supervision). However, MomResp tends to quickly reach a performance plateau after which additional annotations do little good. The performance of MomResp is also highly dataset dependent: it is good on 20 Newsgroups, mediocre on WebKB, and poor on CADE12. By contrast, csLDA is relatively stable across datasets.

To understand the different behavior of the two generative models, recall that MomResp is identical to ItemResp save for its multinomial data model. Indeed, the equations governing inference of label y in MomResp simply sum together terms from an ItemResp model and terms from a mixture of multinomials clustering model (and for reasons explained in Section 5.2.1, the multinomial data model terms tend to dominate). Therefore when MomResp diverges from ItemResp it is because MomResp is attracted toward a y assignment that satisfies the multinomial data model, grouping similar documents together. This can both help and hurt. When data clusters and label classes are misaligned, MomResp falters (as in the case of the Cade12 dataset). In

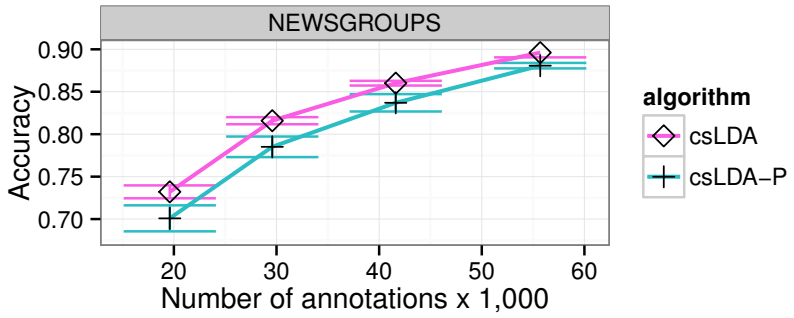


Figure 5.7: Joint inference for CSLDA vs pipeline inference (CSLDA-P).

contrast, CSLDA's flexible mapping from topics to labels is less sensitive: topics can diverge from label classes so long as there exists some linear transformation from the topics to the labels.

Many corpus annotation projects are not complete until the corpus achieves some target level of quality. We repeat the experiment reported in Figure 5.6, but rather than simulating seven annotations for each instance before moving on, we simulate one annotation for each instance, then two, and so on until each instance in the dataset is annotated 20 times. Table 5.3 reports the minimal number of annotations before an algorithm's inferred labels reach an accuracy of 95%, a lofty goal that can require significant amounts of annotation when using poor quality annotations. CSLDA achieves 95% accuracy with fewer annotations, corresponding to reduced annotation cost.

5.3.3 Joint vs Pipeline Inference

To isolate the effectiveness of joint inference in CSLDA, we compare against the pipeline alternative where topics are inferred first and then held constant during inference [73]. Joint inference yields modest but consistent benefits over a pipeline approach. Figure 5.7 highlights a portion of the learning curve on the Newsgroups dataset (based on the experiments summarized in Table 5.3). This trend holds across all of the datasets that we examined.

5.3.4 Error Analysis

Class-conditional models like MOMRESP include a feature that data-conditional models like CSLDA lack: an explicit prior over class prevalence. Figure 5.8a shows that CSLDA performs poorly on the

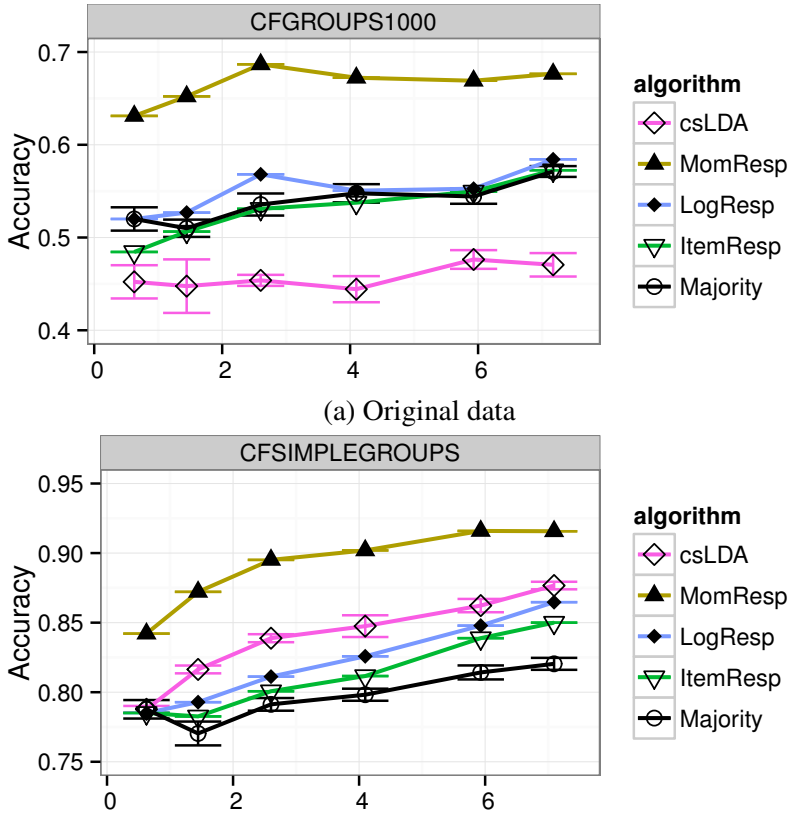
CrowdFlower-annotated Newsgroups documents described at the beginning of Section 5.3 (not the synthetic annotations). Error analysis uncovers that CSLDA lumps related classes together in this dataset. This is because annotators could specify up to 3 simultaneous labels for each annotation, so that similar labels (e.g., "talk.politics.misc" and "talk.politics.mideast") are usually chosen in blocks. Suppose each member of a set of documents with similar topical content is annotated with both label A and B. In this scenario it is apparent that CSLDA will achieve its best fit by inferring all documents to have the same label either A or B. By contrast, MOMRESP's uniform prior distribution over θ leads it to prefer solutions with a balance of A and B.

The hypothesis that class combination explains CSLDA's performance is supported by Figure 5.8b, which shows that CSLDA recovers after combining the classes that were most frequently co-annotated. We greedily combine label class pairs to maximize Krippendorf's α until only 10 labels were left: "alt.atheism," religion, and politics classes were combined; also, "sci.electronics" and the computing classes. The remaining eight classes were unaltered. However, one could also argue that the original behavior of CSLDA is in some ways desirable. That is, if two classes of documents are mostly the same both topically and in terms of annotator decisions, perhaps those classes ought to be collapsed. We are not overly concerned that MOMRESP beats CSLDA in Figure 5.8, since this result is consistent with early relative performance in simulation.

5.4 Additional Related Work

This section reviews related work not already discussed. A growing body of work extends the item-response model to account for variables such as item difficulty [104, 136, 142], annotator trustworthiness [57], correlation among various combinations of these variables [143], and change in annotator behavior over time [120].

Welinder et al. [135] carefully model the process of annotating objects in images, including variables for item difficulty, item class, and class-conditional perception noise. In follow-up work, Liu et al. [79] demonstrate that similar levels of performance can be achieved with the simple item-response model by using variational inference rather than EM. Alternative inference algorithms



(a) Original data

(b) After combining frequently co-annotated label classes

Figure 5.8: An illustrative failure case. CSLDA, lacking a class label prior, prefers to combine label classes that are highly co-annotated.

have been proposed for crowdsourcing models [27, 48, 62, 141]. Some crowdsourcing work regards labeled data not as an end in itself, but rather as a means to train classifiers [76]. The fact-finding literature assigns trust scores to assertions made by untrusted sources [106].

5.5 Conclusion and Future Work

We describe CSLDA, a generative, data-aware crowdsourcing model that addresses important modeling deficiencies identified in previous work. In particular, CSLDA handles data in which the natural document clusters are at odds with the intended document labels. It also transitions smoothly from situations in which few annotations are available to those in which many annotations are available. Because of the flexible mapping in CSLDA to class labels, many structural variants are possible in future work. For example, this mapping could depend not just on inferred topical

content but also directly on data features (c.f. Nguyen et al. [99]) or learned embedded feature representations.

The large number of parameters in the learned confusion matrices of crowdsourcing models present difficulty at scale. This could be addressed by modeling structure both inside of the annotators and classes. Redundant annotations give unique insights into both inter-annotator and inter-class relationships and could be used to induce annotator or label class hierarchies with parsimonious representations. Simpson et al. [121] identify annotator clusters using community detection algorithms but do not address annotator hierarchy or scalable confusion representations.

Chapter 6

Semantic Annotation Aggregation with Conditional Crowdsourcing Models and Word Embeddings

To be submitted to TACL 2016

This chapter completes the thread exploring how to best incorporate data into crowdsourcing models by answering the question of what to do when an appropriate generate data model does not exist for the data being labeled. We identify a methodology for dealing with difficult or unusual data that yields state-of-the-art results on a number of annotation aggregation tasks.

Abstract

In modern text annotation projects, crowdsourced annotations are often aggregated using item response models or by majority vote. Recently, item response models enhanced with generative data models have been shown to yield substantial benefits over those with conditional or no data models. However, suitable generative data models do not exist for many tasks, such as semantic labeling tasks. When no generative data model exists, we demonstrate that similar benefits may be derived by conditionally modeling documents that have been previously embedded in a semantic space using recent work in vector space models. We use this approach to show state-of-the-art results on a variety of challenging semantic annotation aggregation tasks.

6.1 Introduction

Text annotation is a crucial part of natural language processing (NLP), enabling content analysis [65] and providing training data for supervised and semi-supervised machine learning algorithms

[81] in NLP. Modern text annotation is often crowdsourced, meaning that the work is divided up and assigned to internet workers on micro-task marketplaces such as Amazon's Mechanical Turk[1] or CrowdFlower.[2] Although crowdsourced annotations tend to be error-prone, high quality labels may be obtained by aggregating multiple redundant low-quality annotations. For many tasks, aggregated crowdsourced judgments have been shown to be more reliable than expert judgments [13, 61, 123, 128].

Traditionally annotations were aggregated via majority vote. More sophisticated approaches jointly model annotator reliability and document labels. By modeling both annotator and annotation data, these models can, in a principled way, down-weight the annotations of workers who often disagree with others, and up-weight the annotations of workers who often agree with others. However, when annotation error is high or few annotations are available it can be difficult for these models of annotation to know which annotators to trust. Jin and Ghahramani [59], Raykar et al. [110], Liu et al. [79], and Yan et al. [140] show that crowdsourcing annotation models can be enhanced by conditioning the model on the document data (e.g., word content), improving the model by identifying annotators whose judgments tend to agree with word patterns found in the documents being annotated.

Recently, Felt et al. [40] argued that generative data models allow annotation models to converge to useful estimates even when few annotations are available, and argue that by the time a conditional model has enough information to be useful, there are often enough annotations available that the problem is mostly solved by majority vote. However, although generative data modeling can be effective for simple problems, such as topical text categorization, effective generative data models are not available for more complex and nuanced semantic tasks such as sentiment labeling, semantic compatibility classification and paraphrase identification. In this paper, we use recent advances in word and document representation to demonstrate that data-conditional annotation models can get benefits similar to those of data-generative annotation models, but for a larger variety of tasks and data.

[1] http://mturk.com
[2] http://crowdflower.com

In Section 6.2 we briefly review annotation models with generative and conditional data components and also discuss representing words and documents via embeddings in a semantic vector space. In Section 6.3 we show that data-conditional annotation models succeed on a variety of text datasets and classification tasks. In Section 6.4 we conduct error analysis on an anomalous dataset, and in Sections 6.5 and 6.6 we list additional related work and summarize our conclusions.

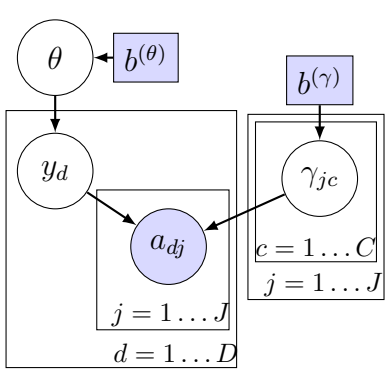
6.2 Background

Most crowdsourcing models extend the item-response model of Dawid and Skene [30]. The Bayesian version of this model, referred to here as ITEMRESP, is illustrated by Figure 6.1a and defines the joint distribution $p(y, a, \theta, \gamma)$. In the generative story for this model, a confusion matrix γ_j is drawn for each human annotator j. Each row γ_{jc} of the confusion matrix γ_j is drawn from $Dir(b_{jc}^{(\gamma)})$, and encodes a probability distribution over label classes that annotator j is apt to choose when presented with a document whose true label is c. A general prior over label classes θ is drawn from $Dirichlet(b^{(\theta)})$, then for each document d an unobserved document label y_d is drawn from categorical distribution $Cat(\theta)$. Finally, annotations are generated as annotator j corrupts the true label y_d according to the multinomial distribution $Mult(\gamma_{jy_d})$, allowing for multiple annotations.

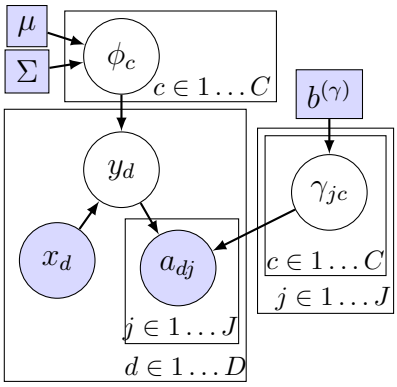
6.2.1 Data-aware annotation models

Notice that the ITEMRESP model entirely ignores document data x (e.g., words). ITEMRESP extensions model the data x and related feature parameters ϕ either conditionally $p(y, a, \gamma, \phi|x)$ or else generatively $p(y, a, x, \theta, \gamma, \phi)$.

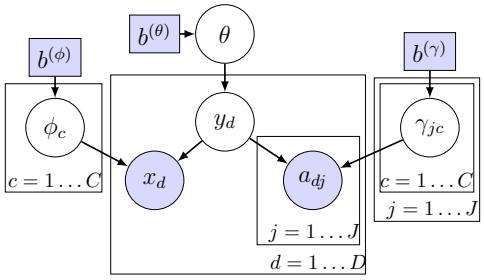
Conditional crowdsourcing models are appealing since they make few assumptions about the data, and can use the same general log-linear structure as maximum entropy classifiers, which have enjoyed success in a large number of classification problems. Figure 6.1b shows a Bayesian formulation of a conditional crowdsourcing model $p(y, a, \gamma, \phi|x)$. For each class k, ϕ_k is drawn from a multivariate Gaussian distribution $Gauss(0, \Sigma)$. Then for each document, y_d is drawn from a log-linear distribution $p(y_d|\phi, x) \propto e^{\phi_{y_d}^T x}$. For this reason, and by analogy with ITEMRESP, we



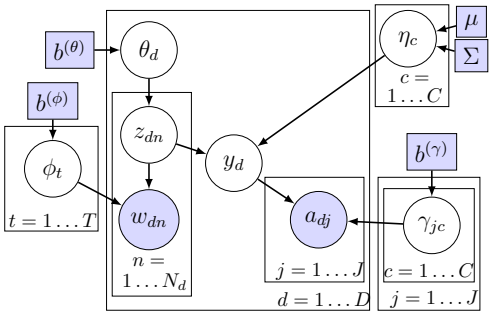
(a) **ITEMRESP** as a plate diagram.



(b) **LOGRESP** as a plate diagram.



(c) **MOMRESP** as a plate diagram.



(d) **CSLDA** as a plate diagram.

Figure 6.1: Round nodes are variables with distributions. Rectangular nodes are values without distributions. Shaded nodes are observed. D, J, C are the number of documents, annotators, and classes, respectively.

refer to this model as LOGRESP. Although there are many variations on this basic structure, LOGRESP is representative of a popular class of conditional crowdsourcing models [59, 79, 110, 140]. Unfortunately, Felt et al. [40] found that in many cases LOGRESP provides only incremental gains over majority vote, in part because its ϕ estimates, like other conditional models, tend to converge relatively slowly with $O(N)$ labeled examples [96]. By the time LOGRESP's ϕ estimates become useful, there are often enough annotations available that majority vote is sufficient.

Generative data-aware crowdsourcing models have complementary strengths. Although they make strong assumptions about the data that they model, their parameters can converge quickly with only $O(logn)$ labeled examples [96]. This means that when data does not violate a generative model's assumptions too badly, the generative model can offer dramatic improvements over majority vote, especially when few annotations are available. Figures 6.1c and 6.1d depict two such generative

models. In Figure 6.1c, each document d draws its data x_d from a class-conditional multinomial word distribution $Mult(\phi_{y_d})$. We call this model MOMRESP because it models data as a mixture of multinomials, and also with reference to ITEMRESP. MOMRESP represents a common class of generative crowdsourcing models [9, 69, 120]. Figure 6.1d shows CSLDA, a more sophisticated generative crowdsourcing model based on supervised topic modeling, recently proposed by Felt et al. [39]. In CSLDA, data x is modeled as an admixture of topics z. True document labels are drawn from a log-linear distribution $p(y_d|\bar{z}_d, \phi)$ where $\bar{z}_d$ is the empirical topical distribution for document d.

For inference in the ITEMRESP, LOGRESP and MOMRESP crowdsourcing models, we use the variational inference algorithms proposed by Felt et al. [40]. Note that variational inference for ITEMRESP is easily derived as a special case of MOMRESP inference where terms involving the data are dropped. Inference for CSLDA is stochastic expectation maximization.

6.2.2 Word and Document Representations

Documents have historically been represented in NLP algorithms by large, sparse word count vectors $x_d = \sum_{n=1}^{|x_d|} \mathbb{1}(x_{dn})$ where $\mathbb{1}(x_{dn})$ is a one-hot vector having length equal to the size of the vocabulary. However, word-count document representations have a number of drawbacks. They define a space that is often so high-dimensional and sparse that inter-document distances and other vector computations have little meaning. In word-count representations, features that strongly relate to one another (e.g., the words "horse" and "equine") are represented as entirely orthogonal dimensions, exploding the number of parameters needed by downstream learning algorithms.

Recently, methods have been developed to represent words using locations in low-dimensional vector spaces where distance and direction encode semantic and syntactic meaning [87, 107]. These embeddings have been shown to improve a variety of language tasks including named entity recognition, phrase chunking [132], relation extraction [98], and part of speech induction [77]. The hypothesis investigated by the current work is that semantic, vector-based text representations can help conditional annotation aggregation models achieve some of the same early performance

advantage seen in their generative counterparts, as well as help them operate on datasets that make challenging semantic distinctions. This hypothesis is plausible *a priori* because using data embeddings is akin to using semi-supervision to enable faster learning. The reason for this is that data embeddings are traditionally induced in an unsupervised manner on extremely large corpora before being applied to a downstream supervised task. In addition, operating on dense, low-dimensional vector data reduces the number of model parameters that must be learned which can also reduce the number of instances required to learn effectively.

Although a large number of alternative embedding methods have been proposed, the word2vec algorithm introduced by Mikolov et al. [87] is still competitive with most general-purpose embedding methods and remains widely used. Accordingly, we choose to use word2vec for the purposes of this paper, understanding that other embedding methods may be swapped in for additional improvements as they are developed. Word2vec embeds individual words, rather than sentences or documents. When entire sentences or documents must be embedded, we do so simply by using the average word embedding. Although this simple approach works well enough for the purposes of this paper, finding good document and phrase embeddings are active areas of research, and better methods of embedding sentences and documents will only improve the results presented here.

6.3 Experiments

In order to test the hypothesis that vector space document representations can improve conditional crowdsourcing model performance on semantic classification tasks, we plot and visually compare learning curves charting the accuracy of the labels inferred by various crowdsourcing models. Learning curves advance as annotations from multiple annotators are incrementally added to the set of annotations available to each model. When annotation timestamps are available, annotations are added in the empirical order in which they were created. When unavailable, annotations are added in randomized breadth-first order so that each document gets one annotation before any document receives a second. This process illustrates model behavior both when few annotations per document

Dataset	Size	Unique Annotators	Annotations per Instance	Classes	Average Doc size	Gold Labels	Timestamps
Sentiment	1,000	83	5	2	12.8	1,000	No
Weather	1,000	102	20	5	13.6	724	Yes
Compatibility	17,977	411	10	2	2×1	15,157	No
Paraphrase	4,000	119	5	2	2×11.2	838	Yes

Table 6.1: Dataset statistics. Evaluation metrics are calculated only over the subset of each dataset for which gold labels are available. The timestamps column indicates whether or not it is known exactly when each annotation was generated. When available, timestamps determine the order of annotation in reported learning curves.

are available (in the early stages of learning curves) and when many annotations per document are available (in the late stages of the learning curves).

For our word embedding model, we use the popular word2vec algorithm, implemented by the gensim document processing library [111] to train word embeddings on a June 2015 snapshot of the English Wikipedia pages and articles dump (approximately 2.1 billion words). The word2vec algorithm requires specifying a number of parameters, which we briefly report here for replicability. We train embeddings using a context window of 10 words, discarding words that occur fewer than 5 times. For training, we use hierarchical sampling with a skip-gram model and no negative sampling. Embeddings of size 300 are learned. All of these settings are rather standard for a large corpus like Wikipedia.

6.3.1 Datasets

In order to calculate the accuracy of inferred labels, we require datasets that have both crowdsourced annotations as well as gold standard labels for evaluation. We identify four suitable datasets, briefly describing both their annotation task as well as the way their gold standard labels are constructed. For all Twitter data, we use the Twitter text normalization dictionary of Han et al. [54] to normalize tweets before embedding them.

Paraphrase. During an exploratory annotation phase, Xu et al. [138] paid Amazon Mechanical Turk workers to annotate 4,000 tweet pairs with binary judgments indicating whether or not the

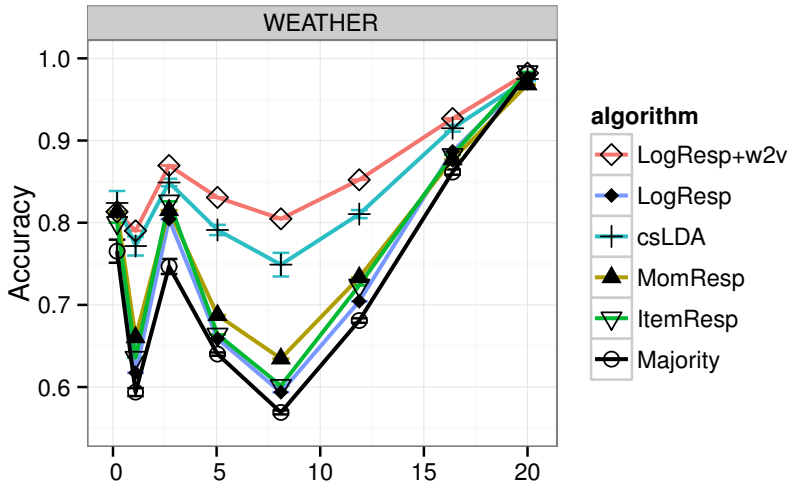
tweet pair communicates the same information.[3] For example, the tweets "Star Wars Return of the Jedi is on" and "My favorite Star Wars movie is on" communicate mostly the same information and are labeled as paraphrases of one another, while the tweet "and of course because I drink and like Star Wars I know nothing about football" communicates different information, and is labeled as not a paraphrase of the other two tweets. Each tweet pair received 5 binary annotations. Gold standard labels were constructed for a subset of the annotations by experts who rated each pair on a scale from 1-5. Following the original authors, expert ratings of 0-2 are labeled *no paraphrase*,and 4-5 are labeled *paraphrase*.

Compatibility. Kruszewski and Baroni [67] paid CrowdFlower workers to rate word pairs according to their degree of semantic compatibility, meaning that the two words can be used to refer to the same real-world entity.[3] For example, the words "artist" and "teacher" are compatible with one another, whereas "bread" and "rattlesnake" are not. Each word pair was rated by 10 different annotators on a 7-point scale. Following the original authors, the gold standard is constructed by labeling items with a mean rating less than 1.6 as incompatible, and those with a mean rating greater than 3.7 as compatible.

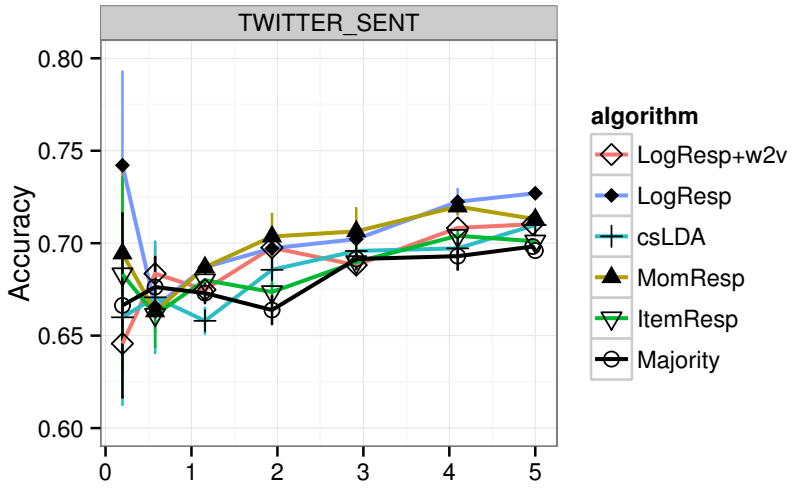
Sentiment. Mozafari et al. [92] paid Amazon Mechanical Turk workers to annotate tweets with binary sentiment labels: *Positive* and *Negative*, and manually created gold standard labels using trusted labelers.[4]

Weather. CrowdFlower has made a number of annotated datasets freely available.[5] In their "Weather sentiment" dataset, 20 annotators were paid to annotate weather-related tweets with sentiment labels: *Negative*, *Neutral*, *Positive*, *Unrelated to weather*, and *I can't tell*. A gold standard was constructed by running a separate evaluation task called "Weather sentiment evaluated" in which 10 additional annotators were paid to annotate the majority vote label from the previous task as correct or incorrect. We form a gold standard from those labels that are judged to be correct by at least 9/10 annotators.

[3]Not publically available at the time of writing.
[4]`http://web.eecs.umich.edu/~mozafari/datasets/crowdsourcing/index.html`
[5]`http://www.crowdflower.com/data-for-everyone`



(a) The **Weather** Dataset



(b) The **Sentiment** Dataset

Figure 6.2: Inferred label accuracy (y axis) learning curves of various crowdsourcing models. The x axis is the number of annotations $\times$ 1,000.

6.3.2 Comparison with lexical methods

Two of our datasets, **Weather** and **Sentiment**, are traditional text classification tasks containing instances with one text document receiving one label. We use these datasets to compare the performance of LOGRESP using vector space text features to the performance of common alternatives that use word-count features, including LOGRESP as well as the generative models MOMRESP and CSLDA. The majority vote and ITEMRESP algorithms serve as baselines. In Figure 6.2a we see that on the **Weather** dataset, LOGRESP with embeddings (LOGRESP+w2v) performs far better than traditional LOGRESP, and even outperforms the previous state-of-the-art for this dataset, CSLDA.

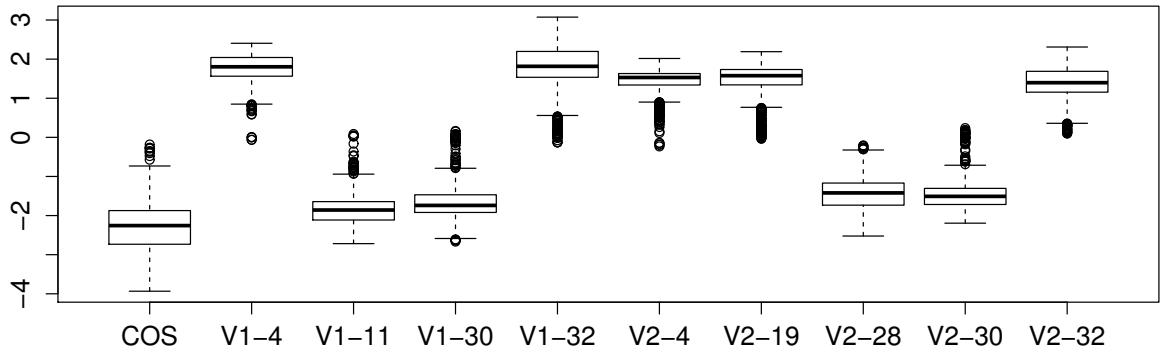


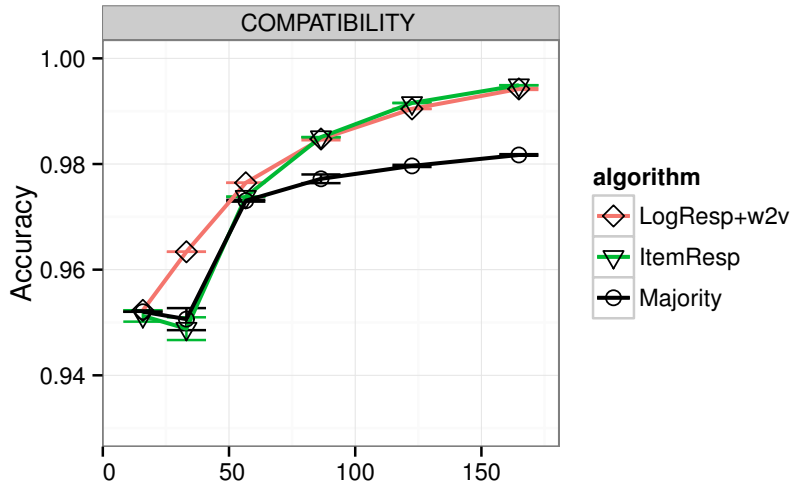
Figure 6.3: The ten most significant features according to regression weight magnitude. Mean and variance is based on regression weights learned at over 2,000 points on the **Paraphrase** dataset learning curve. *COS* is $cos(v_1 - v_2)$ and *Vn-k* is the kth dimension of $PC_{50}(v_n)$.

Although all algorithms eventually reach a high level of performance on the **Weather** dataset, we prefer algorithms like LOGRESP+w2v that reach high levels of accuracy using as few annotations as possible, potentially allowing us to save money by paying for fewer annotations.

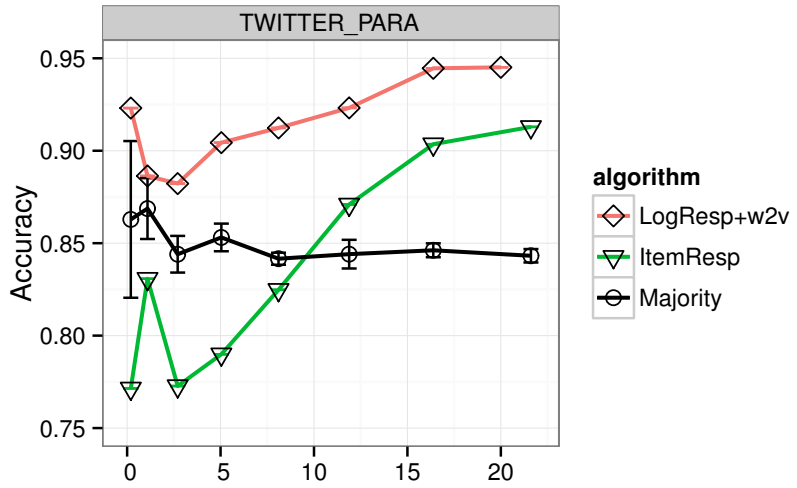
In Figure 6.2b we see that no algorithm improves much over majority vote on the **Sentiment** dataset. Also, the baseline accuracy levels at the end of the curves are extremely low for a binary classification task with 5 annotations per instance, meaning that annotator accuracy is unusually low. We include this dataset as a reminder that the "no free lunch" theorem applies to crowdsourcing models the same as to any other class of models. In Section 6.4 we explore in more detail what makes the **Sentiment** dataset particularly difficult for crowdsourcing models.

6.3.3 When lexical methods do not apply

The datasets **Compatibility** and **Paraphrase** both involve data pairs being compared for semantic content (see Section 6.3.1 for examples of these tasks). **Compatibility** compares the semantic compatibility of word pairs while **Paraphrase** compares the semantic similarity of tweets pairs. Generative crowdsourcing models such as MOMRESP and CSLDA do not natively accommodate such paired data since the data does not comport with these models' generative stories. To make them do so would require restructuring the models and their inference procedures. On the other



(a) The **Compatibility** Dataset



(b) The **Paraphrase** Dataset

Figure 6.4: Inferred label accuracy (y axis) learning curves of vector space crowdsourcing models on tasks with paired-comparison data for which generative crowsdsourcing models are unsuitable. The x axis is the number of annotations $\times$ 1,000.

hand, it is straightforward to combine the semantic vector representations of two documents v_1 and v_2. We do so by forming a new feature vector

$$v_{new} = \langle cos(v_1, v_2), \tag{6.1}$$

$$L1(v_1 - v_2), L2(v_1 - v_2), \tag{6.2}$$

$$PC_{50}(v_1), PC_{50}(v_2), \tag{6.3}$$

$$PC_{50}(v_1) - PC_{50}(v_2)\rangle \tag{6.4}$$

where $cos(\cdot)$ is cosine distance, $L1(\cdot)$ and $L2(\cdot)$ are the first two p-norms, and $PC_n(v)$ is a vector consisting of the top-n components of v, found via PCA on the set of embedded documents.

Figure 6.4a shows that LOGRESP with semantic embeddings outperforms majority vote and ITEMRESP baselines on the word compatibility dataset when there fewer than 3 annotations per instance. When annotations become highly available in the later portions of the curves (up to 20 per instance), the absolute accuracy level of ITEMRESP approaches 1 and the data component has little effect. When information from annotations is sufficiently abundant, informing predicted labels from other sources (such as the data) yields very little additional gain. Fortunately, incorporating data information using LOGRESP with semantic embeddings appears to never actually hurt compared with using just ITEMRESP.

On the other hand, Figure 6.4b shows that on the paraphrase dataset, LOGRESP with semantic embeddings dramatically outperforms the baselines along the entire learning curve. Paraphrase dataset instances have only between 0-5 annotations per instance, and the task seems to be slightly more difficult than the compatibility task, with the result that there is room for improvement by a good data model.

In order to get some idea of which features are most helpful, in Figure 6.3 we show the top 10 features according to their binary logistic regression weight magnitudes on the **Paraphrase** dataset. As expected, the $cos(v_1, v_2)$ feature is most informative. This is followed by select PCA dimensions. Features of the form $PC_{50}(v_1) - PC_{50}(v_2)$ appear to be less important. This is confirmed by a handful of tests (not reported here) showing large drops in accuracy when cosine or PCA features were dropped and small perturbations when $PC_{50}(v_1) - PC_{50}(v_2)$ features were dropped.

6.3.4 Summary of experiments

Overall, with the exception of the somewhat anomalous **Sentiment** dataset, which we examine in more detail in Section 6.4, running LOGRESP on semantically embedded data always helps. The gains in Figures 6.4a and 6.4b strongly confirm the hypothesis that semantic embeddings can allow crowdsourcing models to see some of the same efficiency gains for challenging semantic

		Alternative Gold Standard			
		Neg	Pos	None	Hard
Gold Standard	Neg	35	**6**	**5**	**1**
	Pos	**9**	29	**11**	**3**

Table 6.2: Disagreement between the original gold standard (rows) and an alternative gold standard (columns) on 100 arbitrarily selected tweets. The alternative gold standard employs a more flexible label set. Neg=*Negative*, Pos=*Positive*, None=*No sentiment*, Hard=*Can't decide*. Bold values reflect various kinds of disagreement between the two labelings.

labeling tasks as previously observed using generative data-aware crowdsourcing models on more straight-forward topical labeling tasks. Not only that, but the fact that semantic embeddings lend themselves to sensible vector-space operations allows data-aware crowdsourcing models to be applied to complex tasks like labeling paired text similarity and compatibility, which was not previously possible.

6.4 Sentiment dataset error analysis

An analysis of the **Sentiment** dataset (results in Figure 6.2b) helps explain why algorithms behave so differently on it than on the other datasets. Kilgarriff [63] identifies three sources of annotation noise: inherent data ambiguity, poor task definition, and annotator error. The crowdsourcing models used here account only for annotator error. The **Sentiment** dataset annotation scheme dictates that each tweet be labeled with a binary sentiment label. Some tweets encode strong sentiment, but whether that sentiment is positive or negative depends on extra-textual context. Errors on these tweets are caused by inherent data ambiguity. For example, "EBTM.com is BACK?!" could be either positive or negative, depending on what the speaker thinks (has previously said) on the subject. Similarly, some tweets encode little or no sentiment. Because a binary label set forces an arbitrary decision, error in these cases grows from an ill-specified annotation task. For example, the tweets

- @comeagainjen if you dont, neither do i
- @sianllewellyn ive txt you this morning

- @meyuy means u should go and shop new pants

all contain very little explicit sentiment. Although we could read sentiment into them (if forced), we would be relying more on our mental models of the world than on textual evidence. Kilgarriff [63] suggests that an important step towards addressing inherent data ambiguity is giving annotators the ability to explicitly identify ambiguous instances.

In order to assess to what degree inherent data ambiguity and task definition affect the **Sentiment** dataset, we arbitrarily chose 100 instances with gold labels and compared them with an alternative gold standard labeled according to a more flexible annotation scheme. The latter labels were generated by a pair of graduate students working in tandem. We added a *No sentiment* label to address problems with task definition and a *Can't decide* label to capture inherent data ambiguity. Table 6.2 shows the confusion matrix between the two gold standard sets. The important thing to note here is the large number of tweets assigned by the alternative gold standard to the case *No sentiment*: fully 16% of the data. This number indicates that task definition affects this dataset strongly. Further, inspecting the disagreements between *Negative* and *Positive* class assignments in the two gold standards revealed that most disagreement happens when tweets contain both positive and negative sentiment. For example, in the tweet "@ceeecil Haha my brain hurts from having to do all the work." the word "haha" indicates positive sentiment but the rest of the tweet seems negative. This suggests that the annotation task definition could be further improved with the addition of a "Both positive and negative" label.

We conclude that the **Sentiment** dataset is strongly affected by the decision to use a binary *Positive*, *Negative* label set to label tweet sentiment. Although this analysis does not make this dataset any less interesting (indeed, the problems associated with modeling and correcting the effects of task misspecification are highly interesting from a theoretical point of view), it does warn us that this dataset is less representative than the others of real-world annotation projects where effort is made up-front to iteratively refine an annotation specification before paying for large number of annotations.

6.5 Additional Related Work

A sizable body of research is currently underway to improve vector word representations. Although most commonly word embeddings are trained in an unsupervised manner, they may be tuned to maximize performance on a particular target task [70]. They may also be supervised by multiple tasks simultaneously [24]. Others fit one embedding per word sense rather than per lexical type, improving model fit [94]. Srikumar and Manning [125] embed not only word types, but also label types, modeling the fact that some labels are more similar than others.

Another line of work explores ways of embedding larger spans of text. Although words tend to compose surprisingly well simply via linear combination, many phrases are more than the sum of their parts (e.g., collocations like "White House"). These can be dealt with by using heuristics to identify and combine token phrases [88]. Other approaches incorporate composition functions as first-class constituents of the objective function itself. Mitchell and Lapata [90] motivate a general composition framework and compare a number of simple instantiations, including additive, multiplicative, and tensor product combination. Socher et al. [124] assign vectors representing semantic content and matrices representing semantic transformations to every node in a parse tree. Fyshe et al. [44] focus on learning phrasal representations whose dimensions are easily interpretable by humans, similar to successful models whose topics are easy for humans to recognize and name because they align with a topic distinction known *a priori* to the human.

In this work we have focused on crowdsourcing models that use data to improve annotation aggregation. Passonneau and Carpenter [104] argue that such probabilistic crowdsourcing models are not only useful for inferring labels from annotations, but also that they should be preferred over traditional chance-adjusted agreement heuristics such as Krippendorff's alpha for assessing corpus quality [65]. Other previous work in crowdsourcing ignores the data being annotated, focusing instead on modeling other aspects of the annotation process, such as item difficulty and noise [135, 136]. Hovy et al. [57] model the non-linear nature of human reliability by adding binary variables to each annotator indicating whether they are a spammer or not. These extensions are

orthogonal to the issue explored by this paper and could be incorporated into any of the models used here.

6.6 Conclusions and Future Work

Previous work indicates that generative crowdsourcing models demonstrate good properties when working with datasets that make topic-based class distinctions. Unfortunately, many text classification datasets make complicated distinctions based on semantic context and relationships, for which no good generative text models currently exist. We have demonstrated that vector space text embeddings can be used to gain similar advantages using conditional models and for an even broader class of data. In particular, a conditional crowdsourcing model with semantically embedded data can be made to handle not just text documents but also pairs of documents that are being compared in terms of their semantic content. Using this approach, we have shown state-of-the-art results on a variety of challenging semantic annotation aggregation tasks. Future work includes experimenting with more sophisticated methods for embedding text. It would also be of interest to assess the effect of jointly learning embeddings and hidden labels rather than pipelining the two tasks.

Chapter 7

Learning from Measurements in Crowdsourcing Models: Inferring Ground Truth from Diverse Annotation Types

To be submitted to TACL 2016

This chapter opens a new line of inquiry investigating how to incorporate heterogeneous annotations types into the same annotation aggregation model. It identifies a strong line of previous work from supervised machine learning that has previously been unused in the crowdsourcing literature, and adapts that framework to the multi-annotator setting with strong results. This chapter culminates in a general purpose active learning algorithm capable of jointly selecting annotators, data instances, and annotation types.

Abstract

Manual annotation labor enables supervised machine learning and data analysis. To reduce the cost of manual annotation, tasks are often redundantly assigned to several low-cost internet workers whose judgments are then combined via crowdsourcing models. Unfortunately, existing crowdsourcing models, including the baseline of majority vote, can aggregate only simple label judgments. We build on previous work in learning from rich prior knowledge in exponential family models to formulate a new class of crowdsourcing model that infers unobserved labels based multiple different kinds of annotations. Annotator judgments are given in the form of the expected value of measurement functions. Measurement functions can encode traditional labels, labeled predicates (e.g., labeled features), and estimated label frequency judgments, among others. Crowdsourcing models using rich measurement evidence infer higher quality labels than similar models that use

only traditional label evidence and additionally enable active sample selection that jointly selects annotator, data item, and annotation type in order to reduce annotation effort.

7.1 Introduction

Supervised machine learning requires labeled training corpora, historically produced by laborious and costly annotation projects. Microtask markets such as Amazon's Mechanical Turk and Crowdflower have turned crowd labor into a commodity that can be purchased with relatively little overhead. Despite their low cost and ready availability, crowdsourced judgments can suffer from high error rates. Therefore it is common practice to obtain multiple redundant crowdsourced judgments, or annotations, and rely on the observation that, in aggregate, non-experts often rival or exceed experts by averaging over individual error patterns [61, 123, 128].

A *crowdsourcing model* harnesses the wisdom of the crowd and infers labels based on the evidence of the available annotations, imperfect though they be. A common baseline crowdsourcing method aggregates annotations by *majority vote*, but this approach ignores important information. For example, some annotators are more reliable than others and their judgments ought to be upweighted accordingly [104]. Because assessing unobserved annotator expertise is tangled with estimating ground truth from imperfect annotations, joint inference of these interrelated quantities is necessary. State-of-the-art crowdsourcing methods often employ a probabilistic formalism to infer hidden labels and annotator expertise.

Traditionally, crowdsourcing models assume that annotations occur only in the form of observed label values reported by untrusted annotators. This is unfortunate because labeling individual examples may not always be the most effective way for annotators to communicate their knowledge. For example, an annotator may know *a priori* that movie reviews containing the word 'uninspired' are usually negative. Although this knowledge could apply to hundreds of reviews, the annotator is limited to examining and labeling each review in turn. On the other hand, frameworks for learning from rich prior knowledge in supervised learning allow humans to communicate with a

Measurement Type	$\sigma_{jk}(x, y)$	$\sum_i E_{q(y_i)}[\sigma_{jk}(x, y)]$	$\max(\sigma_{jk})$
Label	$\mathbb{1}(x = x_m, y = c)$	$q(y_m = c)$	1
Labeled Predicate	$\mathbb{1}(f(x) = 1, y = c)$	$\sum_{i \in f(X)} q(y_i = c)$	$\sum_i \mathbb{1}(f(x_i) = 1)$
Label Proportion	$\mathbb{1}(y = c)$	$\sum_i q(y_i = c)$	N

Table 7.1: Some basic measurement features and their expected values.

machine expressively [46], but these frameworks make the simplifying assumption that all human input is equally trustworthy.

In this work we combine the insights of crowdsourcing models with those of rich prior knowledge learning frameworks to formulate a family of crowdsourcing models that are capable of learning from multiple untrusted annotators who are each producing multiple rich annotation types. In Section 7.3 we outline a a general architecture that applies to structured labeling tasks as well as classification, and generalizes a number of existing crowdsourcing models. In Section 7.4 we instantiate this general architecture for document classification and derive mean-field inference for the resulting model. In Section 7.5 we demonstrate that rich annotations allow our model to dramatically reduce annotation costs for a sentiment classification task compared with traditional crowdsourcing models that use only label evidence. Finally, in Section 7.6 we present active measurement selection for this class of models and demonstrate the process of formulating a novel measurement type that leverages document embeddings.

7.2 Background on Measurements

Liang et al. [74] first introduced the learning from measurements framework as a Bayesian framework for learning from multiple types of evidence in the context of supervised machine learning with exponential family models. Central to the measurements framework is the concept of a measurement feature $\sigma_k : \mathcal{X}, \mathcal{Y} \mapsto \mathcal{R}$. The measurement feature σ_k tests whether property k holds for the pair x, y where x is a data item such as a sentence or document and y is its (possibly structured) label. For example, some measurement feature $\tilde{k}$ might encode the judgment that the third document should be labeled as containing positive sentiment by always returning 0 except when those conditions

are met. Specifically, $\sigma_{\tilde{k}} = \mathbb{1}(x = x_3, y = Positive)$. Slightly more broadly, measurement $\hat{k}$ might encode the judgment that documents containing the word "lackluster" should be labeled as containing negative sentiment. $\sigma_{\hat{k}} = \mathbb{1}(\text{"lackluster"} \in x, y = Negative)$.

Thus measurement features map the data into an extremely sparse and high dimensional (partially) observed space. Furthermore, to enable measurements features to span multiple instances, each measurement feature is summed over the dataset $\sigma_k(x, y) = \sum_i \sigma_k(x_i, y_i)$. The learning from measurements framework defines K measurement features, one for every possible labeling. In the case of structured data, measurement functions are additionally summed over the internal structure of each instance; for example, when y is a sequence of sequence tags for sentence x then $\sigma_k(x, y) = \sum_i \sum_t \sigma_k(x_{it}, y_{it})$ where t iterates over positions in the sentence.

Measurement features capture a surprisingly rich variety of input types, including full and partial labels, labeled predicates (e.g., "documents containing the word 'unpleasant' tend to be negative" in sentiment classification), structure constraints (e.g., "the entity type NP-ORG should appear only once per earnings statement" in named entity recognition), relative label proportions (e.g., "verbs are more common than adverbs" in POS tagging), and so on. Refer to Liang et al. [74] for a description of some structured measurement features, and to Table 7.1 for a description of some classification measurement features.

The measurements framework treats observed measurement values τ as the result of measurement noise ϵ applied to the result of measurement features σ in the following generative story, illustrated in Figure 7.1.

1. Draw parameter vector θ.
2. Draw measurement noise prior γ.
3. For $i \in N$ instances:
    - Draw label y_i from conditional exponential model family $p(y_i|x_i, \theta)$.
4. For $k \in K$ measurements:

- Draw measurement noise ϵ_k from $p(\epsilon_k|\gamma)$.
- Draw measurement τ_k from $p(\tau_k|\sum_i \sigma_k(x_i, y_i), \epsilon_k)$.

Note that although document labels y are drawn as a part of the hypothetical generative story according to this model, at inference time y is always unobserved (see Figure 7.1). A basic insight that the learning from measurements framework shares with many crowdsourcing models is that while the true labels y cannot be observed directly, some byproduct τ of y can be observed and provide the evidence necessary for inferring y.

Although the measurements framework provides a probabilistically coherent context in which to incorporate diverse kinds of input, it has previously been unsuitable for dealing with crowdsourced measurements for two reasons. First, the framework's structure gives no guidance as to how measurement noise should be modeled, with the result that in practice measurement noise has previously been entirely ignored, reflecting an assumption that all measurements come from a single, trusted source [74]. If each observed measurement were given its own noise distribution, there would be far too many noise parameters to learn from the data. However, there has previously been no good basis for tying noise parameters. Second, Liang et al. [74] were principally focused on generalization performance via inferred model parameters θ. Accordingly, their inference procedure analytically integrates out y. By contrast, a primary objective in crowdsourcing is to infer unobserved y values. We address the first issue in Section 7.3 by grouping measurement noise by annotator, and the second in Section 7.4 by introducing a novel inference algorithm that explicitly infers distributions over y.

7.3 Multi-annotator Measurements Architecture

In this section we extend the learning from measurements framework described in Section 7.2 to accommodate multiple untrusted annotators. We do this by replicating each measurement k for each annotator j.

The new generative story, illustrated in Figure 7.1 becomes

1. Draw parameter vector θ.

2. Draw measurement noise prior γ.

3. For $i \in N$ instances:

   - Draw label y_i from conditional exponential model family $p(y_i|x_i, \theta)$.

4. For $j \in J$ annotators:

   - For $k \in K$ measurements:

     – Draw measurement noise ϵ_{jk} from $p(\epsilon_{jk}|\gamma)$.

     – Draw measurement τ_{jk} from $p(\tau_{jk}|\sum_i \sigma_{jk}(x_i, y_i), \epsilon_{jk})$.

The adapted framework accommodates multiple untrusted annotators by ensuring that each annotator j has their own copy of each measurement feature σ_{jk} parameterized by annotator-specific noise parameter ϵ_{jk}. In addition, we have added a hierarchical prior γ over noise parameters to reflect that in many cases it makes sense for a crowdsourcing model to impose structure such as parameter tying among measurement noise distributions. For example, we can tie all ϵ_{jk} for annotator j and be left with a single noise parameter ϵ_j per annotator by imposing a prior where $\forall j \exists k, k'(\epsilon_{jk} \neq \epsilon_{jk'}) \implies p(w|\gamma) = 0$. Similarly, if we tie all $\epsilon_{jk}, \epsilon_{j'k}$ across annotators j and j', then we recover the original measurements framework. It could rightly be argued that our changes to the measurements framework are more conceptual than technical; the K measurements in the original framework could be defined so as to replicate the structure in Figure 7.1. However, we feel that the conceptual shift is important. Making untrusted knowledge sources (e.g., human annotators) a first-class component of the measurements framework gives direction as to how to define useful noise distributions based on annotator identity.

The architecture in Figure 7.1 defines a framework rather than a model because it leaves unspecified design decisions such as the structure of y, the conditional exponential family used to model $p(y|x, \theta)$, and definition of the distributions $p(\theta)$ and $p(\epsilon_{jk})$. Because of the flexibility afforded by these choices, this framework generalizes a number of existing crowdsourcing models.

For example, if y is a discrete class label, $p(y|x,\theta)$ is a multinomial distribution that ignores x, and ϵ_j is defined to be a confusion matrix of Dirichlet-distributed rows whose component at position k, k' is tied to all measurement features that encode annotator j seeing an instance with true label k and calling it k', then this framework recovers a Bayesian version of the item-response crowdsourcing model [30]. Using the same settings but defining $p(y|x,\theta) \propto exp[\theta^T f(x,y)]$, then we recover a common data-conditional crowdsourcing model [40, 110, 140].

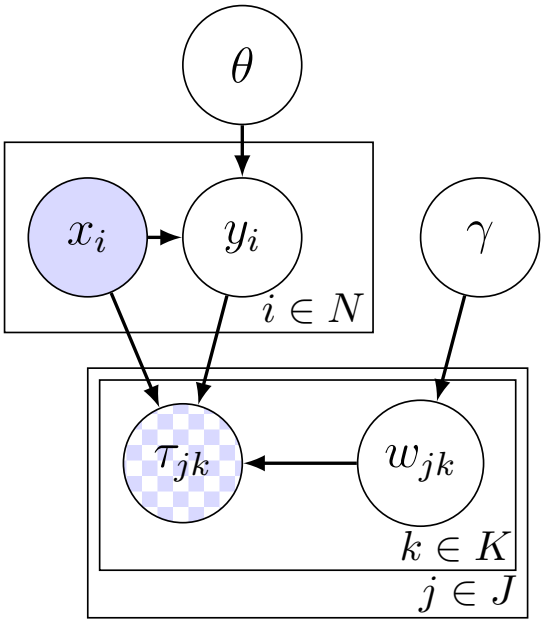


Figure 7.1: Plate Diagram depiction of the measurement framework of Liang et al. [74] adapted to handle multiple untrusted annotators. Shaded nodes have observed values. Partially shaded nodes have some observed values.

7.4 Per-annotator Normal Measurement Model for Classification

In this section we present a basic instantiation of the framework from Section 7.3 that will be used in experiments in Sections 7.5 and 7.6. We address a simple classification problem, so y is not structured. We choose $p(y|x,\theta) = p(y|\theta)$ to be a categorical distribution. Finally, we model measurement noise using per-annotator normal distributions. For brevity, we will refer to this model as the PAN (per-annotator normal) measurement model. The plate diagram for the PAN model is in Figure 7.2, and the generative story is summarized below.

1. Draw a stochastic vector θ over C classes from a symmetric Dirichlet $SymDir(\delta)$

2. For $i \in N$ documents, draw label y_i from categorical $Cat(\theta)$.

3. For $j \in J$ annotators:

   - draw measurement noise ϵ_j from inverse gamma $IG(\alpha, \beta)$.
   - For $k \in K$ measurements, draw τ_{jk} from a normal $Norm(\sum_i \sigma_{jk}(x_i, y_i), \epsilon_j)$

The PAN model's conditional log joint distribution is

$$\begin{aligned} &\log p(\theta, y, w, \tau | x) = \\ &- \log B(\delta) + J\alpha \log \beta - J \log \Gamma(\alpha) \\ &\quad - \sum_j \frac{K_j}{2} \log(2\pi) + \sum_c (\delta + n_c - 1) \log \theta_c \\ &\quad + \sum_j (-(\alpha + \frac{K_j}{2}) - 1) \log \epsilon_j \\ &\quad - \left(\frac{\beta + \frac{1}{2} \sum_k (\tau_{jk} - \sum_i \sigma_{jk}(x_i, y_i))^2}{\epsilon_j} \right) \end{aligned} \tag{7.1}$$

where $B(\cdot)$ is the multivariate beta function and K_j is the number of measurements values observed by annotator j. We employ mean-field variational inference as an efficient approximation to true inference in this model by assuming a fully factorized approximate model

$$\begin{aligned} &q(\theta, y, w | \nu) = \\ &\quad q(\theta | \nu^{(\delta)}) \prod_i q(y_i | \nu_i^{(y)}) \prod_j q(\epsilon_j | \nu_j^{(\alpha)}, \nu_j^{(\beta)}) \end{aligned} \tag{7.2}$$

and then minimizing $KL(q||p)$ via coordinate ascent by iteratively updating the variational parameters ν. That is, $q(\theta|\nu^{(\delta)}) \sim SymDir(\nu^{(\delta)})$ where

$$\nu_c^{(\delta)} = \delta + \sum_i \nu_{y_i, c}^{(\delta)}. \tag{7.3}$$

Similarly, $q(\epsilon_j|\nu_j^{(\alpha)}, \nu_j^{(\beta)}) \sim IG(\nu^{(\alpha)}, \nu^{(\beta)})$ where

$$\nu_j^{(\alpha)} = \alpha + \frac{K_j}{2} \tag{7.4}$$

and

$$\nu_j^{(\beta)} = \beta + \frac{1}{2} \sum_{k \in S(j)} E_{y_i} \left[\left(\sum_i \sigma_{jk}(x_i, y_i) \right)^2 \right] \tag{7.5}$$

where $S(j)$ is the set of measurements provided by annotator j. Although the term $E_{y_i}\left[(\sum_i \sigma_{jk}(x_i, y_i))^2\right]$ appears intractable, we can simplify it by introducing terms allowing us to complete the square.

$$\begin{aligned} &E_{y_i}\Big[\big(\sum_i \sigma_{jk}(x_i, y_i)\big)^2\Big] = \\ &\sum_i \Big(E_{q(y_i)}\Big[\sigma_{jk}(x_i, y_i))\Big]\Big)^2 \\ &- \sum_i E_{q(y_i)}\Big[\sigma_{jk}(x_i, y_i)\Big]^2 + E_{q(y_i)}\Big[\sigma_{jk}(x_i, y_i)^2\Big]. \end{aligned} \tag{7.6}$$

Finally, $q(y_i|\nu_i^{(y)}) \sim Cat(\nu_i^{(y)})$. We update $\nu_i^{(y)}$ by evaluating $\log \nu_i^{(y)} + const$ for each setting of y_i and then exponentiating and normalizing the resulting vector.

$$\begin{aligned} \log \nu_i^{(y)} &= \psi(\nu_{y_i}^{(\delta)}) - \psi(\sum_{y_i} \nu_{y_i}^{(\delta)}) + const \\ &+ \sum_j \frac{\nu_j^{(\alpha)}}{2\nu_j^{(\beta)}} \Bigg(\sum_{k \in S(i,j)} 2\tau_{jk}\sigma_{jk}(x_i, y_i) \\ &- \sigma_{jk}(x_i, y_i)^2 \\ &- 2\sigma_{jk}(x_i, y_i) \sum_{i' \neq i} E_{q(y_{i'})}\left[\sigma_{jk}(x_{i'}, y_{i'})\right] \Bigg) \end{aligned} \tag{7.7}$$

where $S(i, j)$ is the set of measurements provided by annotator j that relate to instance i. More formally, $S(i, j)$ is the set of measurements k where there is some setting of y_i that makes the measurement feature $\sigma_{jk}(x_i, y_i)$ evaluate to a non-zero value.

7.4.1 Implementation Considerations

The updates in the previous section contain terms like $\sum_j \sum_k \sum_i E_{q(y_i)}[\sigma_{jk}(x_i, y_i)]$. These terms must be calculated for each of $O(N)$ variables, resulting in a naïve inference complexity of $O(J \times K \times N^2)$ for each full iteration of variational parameter updates where J is the number of annotators, K is the number of measurements, and N is the size of the corpus. The N^2 factor is particularly problematic since datasets may grow large. Fortunately, many of these sums are sparse. For one thing, most annotators provide only a small subset of the measurements they could provide. This reduces the $\sum_k$ to $\sum_{k \in S(j)}$ or $\sum_{k \in S(i,j)}$ depending on context. In addition, the innermost sum $\sum_i$ is often sparse since measurement features are non-zero over only a limited subset of the dataset. In particular, simple label measurement features (see Table 7.1) may be calculated using only a single i, resulting in runtime complexity of $O(J \times K \times N)$. Other functions like labeled predicates depend only on subsets of the dataset. Even when measurements involve the entire dataset, like label proportions measurements, many values inside of the update equations may be cached and re-used with incremental updates in subsequent calculations.

Another important implementation consideration has to do with the scale of measurements. In Equation 7.1, observed measurement values τ are compared with the value of measurement features summed over the dataset $\sum_i \sigma(x_i, y_i)$. This latter quantity is bounded by a different range for each measurement feature (see Table 7.1). However, humans will usually prefer to specify proportions normalized to between 0 and 1, where 1 means "this happens as often as possible." Therefore it is important to scale human-specified measurements to the appropriate range, substituting $\max(\sigma_{jk}) \times \tau_{jk}$ for τ_{jk}. A related point is that each measurement type is defined on a different scale, but the PAN model fits a common variance to all types. For this to make sense, it is necessary to re-scale each measurement to a common range before running inference. For all experiments reported in this paper, we choose the range $[0..1]$. Concretely, substitute $\frac{\sigma_{jk}(x_i,y_i)}{\max(\sigma_{jk})}$ for $\sigma_{jk}(x_i, y_i)$, $\frac{E_{q(y_i)}[\sigma_{jk}(x_i,y_i)]}{\max(\sigma_{jk})}$ for $E_{q(y_i)}[\sigma_{jk}(x_i, y_i)]$, and $\frac{\tau_{jk}}{\max(\sigma_{jk})}$ for τ_{jk}. After applying both transformations, measurement values τ experience no net change.

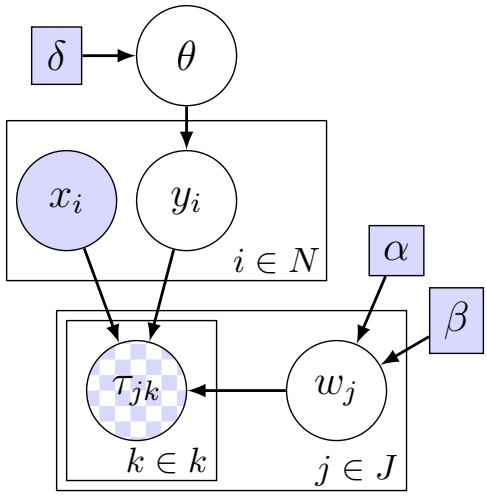


Figure 7.2: Plate Diagram depiction of the per-annotator normal (PAN) measurement model for classification. Round nodes are variables with distributions. Rectangular nodes are hyperparameters (without distributions).

7.5 Experiments

Our primary experimental goal in this section is to demonstrate the potential of the general learning from measurements crowdsourcing framework presented in Section 7.3 via its instantiation in the PAN model presented in Section 7.4. In addition to being interesting in its own right, the preliminary success of the PAN model (which is nearly the simplest possible instantiation of the general architecture) suggests that there is plenty of room for additional improvement using more interesting, possibly hierarchical measurement noise structures.

7.5.1 Baselines

Previous work in establishing benchmark crowdsourcing tasks indicates that there is no clear winner among crowdsourcing algorithms [119]. Multiply-annotated datasets can vary along so many dimensions—annotator reliability, redundancy per annotator, number of annotators, task difficulty, and task structure to name a few—that no single algorithm performs well on every dataset. Besides underscoring the need for a general and highly adaptable crowdsourcing framework, this also means that there is no dominant state-of-the-art to compare against. We choose two common and competitive baselines.

Majority Vote (MV). This algorithm chooses the label with the largest number of votes for each item, ignoring annotator identify. Ties are broken randomly. Although simple, majority vote is widely used and surprisingly successful across a variety of tasks.

Dawid & Skene (DS). Dawid and Skene [30] model annotator error over discrete responses via per-annotator confusion matrices. A large body of subsequent work uses the same basic structure, making it a good point of comparison. We use a Bayesian version of this model with variational inference adapted from [40]. As noted in Section 7.3, the general learning from measurements crowdsourcing framework generalizes this model and many of its extensions.

7.5.2 Simulated Data

We first generate confidence in our implementation by running on simulated annotator measurements that largely conform to PAN's Gaussian noise assumptions. Document label annotations are simulated by corrupting the true labels (according to the existing corpus) via confusion matrices for 5 annotators with 50%, 55%, 60%, 65%, and 70% accuracy rates, respectively. Labeled predicates are simulated by choosing a word ϵ and document label y uniformly at random, and then calculating the empirical rate of seeing documents with label y given that they contain word ϵ. We add Gaussian noise proportional to the total number of documents containing word ϵ and inversely proportional to the accuracy of the annotator. Label proportions are relatively easy to specify manually, and we do so adding 20 measurements that encode our belief each class appears roughly the same number of times N/C.

Figure 7.3 shows that PAN is able to bring together disparate measurement types to outperform baselines on the 20 Newsgroups dataset. *PAN10* shows performance with 10,000 labeled predicates. *PAN5* shows performance with 5,000 labeled predicates. For reference, performance is also shown with no labeled predicates *PAN*. In all cases we plot the accuracy of the labels inferred by each algorithm against the total number of simulated document label annotations. Annotation order is random. The question of which measurements to use in practice is addressed in Section 7.6.1. Documents are assigned labels randomly without replacement until each document in the

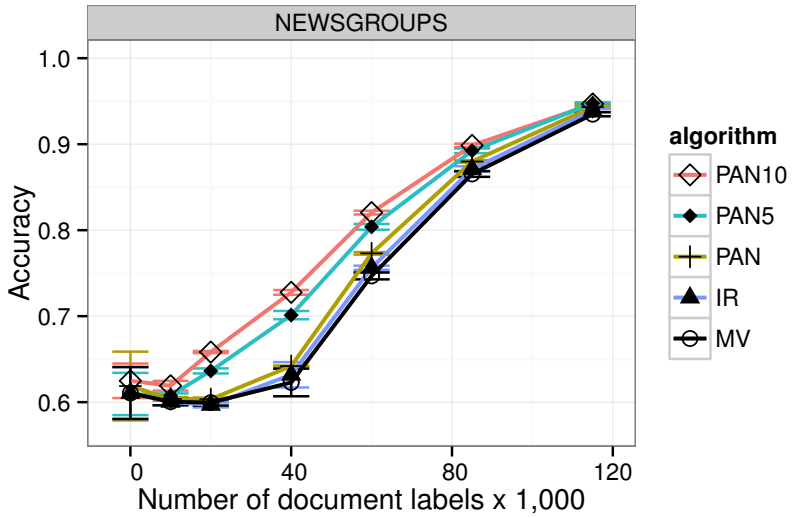


Figure 7.3: Inferred labeled accuracy of Majority vote (MV), the item-response (IR) model, and the per-annotator normal (PAN) measurement model as the corpus is annotated until each document has approximately 6 document label annotations. PAN performance is shown with 5,000 (*PAN5*), and 10,000 (*PAN10*) additional predicate labels.

corpus is labeled once, after which the process is repeated until at the final point each of the 20,000 newsgroups documents has approximately 6 document label annotations. Thus at each horizontal location on the learning curve the baselines and PAN have access to the same number of document label annotations, but PAN makes use of additional measurement supervision. Unsurprisingly, there seems to be a diminishing returns property associated with additional labeled predicates. The difference between 0 and 5,000 labeled predicates is dramatic; the difference between $5,000$ and $10,000$ less so.

7.5.3 Sentiment Classification

We verify that the same trends hold when working with real annotator judgments using the "Weather Sentiment" dataset available at `http://www.crowdflower.com/data-for-everyone`. In this dataset 20 annotators labeled 1,000 tweets as either "Positive," "Negative," "Neutral," or "Not related to the weather." Majority vote labels were then evaluated in the related "Weather Sentiment Evaluated" task where an additional 10 annotators judged whether each consensus label was correct

or not. We use as a gold standard set the 724 consensus labels which 9 or more evaluation task annotators judged to be correct.

Annotations are already available for this dataset, but no labeled predicates or label proportion judgments. We paid CrowdFlower workers to generate labeled predicates by showing them groups of 10 randomly selected weather tweets and asking them to list words that characterize each class of tweet. Although a more highly trained workforce could have generated and labeled more sophisticated predicates such as regular expressions, we considered that labeled words were sufficient as a first test. Furthermore, we encode an *a priori* belief that each class occurs roughly the same number of times by manually adding 4 trusted label proportion measurements stating that each class occurs $N/C = 250$ times. Trusted measurements are expressed in this framework by defining an annotator with a strong prior assigning them little measurement noise, and authoring measurements under that trusted annotator's id.

We gathered 864 word lists with 2,482 total word tokens. Of the 2,482 resulting labeled predicates, 216 did not match any words in the corpus (we allowed users the freedom to draw on their own background knowledge as well as words of the corpus that they were shown). Among all annotators, 995 unique words were used. Interestingly, a brief manual examination of the word lists did not uncover any abusive behavior, although several annotator clearly wanted to match phrases rather than just words. Our data and processing scripts are available via git.[1] We matched these words to document words after removing punctuation, converting to lower case, and running a Porter stemmer over both, and additionally removed words from the standard MALLET stopword list [85].

Figure 7.4 shows that labeled predicates have a positive effect on PAN, resulting in performance significantly superior to the approaches that were unable to take advantage of this additional information. While it is true that labeled predicates are not free, recall that whereas it took 20,000 judgments to generate the annotation data in these experiments ($200 at $0.01 per judgment), it took only 864 judgments to generate our labeled predicates (about $9 at $0.01 per judgment).

[1] git://nlp.cs.byu.edu/plf1/crowdflower-weather-measurements.git

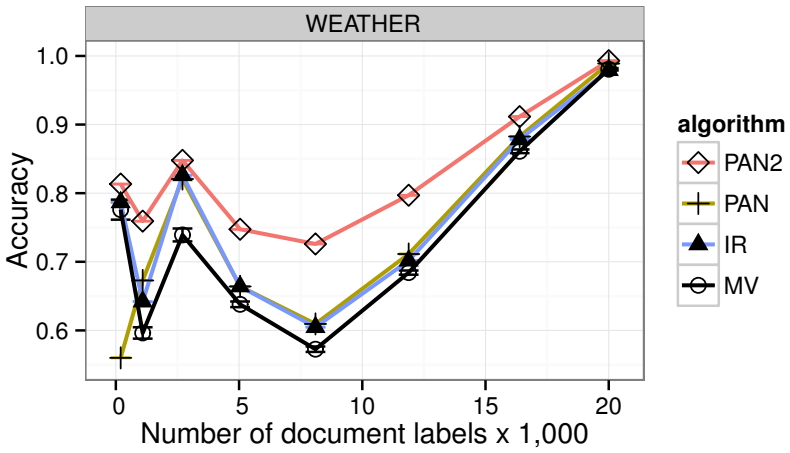


Figure 7.4: Inferred labeled accuracy of Majority vote (MV), the item-response (IR) model, and the per-annotator normal (PAN) measurement model. PAN is shown with and without access to 2,266 labeled predicates from CrowdFlower.

7.6 Model Extensions

In this section we describe two extensions to the general framework introduced in Section 7.3. First, we present an algorithm for active measurement selection. Second, we demonstrate the ease of formulating novel measurements. Although both of these extensions are general to any model in the framework, we make them concrete by applying them to the PAN model.

7.6.1 Active Measurement Selection

In Section 7.5 we assumed that we first obtained labeled predicates and proportions in a lump at the beginning, and then received ordered document label annotations either randomly (in Section 7.5.2) or else in the largely arbitrary empirical order determined by their creation timestamp (in Section 7.5.3). Active learning answers the question of which instances should be labeled in order to optimize some criterion. Traditionally, active learning has been posed as the problem of choosing instances to label in the framework of supervised machine learning, assuming a single infallible annotator [117]. A fair amount of work addresses active learning in the context of multiple

annotator [31, 97, 139]. However, we are unaware of any previous work that simultaneously selects an annotator and one of a number of different measurement types.

Fortunately, an active measurement selection algorithm that is grounded in decision theory is already defined by Liang et al. [74] and requires only minor adaptations to fit the crowdsourcing scenario. Algorithm 2 reports the adapted pseudocode where $p(\tau_{jk}|\tau_0, q_0)$ uses the approximate posterior q_0 to calculate a distribution over τ_{jk} using the structure of the original model p. For example, for PAN $p(\tau_{jk}|\tau_0, q_0) = Normal(\sum_i \sigma_{jk}(x_i, y_i), \frac{\nu^{(\beta)}}{\nu^{(\alpha)}-1})$. Notice that we use the mean value $\frac{\nu^{(\beta)}}{\nu^{(\alpha)}-1}$ of our posterior annotator noise distributions. In the absence of computational constraints, one could be more correct and stochastically integrate over variance values by sampling from $IG(\nu^{(\alpha)}, \nu^{(\beta)})$. Finally, the term $U_{\tilde{q}}(\sigma, \tau)$ is expected net utility approximated as $E_{p*(X)} \max_{\hat{Y}} E_{\tilde{q}(Y)}[r(Y, \hat{Y})] - C(\tau)$ where $r(Y, \hat{Y})$ is a reward function like label accuracy, $C(\tau)$ is the expected cost of observing measurements τ, $p*$ is the empirical distribution in a dataset and assuming variational inference, and $\tilde{q}(y)$ approximates $p(y|\tau, X)$. Concretely, with a label accuracy reward function and constant cost function, the final expected utility is $\sum_i \max_{\hat{y}} q(y_i = \hat{y}) = \sum_i \max_{\hat{y}} \nu^{(y)}_{i\hat{y}}$.

By default, Algorithm 2 jointly select an annotator j and measurement k, but it can also be used in other ways. Haertel et al. [52] argue that in realistic active sample selection scenarios the active learning algorithm typically cannot control when annotators are available, but rather must respond to annotator requests for work. Algorithm 2 returns the best measurement k given annotator $\hat{j}$ by returning $\text{argmax}_k \mu_{\tau_{\hat{j}k}}$. Similarly, one can calculate the best annotator j for a desired measurement $\hat{k}$ as $\text{argmax}_j \mu_{\tau_{j\hat{k}}}$.

Unfortunately, Algorithm 2 is computationally expensive. During every measurement selection, each candidate measurement must be considered and a model retrained using t different speculative candidate measurements. However, by applying a number of additional approximations we were able to run on the weather sentiment dataset from Section 7.5.3 with over 22,000 candidate measurements. We set the number of samples t to 3, and models trained in the inner loop (line 13 of Algorithm 2) are initialized using q_0 and then trained only one additional iteration. More importantly, we scored only 25 randomly selected candidates at each round and selected batches

of the 10 most promising measurements at each round. In Figure 7.5 we use the measurements from the sentiment classification experiment in 7.5.3, selecting them both randomly as well as using Algorithm 2. The active measurement selection strategy shows marked improvement over a random baseline even with aggressive efficiency settings.

Algorithm 2 Decision-theoretical active measurement selection algorithm of Liang et al. [74] adapted to the crowdsourcing scenario (and notationally assuming variational inference with approximate posterior q).

function MEASUREMENTSELECTION(τ_0)
  $q_0 \leftarrow$ Inference(τ_0)
  while more measurements are desired **do**
    $j, k \leftarrow$ NextMeasurement(τ_0, q_0)
    $\tau_0 \leftarrow \tau_0 \cup$ ObserveMeasurement(j, k)
    $q_0 \leftarrow$ Inference(τ_0)
  return q
function NEXTMEASUREMENT(τ_0, q_0)
  for annotator j **do**
    for candidate measurement j, k **do**
      draw t samples from $p(\tau_{jk}|\tau_0, q_0)$
      for sampled τ_{jkt} **do**
        $\tilde{q} \leftarrow$ Inference($\tau_{jkt} \cup \tau_0$)
        $\mu_{\tau_{jkt}} \leftarrow U_{\tilde{q}}(\tau_{jkt} \cup \tau_0)$
      $\mu_{\tau_{jk}} \leftarrow \frac{1}{t} \sum_t \mu_{\tau_{jkt}}$
  return $\operatorname{argmax}_{j,k} \mu_{\tau_{jk}}$

7.6.2 Labeled Location Measurements

One of the advantages of using the learning from measurements framework is that it is trivial to formulate a new measurement type and add it to the model without changing existing inference. Measurements may be based on X or any supervised or unsupervised transform of X, including POS tags, parse trees, semantic frames, named entities, topics, clusters, or vector space locations. We demonstrate this by defining a novel measurement we call labeled locations based on vector-space document embeddings. The idea is that rather than attaching labels to documents in a sparse high dimensional space (where neighborhoods are largely meaningless), instead we will label locations in a dense low dimensional vector space where neighborhoods have semantic meaning,

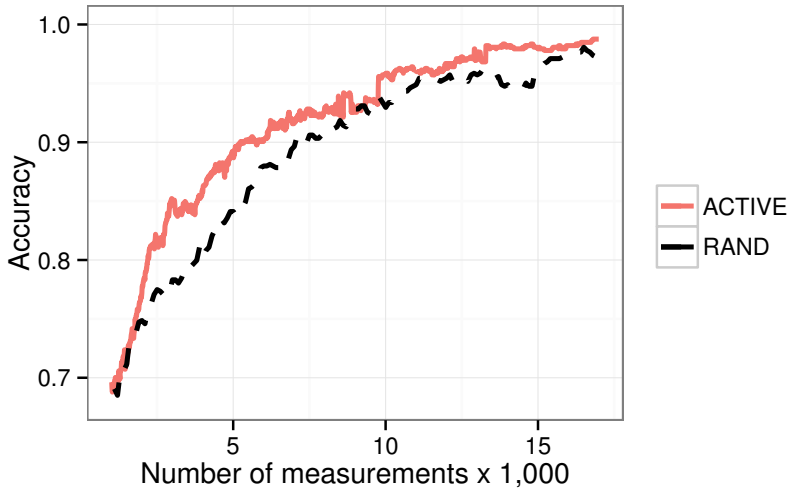


Figure 7.5: Inferred labeled accuracy of the per-annotator normal measurement model selecting measurements randomly (*RAND*) compared with a strategy that selects measurements and annotators actively (*ACTIVE*).

and allow these labeled locations to inform neighboring documents. A measurement k encoding the fact that location v is assigned label c is defined as $\sigma_{jk}(x_i, y_i) = \cos(v, x_i)\mathbb{1}(y = c)$ so that each document is expected to have label c with strength proportional to its cosine distance to location v. The expected value of this measurement is $\sum_i E_{q(y_i)}[\sigma_{jk}(x_i, y_i)] = \sum_i \cos(v, x_i)q(y_i = c)$ and $\max(\sigma_{jk}) = \sum_i \cos(v, x_i)$. This measurement can be thought of as a soft labeled predicate where, rather than selecting a subset of instances of label, the predicate matches all instances with varying strength, selecting more strongly those that are more related to labeled location. In order to reduce computational requirements, labeled locations may be modified to ignore all but the top K neighbors: $\sigma_{jk}(x_i, y_i) = \cos(v, x_i)\mathbb{1}(x_i \in \text{nearestK}(v))\mathbb{1}(y = c))$.

Figure 7.6 demonstrates the result of interpreting existing annotations as labeled locations in an embedded document vector space using 20 nearest neighbors. Documents are embedded by summing word embeddings produced by a word2vec model trained on Wikipedia [87]. To encourage word matches across Wikipedia and Twitter language, we normalize tweets before looking up words using the Twitter normalization dictionary of Han et al. [54].

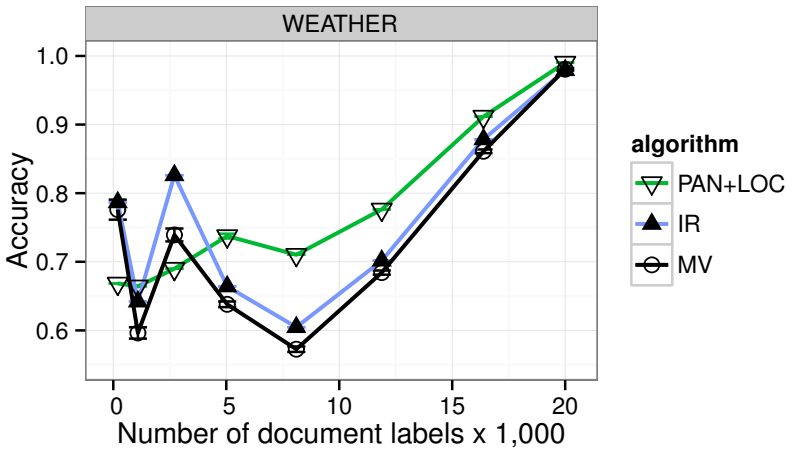


Figure 7.6: Inferred labeled accuracy of Majority vote (MV), the item-response (IR) model, and the per-annotator normal (PAN) measurement model. In this plot, PAN treats annotations as labeled locations in a vector space induced by a word2vec embedding model trained on Wikipedia.

7.7 Additional Related Work

Other supervised learning frameworks that incorporate rich prior knowledge include constraint-driven learning based on integer linear programming [20], generalized expectation criteria [32], and the posterior regularization [46]. Ganchev et al. [46] explain each of these three frameworks can be derived as a special case of the learning from measurements framework of Liang et al. [74] by making particular approximations for the sake of tractability.

Traditional corpus construction assesses inferred label quality using annotator agreement heuristics such as Krippendorffs alpha [65]. Passonneau and Carpenter [104] argues that inference in probabilistic models in general yields higher quality labels at lower cost, and should be preferred over agreement heuristics. A number of notable crowdsourcing models exist. Hovy et al. [57] include Bernoulli switching variables to identify and eliminate malicious contributors (spammers). Raykar and Yu [109] iteratively run inference and exclude problematic annotators in order to eliminate spammers. Raykar et al. [110], Yan et al. [140], and Felt et al. [39] model data jointly with labels, allowing patterns in the data to inform inferred labels. Simpson and Roberts [120] model annotator dynamics, tracking the ways that annotator decision making changes over time in

response to factors such as training and fatigue. Welinder et al. [135] and Whitehill et al. [136] both model item difficulty, reducing the effect of inherently ambiguous or difficult items on annotator reliability estimates.

Each of these crowdsourcing models focuses on incorporating one or more insights about the annotation process. We leave it to future work to incorporate these insights into crowdsourcing models that learn from measurements, either via measurement noise or via new measurement formulations. For example, item difficulty could be modeled by creating a measurement feature $\sigma_{jk}(x_i = x_{i'})$ for each $x_{i'}$ and assigning a separate measurement noise to each (perhaps with a shared hierarchical prior noise).

7.8 Conclusion and Future Work

We have described a framework for learning from measurements in crowdsourcing models that builds on compelling but under-used previous work [74]. This framework allows the principled combination of rich input types from multiple imperfect annotators, and therefore enables principled active learning algorithm that simultaneously selects annotators as well as measurement types. This framework generalizes several common crowdsourcing models.

We showed promising results using one of the simplest possible instantiations of this framework with per-annotator Gaussian measurement noise. The success of this simple model suggests great potential for future improvement with more sophisticated measurement noise priors. For example, Gaussian Processes could be used as measurement noise priors to relax the assumption of symmetric normal noise. Also, the noise associated with each measurement type could be modeled separately, connected via hierarchical annotator reliability priors. Finally, a large number of existing insights in the crowdsourcing literature will fit into this framework, including using exponential family distributions over $p(y|x)$ to inform inferred labels and modeling item difficulty. Finally, we demonstrated the simplicity of the adding a new measurement type to one of these models by formulating a measurement that labels locations in a semantically meaningful vector

space. This new measurement type has the potential to improve performance on existing annotation datasets by allowing labeled instances to influence their neighbors.

Chapter 8

Conclusions and Future Work

This dissertation provides tools and insights to greatly reduce the cost required to label a new corpus. In Chapter 2 we showed how transfer learning can be used to incorporate information from outdated annotation generated under old annotation schemes in order to improve pre-annotation on current documents and thus reduce annotation costs. In Chapters 3, 4, and 5 we identified the fact that annotation aggregation models with generative data components (models that jointly explain observed data and annotations) have several important advantages over their traditionally more popular conditional counterparts. This line of work culminated in a state-of-the-art annotation aggregation algorithm, CSLDA. In Chapter 6 we presented an approach for incorporating data conditionally when a generative data component is unavailable. We demonstrated that data-conditional annotation aggregation models using text embedded in semantically meaningful vector-spaces greatly outperform similar models using conventional text representations. Finally, in Chapter 7 we identified a family of aggregation models capable of learning from diverse annotation types. We also demonstrated for the first time the viability of a multiannotator active learning algorithm that jointly selects an annotator, a data item, and a measurement type. Modular Java code for all models is available via git by checking out the following inter-dependent projects:

- `git://nlp.cs.byu.edu/aml/byunlp/Utilities.git`
- `git://nlp.cs.byu.edu/aml/byunlp/StatnlpProjects/Classification.git`
- `git://nlp.cs.byu.edu/aml/byunlp/StatnlpProjects/Crowdsourcing.git`
- `git://nlp.cs.byu.edu/aml/byunlp/StatnlpProjects/ActiveLearning.git`

These chapters suggest a wealth of future work. Chapter 2 was a first step in dealing with annotation scheme revisions. Follow-on work would involve working with large-scale practical annotation projects in order to gather data tracking actual annotation scheme revisions along with sets of out-dated annotations. This data would inform new algorithms, perhaps based on work in chapters 3-7. Indeed, we imagine that the measurement framework presented in Chapter 7 could be used to unify the information from annotations produced over multiple revisions. For example, if class label *A* were being split into *B* and *C*, a labeled predicate could be formulated to enforce the semantics, "if a document previously was annotated A, then the new label should be either B or C."

Also in future work, the crowdsourcing models from Chapters 3-7 could be enhanced to operate on structured labels such as sequences and trees. Similarly, this dissertation deals exclusively with text data, but it would be interesting to see which methods generalize well to annotation tasks on different kinds of media, such as image segmentation or video annotation. The measurements framework could be ported to image classification by identifying a few informative image-centric predicates based on existing image feature extraction techniques, but it is less clear how it could be applied to object identification within an image. Other potential future work includes enhancing the models from Chapters 3-7 to suggest new classes when appropriate by adding non-parametric priors over the classes.

Parallel to all of the modeling work explored in this dissertation, there is the important question of how best to help humans to interact with these models. Experiments in this work assume a very basic unidirectional workflow in which humans produce annotations and then the model aggregates those annotations. This process is iterated until inferred label quality reaches an acceptable level according to some heldout gold standard set. Relatively little effort to date has been made to formalize and explore issues that arise as the result of multiple humans interacting with an annotation system. For example, the annotation system could attempt to decrease annotation effort by prompting annotators with its current inferred label distribution. Can this effect be used without introducing too much bias or allowing crowd workers to game the system? The system might also engage annotators in personalized training based on their inferred annotator error characteristics,

or flag potential problems with the annotation scheme based on common errors. In addition, in distributed collaborative projects, two or more valid competing schools of thought may lead to the data being annotated in conflicting ways. Existing annotation aggregation models can be adapted to map from their default consensus labels to labels reflecting the probably viewpoint of a particular annotator or group. Accomodating divergent annotation schemes could decrease the overhead associated with coordinating the efforts of independently motivated annotators. Finally, although excellent work has been done in modeling the way annotator behavior changes over time [120] and understanding the way crowd workers respond to payment incentives [56], more work remains to be done on these important subjects.

We feel that the work in Chapter 7 on crowdsourcing models that learn from rich, heterogeneous annotation types opens up a particularly interesting realm of potential future work. Nearly all of the insights driving existing crowdsourcing models—the desire to model the data, item difficulty, annotator error correlation, etc.—may all be added to models that learn from rich annotation types. Also, in the same way that using a model capable of learning from rich annotation types introduced an additional dimension into multi-annotator active learning in Chapter 7, these models add a new dimension to a series of important questions that arise in corpus annotation. For example, can we effectively estimate the costs of various annotation types during the course of an annotation project? What is the best strategy for paying crowdsourced workers when multiple annotation types are in play? Does the context switching associated with changing annotation types help reduce fatigue or boredom among contributors, or does it increase cognitive load and hurt performance? Answers to these kinds of questions will have the potential for great impact on a variety of real-world annotation projects.

Chapter 9

Appendix: Supplementary Material for *Early Gains Matter: A Case for Preferring Generative over Discriminative Crowdsourcing Models*

Derivation of mean-field variational updates

Introduction. Here we derive mean field variational updates for MOMRESP. Although this derivation is largely a mechanical exercise, it is our belief that there is a contingent of crowdsourcing practitioners whose background is more practical than theoretical and who may appreciate seeing the mechanics of mean-field variational inference presented in a high level of detail for a model they are familiar with. The updates for LOGRESP involve so much overlap with those for MOMRESP that we leave them as an exercise for the interested reader.

Problem Setup. Given some posterior distribution p^* over variables $\boldsymbol{z}$, our goal is to search among some family of simpler approximate tractable models Q and identify the $q(\boldsymbol{z}) \in Q$ that most closely resembles $p^*(\boldsymbol{z})$. If we choose Q to be the set of fully factored models such that $q(\boldsymbol{z}) = \prod q_i(z_i)$ (the mean-field assumption) then the q that minimizes KL divergence $KL(q||p^*)$ can be shown to have the following form:

$$\log q_i(z_i) = E_{q_{-i}}[\log \tilde{p}] + \text{const} \tag{9.1}$$

where $\tilde{p}$ is the unnormalized posterior and $q_{-i} = \{q_{i'} : i' \neq i\}$. For a derivation of this property, see Chapter 21.3 of Murphy [93].

Notation. We adopt slightly different notation here than is used in the paper. Variables that represent non-scalars (e.g., vectors or matrices) after resolving subscripts are bolded. That is, we might use $\boldsymbol{z}$ to denote a matrix, $\boldsymbol{z}_i$ a vector, and z_{ij} a scalar. Furthermore, in order to clearly distinguish

among variational distributions each is given a unique distributional name rather than simply being distinguished by its arguments as in the main paper. Sums are simplified by providing only a single subscript. $\sum_i$ is short for $\sum_{i \in N}$, $\sum_j$ is short for $\sum_{j \in J}$, and so on.

Mean-Field Variational Update Equations

The unnormalized posterior $\tilde{p}$ required by Equation 9.1 is proportional to the full joint. Therefore we begin by writing out the full (unnormalized) joint according to MOMRESP by starting with Equation 4.3 from the main paper, plugging in distributional forms, and then simplifying by omitting constants and combining terms from conjugate distributions:

$$\tilde{p}(\boldsymbol{\theta}, \boldsymbol{\gamma}, \boldsymbol{\phi}, \boldsymbol{y} | \boldsymbol{x}, \boldsymbol{a}) \propto p(\boldsymbol{\theta}, \boldsymbol{\gamma}, \boldsymbol{\phi}, \boldsymbol{y}, \boldsymbol{x}, \boldsymbol{a}) \tag{9.2}$$

$$\propto \left(\prod_k \theta_k^{b_k^{(\theta)} + n_k^{(\theta)} - 1} \right) \left(\prod_j \prod_k \prod_{k'} \gamma_{jkk'}^{b_{jkk'}^{(\gamma)} + n_{jkk'}^{(\gamma)} - 1} \right) \left(\prod_k \prod_f \phi_{kf}^{b_{kf}^{(\phi)} + n_{kf}^{(\phi)} - 1} \right) \tag{9.3}$$

$$\cdot \left(\prod_i \prod_j \frac{|\boldsymbol{a}_{ij}|_1!}{\prod_k a_{ijk}!} \right) \left(\prod_i \frac{|\boldsymbol{x}_i|_1!}{\prod_f x_{if}!} \right)$$

where $n_k^{(\theta)} = \sum_i \mathbb{1}(y_i = k)$, $n_{jkk'}^{(\gamma)} = \sum_i a_{ijk'} \mathbb{1}(y_i = k)$, and $n_{kf}^{(\phi)} = \sum_i x_{if} \mathbb{1}(y_i = k)$. That is, $n_k^{(\theta)}$ is the number of instances labeled k, $n_{jkk'}^{(\gamma)}$ is the number of times that annotator j chose annotation k' on instances with true label k, and $n_{kf}^{(\phi)}$ is the number of times feature f occurs with instances having label k.

We next take the log of Equation 9.3 to get the unnormalized posterior $\tilde{p}$. Note that in this step the multinomial coefficients consituting the last two terms of Equation 9.3 are absorbed into the constant of proportionality because they are constant in the context of posterior inference of $p^*(\boldsymbol{\theta}, \boldsymbol{\gamma}, \boldsymbol{\phi}, \boldsymbol{y} | \boldsymbol{x}, \boldsymbol{a})$ where $\boldsymbol{a}$ and $\boldsymbol{x}$ are fixed and known.

$$\log \tilde{p} = \sum_k (b_k^{(\theta)} + n_k^{(\theta)} - 1) \log \theta_k + \sum_j \sum_k \sum_{k'} (b_{jkk'}^{(\gamma)} \gamma_j + n_{jkk'}^{(\gamma)} - 1) \log \gamma_{jkk'} \tag{9.4}$$

$$+ \sum_k \sum_f (b_{kf}^{(\phi)} + n_{kf}^{(\phi)} - 1) \log \phi_{kf}$$

Let the approximate distribution q be fully factored:

$$q(\boldsymbol{\theta}, \boldsymbol{\gamma}, \boldsymbol{\phi}, \boldsymbol{y}) = \pi(\theta) \prod_j \prod_k \nu(\gamma_{jk}) \prod_k \lambda(\phi_k) \prod_i g(y_i) \tag{9.5}$$

We derive the mean-field update equation for each factor in turn. The update equation will lead us to a concrete functional form for each variational distribution with appropriate variational paramters. The derivations make heavy use of the following five properties of expectations:

$$E_{p(x,y)}[af(x) + bg(y)] = aE_{p(x,y)}[f(x)] + bE_{p(x,y)}[g(y)] \quad \text{linearity} \tag{9.6}$$

$$E_{p(x,y)}[f(x)] = E_{p(x)}[f(x)] \quad \text{marginal} \tag{9.7}$$

$$E_{p(x)}[\mathbb{1}(x = k)] = p(k) \quad \text{delta} \tag{9.8}$$

$$E_{p(x,y)}[f(x)g(y)] = E_{p(x)}[f(x)]E_{p(y)}[g(y)] \quad \text{iff } p(x, y) = p(x)p(y) \quad \text{independence} \tag{9.9}$$

$$E_{p(x)}[K] = K \quad \text{constant} \tag{9.10}$$

Mean-field update and functional form for $\pi(\theta)$

$$\log \pi(\theta) = E_{\boldsymbol{\nu},\boldsymbol{g},\boldsymbol{\lambda}}[\log \tilde{p}] \tag{9.11}$$

$$= E_{\boldsymbol{\nu},\boldsymbol{g},\boldsymbol{\lambda}}[\sum_k (b_k^{(\theta)} + n_k^{(\theta)} - 1) \log \theta_k] + const \tag{9.12}$$

$$= \sum_k E_{\boldsymbol{g}}[(b_k^{(\theta)} + n_k^{(\theta)} - 1)] \log \theta_k + const \tag{9.13}$$

$$= \sum_k (b_k^{(\theta)} + E_{\boldsymbol{g}}[n_k^{(\theta)}] - 1) \log \theta_k + const \tag{9.14}$$

$$= \sum_k (b_k^{(\theta)} + \sum_i E_{g_i}[\mathbb{1}(y_i = k)] - 1) \log \theta_k + const \tag{9.15}$$

$$= \sum_k (b_k^{(\theta)} + \sum_i g_i(k) - 1) \log \theta_k + const \tag{9.16}$$

$$\pi(\theta) \propto \prod_k \theta_k^{b_k^{(\theta)} + \sum_i g_i(k) - 1} \tag{9.17}$$

$$= Dirichlet(\boldsymbol{\alpha}^{(\theta)}) \tag{9.18}$$

where $\alpha^{(\theta_k)} = b_k^{(\theta)} + \sum_i g_i(k)$.

Explanation. Equation 9.11 instantiates Equation 9.1 for π. Equation 9.12 plugs in the functional form of the posterior from Equation 9.4 and applies the linearity of expectations (Equation 9.6) to distribute the expectation over the sum. Terms not involving θ are absorbed into a constant. Equation 9.13 again applies the linearity of expectation over addition and multiplication and then applies the marginal property of expectations (Equation 9.7) so that the expectation is with respect to only the variational distribution $\boldsymbol{g}$ needed to compute the expectation of $n_k^{(\theta)}$. Equation 9.14 applies the linearity of expectations and simplifies the expectations of constants (Equation 9.10). Equation 9.15 substitutes the definition of $n_k^{(\theta)}$ and applies the linearity and marginal properties of expectations. Equation 9.16 simplifies expectations applied to delta functions (Equation 9.8). We then recognize the kernel of a log Dirichlet distribution.

The remaining derivations follow very similar lines of reasoning, so we only offer explanations where they differ from the patterns seen above. In particular, $\nu_{jk}(\gamma_{jk})$ and $\lambda_k(\phi_k)$ are nearly identical so no explanation will be given.

Mean-field update and functional form for $\nu_{jk}(\gamma_{jk})$

$$
\begin{aligned}
\log \nu_{jk}(\gamma_{jk}) &= E_{\boldsymbol{\nu}_{-jk},\boldsymbol{g},\boldsymbol{\lambda},\pi}[\log \tilde{p}] \\
&= E_{\boldsymbol{\nu}_{-jk},\boldsymbol{g},\boldsymbol{\lambda},\pi}[\sum_{j'}\sum_{k''}\sum_{k'}(b^{(\gamma)}_{j'k''k'} + n^{(\gamma)}_{j'k''k'} - 1)\log \gamma_{j'k''k'}] + const \\
&= \sum_{j'}\sum_{k''}\sum_{k'} E_{\boldsymbol{g}}[(b^{(\gamma)}_{j'k''k'} + n^{(\gamma)}_{j'k''k'} - 1)\log \gamma_{j'k''k'}] + const \\
&= \sum_{k'}(b^{(\gamma)}_{jkk'} + E_{\boldsymbol{g}}[n^{(\gamma)}_{jkk'}] - 1)]\log \gamma_{jkk'} + const \\
&= \sum_{k'}(b^{(\gamma)}_{jkk'} + \sum_{i} a_{ijk'} E_{g_i}[\mathbb{1}(y_i = k)] - 1)\log \gamma_{jkk'} + const \\
&= \sum_{k'}(b^{(\gamma)}_{jkk'} + \sum_{i} a_{ijk'} g_i(k) - 1)\log \gamma_{jkk'} + const \\
\nu_{jk}(\gamma_{jk}) &\propto \prod_{k'} \gamma_{jkk'}^{b^{(\gamma)}_{jkk'} + \sum_i a_{ijk'} g_i(k) - 1} \\
&= Dirichlet(\boldsymbol{\alpha}^{(\gamma)}_{jk})
\end{aligned}
$$

where $\alpha^{(\gamma)}_{jkk'} = b^{(\gamma)}_{jkk'} + \sum_i a_{ijk'} g_i(k)$.

Mean-field update and functional form for $\lambda_k(\phi_k)$

$$
\begin{aligned}
\log \lambda_k(\phi_k) &= E_{\boldsymbol{\nu},\boldsymbol{g},\boldsymbol{\lambda}_{-k},\pi}[\log \tilde{p}] + const \\
&= E_{\boldsymbol{\nu},\boldsymbol{g},\boldsymbol{\lambda}_{-k},\pi}\Big[\sum_{k'}\sum_{f}(b^{(\phi)}_{k'f} + n^{(\phi)}_{k'f} - 1)\log \phi_{k'f}\Big] + const \\
&= E_{\boldsymbol{g},\boldsymbol{\lambda}_{-k}}\Big[\sum_{f}(b^{(\phi)}_{kf} + n^{(\phi)}_{kf} - 1)\Big]\log \phi_{kf} + const \\
&= \sum_{f}(b^{(\phi)}_{kf} + E_{\boldsymbol{g}}[n^{(\phi)}_{kf}] - 1)\log \phi_{kf} + const \\
&= \sum_{f}(b^{(\phi)}_{kf} + E_{\boldsymbol{g}}[\sum_{i} x_{if}\mathbb{1}(y_i = k)] - 1)\log \phi_{kf} + const \\
&= \sum_{f}(b^{(\phi)}_{kf} + \sum_{i} x_{if} g_i(k) - 1)\log \phi_{kf} + const \\
\lambda_k(\phi_k) &\propto \prod_{f} \phi_{kf}^{b^{(\phi)}_{kf} + \sum_i x_{if} g_i(k) - 1} \\
&= Dirichlet(\boldsymbol{\alpha}^{(\phi)}_{k})
\end{aligned}
$$

where $\alpha_{kf} = b^{(\phi)}_{kf} + \sum_i x_{if} g_i(k)$

Mean-field update and functional form for $g_i(y_i)$

$$
\begin{aligned}
\log g_i(y_i) &= E_{\boldsymbol{\nu},\boldsymbol{g}_{-i},\boldsymbol{\lambda},\pi}[\log \tilde{p}] + const \\
&= E_{\boldsymbol{\nu},\boldsymbol{g}_{-i},\boldsymbol{\lambda},\pi}[\sum_k (b_k^{(\theta)} + n_k^{(\theta)} - 1)\log\theta_k + \sum_j\sum_k\sum_{k'}(b_{jkk'}^{(\gamma)} + n_{jkk'}^{(\gamma)} - 1)\log\gamma_{jkk'} \\
&\quad + \sum_k\sum_f (b_{kf}^{(\phi)} + n_{kf}^{(\phi)} - 1)\log\phi_{kf}] + const \\
&= \sum_k E_{\boldsymbol{g}_{-i},\pi_k}[(b_k^{(\theta)} + n_k^{(\theta)} - 1)\log\theta_k] + \sum_j\sum_k\sum_{k'} E_{\nu_{jk},\boldsymbol{g}_{-i}}[(b_{jkk'}^{(\gamma)} + n_{jkk'}^{(\gamma)} - 1)\log\gamma_{jkk'}] \\
&\quad + \sum_{k'}\sum_f E_{\boldsymbol{g}_{-i},\lambda_k}[(b_{kf}^{(\phi)} + n_{kf}^{(\phi)} - 1)\log\phi_{kf}] + const \\
&= \sum_k (b_k^{(\theta)} + E_{\boldsymbol{g}_{-i}}[n_k^{(\theta)}] - 1)E_{\pi_k}[\log\theta_k] + \sum_j\sum_k\sum_{k'}(b_{jkk'}^{(\gamma)} + E_{\boldsymbol{g}_{-i}}[n_{jkk'}^{(\gamma)}] - 1)E_{\nu_{jk}}[\log\gamma_{jkk'}] \\
&\quad + \sum_k\sum_f (b_{kf}^{(\phi)} + E_{\boldsymbol{g}_{-i}}[n_{kf}^{(\phi)}] - 1)E_{\lambda_k}[\log\phi_{kf}] + const \\
&= \sum_k (b_k^{(\theta)} + E_{\boldsymbol{g}_{-i}}[\sum_{i'}\mathbb{1}(y_{i'} = k)] - 1)E_{\pi_k}[\log\theta_k] \\
&\quad + \sum_j\sum_k\sum_{k'}(b_{jkk'}^{(\gamma)} + E_{\boldsymbol{g}_{-i}}[\sum_{i'} a_{i'jk'}\mathbb{1}(y_{i'} = k)] - 1)E_{\nu_{jk}}[\log\gamma_{jkk'}] \\
&\quad + \sum_k\sum_f E_{\boldsymbol{g}_{-i}}[\sum_{i'} x_{i'f}\mathbb{1}(y_{i'} = k)]E_{\lambda_k}[\log\phi_{kf}] + const \\
&= \sum_k E_{\boldsymbol{g}_{-i}}[\sum_{i'}\mathbb{1}(y_{i'} = k)]E_{\pi_k}[\log\theta_k] \\
&\quad + \sum_j\sum_k\sum_{k'} E_{\boldsymbol{g}_{-i}}[\sum_{i'} a_{i'jk'}\mathbb{1}(y_{i'} = k)]E_{\nu_{jk}}[\log\gamma_{jkk'}] \\
&\quad + \sum_k\sum_f x_{if}\mathbb{1}(y_i = k)E_{\lambda_k}[\log\phi_{kf}] + const \\
&= \sum_k \mathbb{1}(y_i = k)E_{\pi_k}[\log\theta_k] + \sum_j\sum_k\sum_{k'} a_{ijk'}\mathbb{1}(y_i = k)E_{\nu_{jk}}[\log\gamma_{jkk'}] \\
&\quad + \sum_k\sum_f x_{if}\mathbb{1}(y_i = k)E_{\lambda_k}[\log\phi_{kf}] + const \\
&= \sum_k \mathbb{1}(y_i = k)\Big[E_{\pi_k}[\log\theta_k] + \sum_j\sum_{k'} a_{ijk'}E_{\nu_{jk}}[\log\gamma_{jkk'}] + \sum_f x_{if}E_{\lambda_k}[\log\phi_{kf}]\Big] + const
\end{aligned}
\tag{9.19}
\tag{9.20}
\tag{9.21}
$$

$$
\begin{aligned}
g_i(y_i) &\propto \prod_k \exp\left[E_{\pi_k}[\log\theta_k] + \sum_j \sum_{k'} a_{ijk'} E_{\nu_{jk}}[\log\gamma_{jkk'}] + \sum_f x_{if} E_{\lambda_k}[\log\phi_{kf}]\right]^{\mathbb{1}(y_i=k)} \\
&\propto \prod_k \alpha_{ik}^{(y)\mathbb{1}(y_i=k)} \\
&= Categorical(\boldsymbol{\alpha}_i^{(y)})
\end{aligned}
$$

where $\alpha_{ik}^{(y)} = E_{\pi_k}[\log\theta_k] + \sum_j \sum_{k'} a_{ijk'} E_{\nu_{jk}}[\log\gamma_{jkk'}] + \sum_f x_{if} E_{\lambda_k}[\log\phi_{kf}]$. As noted in the main paper, the expected value of a log term with respect to a Dirichlet distribution, such as $E_{\pi_k}[\log\theta_k]$, can be computed analytically as $\psi(\alpha_k^{(\theta)}) - \psi(\sum_{k'} \alpha_{k'}^{(\theta)})$, where ψ is the digamma function (and similarly for $E_{\nu_{jk}}[\log\gamma_{jkk'}]$ and $E_{\lambda_k}[\log\phi_{kf}]$).

Explanation. This derivation differs slightly in a couple of locations from what we have seen before. In Equation 9.19 we use the independence property of expectations (Equation 9.9) to separate expectations with repect to $\boldsymbol{g}$ from expectations with respect to approximate distributions $\boldsymbol{\pi}$, $\boldsymbol{\nu}$, and $\boldsymbol{\lambda}$. We can do this because all distributions in our approximate model are independent of one another. Equation 9.20 distributes Dirichlet expectation terms such as $E_{\nu_{jk}}[\log\gamma_{jkk'}]$ and then absorbs terms that do not involve y_i into the constant. Equation 9.21 combines sums over k and then factors out the common multiplier $\mathbb{1}(y_i = k)$.

Lower Bound

The mean-field variational updates we derived above are designed to minimize the objective $KL(q||p^*)$. Computing values of this objective is highly useful during optimization. Tracking the objective's rate of change after each iteration of the optimization algorithm allows us to assess convergence. Second, the objective function provides a powerful debugging tool: if the objective does not improve after an update (modulo floating point noise after convergence) then there is a bug in that particular update. Unfortunately, directly computing $KL(q||p^*)$ is intractable because it involves evaluating the intractable denominator of $p^*(\theta,\gamma,\phi,y|x,a) = \frac{p(\theta,\gamma,\phi,y,x,a)}{p(a,x)} = \frac{p(\theta,\gamma,\phi,y,x,a)}{\int p(\theta,\gamma,\phi,y,x,a)d\theta,\gamma,\phi,y}$. However, because $KL(q||p^*) = KL(q||p) + p(x,a)$ and $p(x,a)$ is constant, minimizing $KL(q||p^*)$ is equivalent to minimizing $KL(q||p)$, which is tractable to evaluate. Finally,

it is typical to treat mean-field optimization as a maximization rather than a minimization problem. We therefore use $-KL(q||p)$ as the objective function for the purposes of tracking convergence and debugging. $-KL(q||p)$ is commonly referred to as a lower bound because it is a lower bound on the log marginal likelihood of the model $p(x, a)$, giving it additional theoretical interest. However, its immediate practical value is as the objective function being optimized by mean-field variational updates.

We first break $-KL(q||p)$ into two manageable parts: entropy $H(q)$, and cross entropy $H(q|p)$:

$$\begin{aligned}
-KL(q||p) &= \int q(\boldsymbol{\theta}, \boldsymbol{\gamma}, \boldsymbol{\phi}, \boldsymbol{y}) \log \frac{p(\boldsymbol{\theta}, \boldsymbol{\gamma}, \boldsymbol{\phi}, \boldsymbol{y}, \boldsymbol{x}, \boldsymbol{a})}{q(\boldsymbol{\theta}, \boldsymbol{\gamma}, \boldsymbol{\phi}, \boldsymbol{y})} d\boldsymbol{\theta}, \boldsymbol{\gamma}, \boldsymbol{\phi}, \boldsymbol{y} \\
&= E_q[\log p(\boldsymbol{\theta}, \boldsymbol{\gamma}, \boldsymbol{\phi}, \boldsymbol{y}, \boldsymbol{x}, \boldsymbol{a})] - E_q[\log q(\boldsymbol{\theta}, \boldsymbol{\gamma}, \boldsymbol{\phi}, \boldsymbol{y})] \\
&= -H(q|p) + H(q)
\end{aligned}$$

Next we simplify each term in the bound separately. An important difference between these derivations and those we did for the mean-field updates is that we must evaluate this function exactly and therefore cannot drop any constants. Therefore when we plug in the log joint distribution we use the full, unsimplified form of each distribution rather than the simplified form in Equation 9.4. The reasoning behind each step here is similar enough to those in the update derivations above that no additional explanation is provided.

The first term of the lower bound $-H(q|p)$ is

$$
\begin{aligned}
&E_q[\log p(\boldsymbol{\theta}, \boldsymbol{\gamma}, \boldsymbol{\phi}, \boldsymbol{y}, \boldsymbol{x}, \boldsymbol{a})] \\
&\quad = E_q\Big[\log p(\boldsymbol{\theta}) + \sum_j \sum_k \log p(\boldsymbol{\gamma}_{jk}) + \sum_k \log p(\boldsymbol{\phi}_k) + \sum_i \log p(y_i|\boldsymbol{\theta}) \\
&\qquad + \sum_i \sum_j \log p(\boldsymbol{a}_{ij}|y_i, \boldsymbol{\gamma}_{jy_i}) + \sum_i \log p(\boldsymbol{x}_i|\boldsymbol{\phi}_{y_i}) \Big] \\
&\quad = E_q\Big[- \log B(\boldsymbol{b}^{(\theta)}) + \sum_k (b_k^{(\theta)} - 1) \log \theta_k \\
&\qquad + \sum_j \sum_k - \log B(\boldsymbol{b}_{jk}^{(\gamma)}) + \sum_{k'} (b_{jkk'}^{(\gamma)} \gamma_{jk} - 1) \log \gamma_{jkk'} \\
&\qquad + \sum_k - \log B(\boldsymbol{b}_k^{(\phi)}) + \sum_f (b_{kf}^{(\phi)} \phi_k - 1) \log \phi_{kf} \\
&\qquad + \sum_i \log \theta_{y_i} \\
&\qquad + \sum_i \sum_j \big(\log |\boldsymbol{a}_{ij}|_1! - \sum_k \log a_{ijk}! \big) + \sum_k a_{ijk} \log \gamma_{jy_i} \\
&\qquad + \sum_i \big(\log |\boldsymbol{x}_i|_1! - \sum_f \log x_{if}! \big) + \sum_f x_{if} \log \phi_{y_{if}} \Big] \\
&\quad = E_q\Big[- \log B(\boldsymbol{b}^{(\theta)}) + \sum_k (b_k^{(\theta)} - 1) \log \theta_k \\
&\qquad + \sum_j \sum_k - \log B(\boldsymbol{b}_{jk}^{(\gamma)}) + \sum_{k'} (b_{jkk'}^{(\gamma)} \gamma_{jk} - 1) \log \gamma_{jkk'} \\
&\qquad + \sum_k - \log B(\boldsymbol{b}_k^{(\phi)}) + \sum_f (b_{kf}^{(\phi)} \phi_k - 1) \log \phi_{kf} \\
&\qquad + \sum_k n_k^{(\theta)} \\
&\qquad + \sum_i \sum_j \big(\log |\boldsymbol{a}_{ij}|_1! - \sum_k \log a_{ijk}! \big) + \sum_i \sum_j \sum_{k'} n_{jkk'}^{(\gamma)} \log \gamma_{jkk'} \\
&\qquad + \sum_i \big(\log |\boldsymbol{x}_i|_1! - \sum_f \log x_{if}! \big) + \sum_k \sum_f n_{kf}^{(\phi)} \log \phi_{kf} \Big]
\end{aligned}
$$

$$
\begin{aligned}
&= E_q\Big[-\log B(\boldsymbol{b}^{(\theta)}) + \sum_k (b_k^{(\theta)} + n_k^{(\theta)} - 1)\log\theta_k \\
&\quad + \sum_j \sum_k -\log B(\boldsymbol{b}_{jk}^{(\gamma)}) + \sum_{k'} (b_{jkk'}^{(\gamma)} + n_{jkk'}^{(\gamma)} - 1)\log\gamma_{jkk'} \\
&\quad + \sum_k -\log B(\boldsymbol{b}_k^{(\phi)}) + \sum_f (b_{kf}^{(\phi)} + n_{kf}^{(\phi)} - 1)\log\phi_{kf} \\
&\quad + \sum_i \sum_j \big(\log|\boldsymbol{a}_{ij}|_1! - \sum_k \log a_{ijk}!\big) + \sum_i \big(\log|\boldsymbol{x}_i|_1! - \sum_f \log x_{if}!\big)\Big] \\
&= -\log B(\boldsymbol{b}^{(\theta)}) + \sum_k E_{\pi,\boldsymbol{g}}[(b_k^{(\theta)} + n_k^{(\theta)} - 1)\log\theta_k] \\
&\quad + \sum_j \sum_k -\log B(\boldsymbol{b}_{jk}^{(\gamma)}) + \sum_{k'} E_{\nu_{jk},\boldsymbol{g}}[(b_{jkk'}^{(\gamma)} + n_{jkk'}^{(\gamma)} - 1)\log\gamma_{jkk'}] \\
&\quad + \sum_k -\log B(\boldsymbol{b}_k^{(\phi)}) + \sum_f E_{\lambda_k,\boldsymbol{h}}[(b_{kf}^{(\phi)} + n_{kf}^{(\phi)} - 1)\log\phi_{kf}] \\
&\quad + \sum_i \sum_j \big(\log|\boldsymbol{a}_{ij}|_1! - \sum_k \log a_{ijk}!\big) + \sum_i \big(\log|\boldsymbol{x}_i|_1! - \sum_f \log x_{if}!\big) \\
&= -\log B(\boldsymbol{b}^{(\theta)}) + \sum_k (b_k^{(\theta)} + \sum_i g_i(k) - 1)E_\pi[\log\theta_k] \\
&\quad + \sum_j \sum_k -\log B(\boldsymbol{b}_{jk}^{(\gamma)}) + \sum_{k'} (b_{jkk'}^{(\gamma)} + \sum_i a_{ijk'} g_i(k) - 1)E_{\nu_{jk}}[\log\gamma_{jkk'}] \\
&\quad + \sum_k -\log B(\boldsymbol{b}_k^{(\phi)}) + \sum_f (b_{kf}^{(\phi)} + \sum_i x_{if} h_i(k) - 1)E_{\lambda_k}[\log\phi_{kf}] \\
&\quad + \sum_i \sum_j \big(\log|\boldsymbol{a}_{ij}|_1! - \sum_k \log a_{ijk}!\big) + \sum_i \big(\log|\boldsymbol{x}_i|_1! - \sum_f \log x_{if}!\big)
\end{aligned}
$$

where $B(\cdot)$ is the multivariate Beta function that normalizes a Dirichlet distribution.

The second term of the lower bound $H(q)$ is

$$
\begin{aligned}
& E_q[\log q(\theta, \boldsymbol{\gamma}, \boldsymbol{\phi}, \boldsymbol{y})] \\
& = E_q\Big[\log \pi(\boldsymbol{\theta}) + \sum_j \sum_k \nu_{jk}(\boldsymbol{\gamma}_{jk}) + \sum_k \lambda_k(\boldsymbol{\phi}_k) + \sum_i g_i(y_i) \Big] \\
& = E_q\Big[- B(\boldsymbol{\alpha}^{(\theta)}) + \sum_k (\alpha_k^{(\theta)} - 1) \log \theta_k \\
& \quad + \sum_j \sum_k - \log B(\boldsymbol{\alpha}_{jk}^{(\gamma)}) + \sum_{k'} (\alpha_{jk}^{(\gamma)} - 1) \log \gamma_{jkk'} \\
& \quad + \sum_k - \log B(\boldsymbol{\alpha}_k^{(\phi)}) + \sum_f (\alpha_{kf}^{(\phi)} - 1) \log \phi_{kf} \\
& \quad + \sum_i \log \alpha_{y_i}^{(y)} \Big] \\
& = -B(\boldsymbol{\alpha}^{(\theta)}) + \sum_k (\alpha_k^{(\theta)} - 1) E_\pi[\log \theta_k] \\
& \quad + \sum_j \sum_k - \log B(\boldsymbol{\alpha}_{jk}^{(\gamma)}) + \sum_{k'} (\alpha_{jk}^{(\gamma)} - 1) E_{\nu_{jk}}[\log \gamma_{jkk'}] \\
& \quad + \sum_k - \log B(\boldsymbol{\alpha}_k^{(\phi)}) + \sum_f (\alpha_{kf}^{(\phi)} - 1) E_{\lambda_k}[\log \phi_{kf}] \\
& \quad + \sum_i E_{g_i}[\log \alpha_{y_i}^{(y)}] \\
& = -B(\boldsymbol{\alpha}^{(\theta)}) + \sum_k (\alpha_k^{(\theta)} - 1) E_\pi[\log \theta_k] \\
& \quad + \sum_j \sum_k - \log B(\boldsymbol{\alpha}_{jk}^{(\gamma)}) + \sum_{k'} (\alpha_{jk}^{(\gamma)} - 1) E_{\nu_{jk}}[\log \gamma_{jkk'}] \\
& \quad + \sum_k - \log B(\boldsymbol{\alpha}_k^{(\phi)}) + \sum_f (\alpha_{kf}^{(\phi)} - 1) E_{\lambda_k}[\log \phi_{kf}] \\
& \quad + \sum_i \sum_k \alpha_k^{(y)} \log \alpha_k^{(y)}
\end{aligned}
$$

Chapter 10

Appendix: Supplementary Material for

Learning from Measurements in Crowdsourcing Models

10.1 Introduction

We describe a crowdsourcing instantiation of the learning from measurements framework proposed by Liang et al. [74]. In this section we introduce the model via a plate diagram in Figure 10.1 and via the generative story given below.

- Draw a distribution θ over classes from a symmetric $Dirichlet(\delta)$
- For each of J annotators, draw a noise level w_j from $InverseGamma(\alpha, \beta)$.
- For each of N documents, draw an unobserved label y_i from $Categorical(\theta)$.
- For each of J annotators, K different measurements are drawn from $Normal(\sum_i \sigma_{jk}(x_i, y_i), w_j)$.

10.2 Variational Inference

First we write the joint probability of the model, simplified as much as possible.

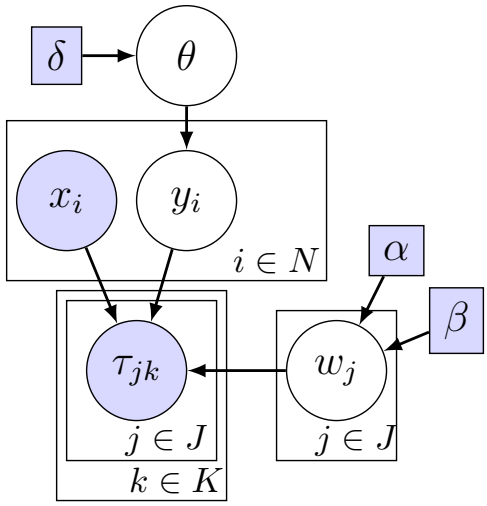


Figure 10.1: Plate Diagram depiction of the measurement model discussed in this paper. Round nodes are variables with distributions. Rectangular nodes are hyperparameters (without distributions). Shaded nodes have known values (although some τ values may be unobserved).

10.2.1 Joint Probability

$$p(\theta, y, w, \tau|x) = p(\theta)\prod_i p(y_i|\theta)\prod_j p(w_j)\prod_j\prod_k p(\tau_{jk}|y, w_j) \tag{10.1}$$

$$= \frac{1}{B(\delta)}\prod_c \theta_c^{\delta-1}\prod_i \theta_{y_i}\prod_j \frac{\beta^\alpha}{\Gamma(\alpha)} w_j^{-\alpha-1}\exp\left(\frac{-\beta}{w_j}\right) \tag{10.2}$$
$$\prod_j\prod_k (2\pi w_j)^{-\frac{1}{2}} exp\left(-\frac{(\tau_{jk} - \sum_i \sigma_{jk}(x_i, y_i))^2}{2w_j}\right)$$

$$= \frac{1}{B(\delta)}\left(\frac{\beta^\alpha}{\Gamma(\alpha)}\right)^J \prod_c \theta_c^{\delta-1}\prod_c \theta_c^{n_c}\prod_j w_j^{-\alpha-1}\exp\left(\frac{-\beta}{w_j}\right) \tag{10.3}$$
$$\prod_j (2\pi)^{(-\frac{K_j}{2})} w_j^{(-\frac{K_j}{2})} exp\left(-\frac{\sum_k(\tau_{jk} - \sum_i \sigma_{jk}(x_i, y_i))^2}{2w_j}\right)$$

$$= \frac{1}{B(\delta)}\left(\frac{\beta^\alpha}{\Gamma(\alpha)}\right)^J \prod_j (2\pi)^{(-\frac{K_j}{2})}\prod_c \theta_c^{\delta+n_c-1} \tag{10.4}$$
$$\prod_j w_j^{-(\alpha+\frac{K_j}{2})-1} exp\left(-\frac{\beta}{w_j} - \frac{\sum_k(\tau_{jk} - \sum_i \sigma_{jk}(x_i, y_i))^2}{2w_j}\right)$$

$$= \frac{1}{B(\delta)}\left(\frac{\beta^\alpha}{\Gamma(\alpha)}\right)^J \prod_j (2\pi)^{(-\frac{K_j}{2})}\prod_c \theta_c^{\delta+n_c-1} \tag{10.5}$$
$$\prod_j w_j^{-(\alpha+\frac{K_j}{2})-1} exp\left(\frac{-\left(\beta + \frac{1}{2}\sum_k(\tau_{jk} - \sum_i \sigma_{jk}(x_i, y_i))^2\right)}{w_j}\right)$$

$$\log p(\theta, y, w, \tau|x) = -\log B(\delta) + J\alpha\log\beta - J\log\Gamma(\alpha) - \sum_j \frac{K_j}{2}\log(2\pi) \tag{10.6}$$
$$+ \sum_c(\delta + n_c - 1)\log\theta_c + \sum_j(-(\alpha + \frac{K_j}{2}) - 1)\log w_j$$
$$- \left(\frac{\beta + \frac{1}{2}\sum_k(\tau_{jk} - \sum_i \sigma_{jk}(x_i, y_i))^2}{w_j}\right)$$

where $n_c = \sum_i \mathbb{1}(y_i = c)$ and K_j is the number of measurements τ provided by annotator j. In equation 10.1 we read the joint distribution from the graphical model. In equation 10.2 we write out each distribution's parametric form. Recall that $\theta \sim Dirichlet(\delta)$, $w_j \sim InverseGamma(\alpha, \beta)$, and $\tau_{jk} \sim Gaussian(\sum_i \sigma(x_i, y_i), w_j)$. In equation 10.3 we move constants to the front, group θ_{y_i}

terms by class, and transform a product over k into a sum in the exponent. In equation 10.4 we combine like terms, in equation 10.5 we simplify further, and in equation 10.6 we convert the whole thing to log space.

10.2.2 Mean field updates

We assume a fully factorized approximate model

$$q(\theta, y, w) = q(\theta|\nu^{(\delta)}) \prod_i q(y_i|\nu_i^{(y)}) \prod_j q(w_j|\nu_j^{(\alpha)}, \nu_j^{(\beta)}) \tag{10.7}$$

so that $KL(q||p)$ will be minimized when variational parameters ν are set such that the following equalities hold [93]:

$$\log q(\theta|\nu^{(\delta)}) = E_{q(y,w)}[\log p(\theta, y, w, \tau|x)] + const \tag{10.8}$$

$$\log q(y_i|\nu_i^{(y)}) = E_{q(\theta,y_{-i},w)}[\log p(\theta, y, w, \tau|x)] + const \tag{10.9}$$

$$\log q(w_j|\nu_j^{(\alpha)}, \nu_j^{(\beta)}) = E_{q(\theta,y,w_{-j})}[\log p(\theta, y, w, \tau|x)] + const. \tag{10.10}$$

We simplify each equation in turn, drawing heavily on properties of expectations found in equations 10.40-10.43.

$$\log q(\theta|\nu^{(\delta)}) = E_{q(y,w)}[\sum_c (\delta + n_c - 1) \log \theta_c] + const \tag{10.11}$$

$$= \sum_c (\delta + E_{q(y)}[n_c] - 1) \log \theta_c + const \tag{10.12}$$

$$= \sum_c (\delta + E_{q(y)}[\sum_i \mathbb{1}(y_i = c)] - 1) \log \theta_c + const \tag{10.13}$$

$$= \sum_c (\delta + \sum_i E_{q(y_i)}[\mathbb{1}(y_i = c)] - 1) \log \theta_c + const \tag{10.14}$$

$$= \sum_c (\delta + \sum_i \nu_{i,c}^{(y)} - 1) \log \theta_c + const \tag{10.15}$$

In equation 10.11 we substitute 10.6 into 10.8 and use the linearity of expectation to move all terms unrelated to w_j out of the expectation and into the additive constant. In equations 10.12-10.14 we use the definition of n_c and the fact that q factors into fully independent distributions to pull as many terms as possible out of the expectation. In equation 10.14 we simplify using the knowledge that for each i, the indicator function will select only a single term from the expectation; namely, $q(y_i = c)$. In equation 10.15 we recognize the log kernel of a Dirichlet distribution with parameters $\nu_c^{(\delta)} = \delta + \sum_i \nu_{y_i,c}^{(\delta)}$.

$$\log q(w_j|\nu_j^{(\alpha)}, \nu_j^{(\beta)})) = E_{q(\theta,y,w_{-j})}\Big[\sum_{j'}(-(\alpha + \frac{K_{j'}}{2}) - 1)\log w_{j'} \tag{10.16}$$
$$- \left(\frac{\beta + \frac{1}{2}\sum_{k\in S(j)}(\tau_{j'k} - \sum_i \sigma_{j'k}(x_i, y_i))^2}{w_{j'}}\right)\Big] + const$$
$$= E_{q(\theta,y,w_{-j})}\Big[(-(\alpha + \frac{K_j}{2}) - 1)\log w_j \tag{10.17}$$
$$- \left(\frac{\beta + \frac{1}{2}\sum_{k\in S(j)}(\tau_{jk} - \sum_i \sigma_{jk}(x_i, y_i))^2}{w_j}\right)\Big] + const$$
$$= (-(\alpha + \frac{K_j}{2}) - 1)\log w_j \tag{10.18}$$
$$- \left(\frac{\beta + \frac{1}{2}\sum_{k\in S(j)} E_{y_i}\left[\left(\sum_i \sigma_{jk}(x_i, y_i)\right)^2\right]}{w_j}\right) + const$$

In equation 10.16 we substitute 10.6 into 10.10, moving all terms that are not related to w_j out of the expectation and absorbing them into the additive constant. In equation 10.17 we push the expectation inside the sum $\sum_j$ by the linearity of expectation, and terms that do not involve w_j are added to the constant. In equation 10.18 we factor additional linear terms out of the expectation, and we recognize the log kernel of an inverse gamma distribution with parameters $\nu_j^{(\alpha)} = \alpha + \frac{K_j}{2}$ and $\nu_j^{(\beta)} = \beta + \frac{1}{2}\sum_{k\in S(j)} E_{y_i}\left[(\sum_i \sigma_{jk}(x_i, y_i))^2\right]$.

Although the term $E_{y_i}\left[\left(\sum_i \sigma_{jk}(x_i, y_i)\right)^2\right]$ appears intractable, we can simplify it.

$$E_{q(y)}\left[\left(\tau_{jk} - \sum_i \sigma_{jk}(x_i, y_i)\right)^2\right] = \tau_{jk}^2 - 2\tau_{jk}\sum_i E_{q(y_i)}\Big[\sigma_{jk}(x_i, y_i)\Big] + E_{y_i}\left[\left(\sum_i \sigma_{jk}(x_i, y_i)\right)^2\right]$$

where the final term is expanded as follows.

$$E_{y_i}\left[\left(\sum_i \sigma_{jk}(x_i, y_i)\right)^2\right] \tag{10.19}$$

$$= \sum_i \sum_{i'} E_{q(y_i, y_{i'})}\Big[\sigma_{jk}(x_i, y_i)\sigma_{jk}(x_{i'}, y_{i'})\Big] \tag{10.20}$$

$$= \sum_i \sum_{i' \neq i} E_{q(y_i, y_{i'})}\Big[\sigma_{jk}(x_i, y_i)\sigma_{jk}(x_{i'}, y_{i'})\Big] + \sum_i E_{q(y_i)}\Big[\sigma_{jk}(x_i, y_i)^2\Big] \tag{10.21}$$

$$= \sum_i \sum_{i' \neq i} E_{q(y_i)}\Big[\sigma_{jk}(x_i, y_i)\Big] E_{q(y_{i'})}\Big[\sigma_{jk}(x_{i'}, y_{i'})\Big] + \sum_i E_{q(y_i)}\Big[\sigma_{jk}(x_i, y_i)^2\Big] \tag{10.22}$$

$$= \sum_i \Big(E_{q(y_i)}\Big[\sigma_{jk}(x_i, y_i))\Big]\Big)^2 - \sum_i E_{q(y_i)}\Big[\sigma_{jk}(x_i, y_i)\Big]^2 + E_{q(y_i)}\Big[\sigma_{jk}(x_i, y_i)^2\Big]. \tag{10.23}$$

Equation 10.20 rewrites the squared sum as a doubly nested loop of term pair products. Equation 10.21 separates out the terms where $i = i'$ into their own sum. Equation 10.22 applies the independent product rule of expectations (Equation 10.42). Finally, equation 10.23 completes the square with missing diagonals $\sum_i E\Big[\sigma_{jk}(x_i, y_i)\Big]^2$ and subtracts the same amount in order to maintain equality. We now have tractable expectations over individual y_i variables (computable via table 10.1), rather than expectations over joint settings of all y variables.

$$\log q(y_i|\nu_i^{(y)}) = E_{q(\theta,y_{-i},w)}\Big[\sum_c(\delta+n_c-1)\log\theta_c \tag{10.24}$$
$$-\sum_j\left(\frac{\beta+\frac{1}{2}\sum_{k\in S(i,j)}(\tau_{jk}-\sum_{i'}\sigma_{jk}(x_{i'},y_{i'}))^2}{w_j}\right)\Big]+const$$
$$=\sum_c(\delta+E_{q(y_{-i})}[n_c]-1)E_{q(\theta_c)}[\log\theta_c]+const \tag{10.25}$$
$$-\sum_j E_{q(w_j)}\left[w_j^{-1}\right]\left(\beta+\frac{1}{2}\sum_{k\in S(i,j)}E_{q(y_{-i})}\left[(\tau_{jk}-\sum_{i'}\sigma_{jk}(x_{i'},y_{i'}))^2\right]\right)$$
$$=\sum_c\big(\delta+\sum_{i'\neq i}q(y_{i'}=c)+\mathbb{1}(y_i=c)-1\big)E_{q(\theta_c)}[\log\theta_c]+const \tag{10.26}$$
$$-\sum_j\frac{1}{2}E_{q(w_j)}[w_j^{-1}]$$
$$\cdot\sum_{k\in S(i,j)}\tau_{jk}^2-2\tau_{jk}\sum_{i'}E_{q(y_{-i})}\left[\sigma_{jk}(x_{i'},y_{i'})\right]+E_{q(y_{-i})}\left[\left(\sum_{i'}\sigma_{jk}(x_{i'},y_{i'})\right)^2\right]$$
$$=\sum_c\mathbb{1}(y_i=c)E_{q(\theta_c)}[\log\theta_c]+const \tag{10.27}$$
$$-\sum_j\frac{1}{2}E_{q(w_j)}[w_j^{-1}]$$
$$\cdot\sum_{k\in S(i,j)}-2\tau_{jk}\sigma_{jk}(x_i,y_i)+\sum_{i'}\sum_{i''}E_{q(y_{-i})}\left[\sigma_{jk}(x_{i'},y_{i'})\sigma_{jk}(x_{i''},y_{i''})\right]$$
$$=E_{q(\theta_{y_i})}[\log\theta_{y_i}]+const \tag{10.28}$$
$$-\sum_j\frac{1}{2}E_{q(w_j)}[w_j^{-1}]$$
$$\cdot\sum_{k\in S(i,j)}-2\tau_{jk}\sigma_{jk}(x_i,y_i)+\sigma_{jk}(x_i,y_i)^2+2\sum_{i'\neq i}E_{q(y_{i'})}\left[\sigma_{jk}(x_{i'},y_{i'})\sigma_{jk}(x_i,y_i)\right]$$
$$=\psi(\nu_{y_i}^{(\delta)})-\psi(\sum_c\nu_c^{(\delta)})+const \tag{10.29}$$
$$-\sum_j\frac{\nu_j^{(\alpha)}}{2\nu_j^{(\beta)}}$$
$$\cdot\sum_{k\in S(i,j)}-2\tau_{jk}\sigma_{jk}(x_i,y_i)+\sigma_{jk}(x_i,y_i)^2+2\sigma_{jk}(x_i,y_i)\sum_{i'\neq i}E_{q(y_{i'})}\left[\sigma_{jk}(x_{i'},y_{i'})\right]$$

Equation 10.24 is obtained by substituting 10.6 into 10.9 and absorbing terms unrelated to y_i into the constant. In equation 10.25 we use the properties of expectations found in Section 10.4 to factor as many terms outside of the expectations as possible. In equation 10.26 we use the definition of n_c and separate the i term from the rest of the sum because it is not included in the expectation. We also recognize that β can be absorbed into the additive constant. Finally, we rewrite the squared sum as a doubly nested sum of multiplied term pairs. In equation 10.27 we distribute the expectation $E_q(\theta_c)[\cdot]$ and fold terms that do not involve y_i into the constant. We also note that double sum form a "matrix" of terms in which only those in row or column i will be absorbed into the additive constant. Because the "matrix" of terms is symmetrical, we calculate it once (minus the diagonal) and multiply it by 2, adding in the diagonal term $E_{q(y_{-i})}[\sigma_{jk}(x_i, y_i)\sigma_{jk}(x_i, y_i)] = \sigma_{jk}(x_i, y_i)^2$ separately. In equation 10.28 the indicator function $\mathbb{1}(y_i = c)$ selects the one summand in which c is equal to y_i. In equation 10.29 we analytically simplify known expected values using identities from Table 10.2. Expectations over measurement functions $E_{q(y_i)})[\sigma_{jk}(x_i, y_i)]$ can be simplified based on the measure function definition. See table 10.1 for the expected values of some common measure functions. Now that we are in a position to evaluate equation 10.29, we can estimate the variational parameter vector $\nu_i^{(y)}$ of the categorical variable y_i by evaluating 10.29 for each value of y_i and then log normalizing the resulting vector.

10.2.3 Lower Bound

At each iteration, the mean field updates shown above simultaneously decrease $KL(q||p)$ and increase the following evidence lower bound (elbo). Calculating this bound allows us to assess convergence as well as debug the update implementation.

$$\mathcal{L} = E_q[\log p(\theta, y, w, \tau|x)] - E_q[\log q(\theta, y, w)] \tag{10.30}$$

We expand these two parts separately. First,

$$E_q[\log p(\theta, y, w, \tau|x)] \tag{10.31}$$

$$= -\log B(\delta) + J\alpha \log \beta - J \log \Gamma(\alpha) - \sum_j \frac{K_j}{2} \log(2\pi)$$
$$+ E_{q(\theta,y,w)} \Big[\sum_c (\delta + n_c - 1) \log \theta_c + \sum_j (-(\alpha + \frac{K_j}{2}) - 1) \log w_j$$
$$- \left(\frac{\beta + \frac{1}{2} \sum_{k \in S(j)} \left(\tau_{jk} - \sum_i \sigma_{jk}(x_i, y_i) \right)^2}{w_j} \right) \Big]$$

$$= -\log B(\delta) + J\alpha \log \beta - J \log \Gamma(\alpha) - \sum_j \frac{K_j}{2} \log(2\pi) \tag{10.32}$$
$$+ \sum_c (\delta + \sum_i E_{q(y_i)}[q(y_i = c)] - 1) E_{q(\theta)}[\log \theta_c]$$
$$+ \sum_j (-(\alpha + \frac{K_j}{2}) - 1) E_{q(w_j)}[\log w_j]$$
$$- E_{q(w_j)}[w_j^{-1}] \left(\beta + \frac{1}{2} \sum_{k \in S(j)} E_{q(y)} \left[\left(\tau_{jk} - \sum_i \sigma_{jk}(x_i, y_i) \right)^2 \right] \right)$$

$$= -\log B(\delta) + J\alpha \log \beta - J \log \Gamma(\alpha) - \sum_j \frac{K_j}{2} \log(2\pi) \tag{10.33}$$
$$+ \sum_c \left(\delta + \sum_i \nu_{i,c}^{(y)} - 1 \right) \left(\psi(\nu_c^{(\delta)}) - \psi(\sum_{c'} \nu_{c'}^{(\delta)}) \right)$$
$$+ \sum_j (-(\alpha + \frac{K_j}{2}) - 1)(\log \nu_j^{(\beta)} - \psi(\nu_j^{(\alpha)}))$$
$$- \frac{\nu_j^{(\alpha)}}{\nu_j^{(\beta)}} \Bigg(\beta + \frac{1}{2} \sum_{k \in S(j)} \tau_{jk}^2 - 2\tau_{jk} \sum_i E_{q(y_i)} \Big[\sigma_{jk}(x_i, y_i) \Big]$$
$$+ \Big(\sum_i E_{y_i} \left[\sigma_{jk}(x_i, y_i) \right] \Big)^2 - \sum_i E_{y_i} \left[\sigma_{jk}(x_i, y_i) \right]^2 + \sum_i E_{y_i} \left[\sigma_{jk}(x_i, y_i)^2 \right] \Bigg)$$

using the same step justifications as the parameter update derivations, but keeping track of all constants.

Similarly,

$$E_q\big[\log q(\theta, y, w)\big] = E_{q(\theta,y,w)}[\log q(\theta) + \sum_j \log q(w_j) + \sum_i \log q(y_i)\big] \tag{10.34}$$

$$= E_{q(\theta)}[\log q(\theta)] + \sum_j E_{q(w_j)}[\log q(w_j)] + \sum_i E_{q(y_i)}[\log q(y_i)] \tag{10.35}$$

$$= E_{q(\theta)}\Big[- \log B(\nu^{(\delta)}) + \sum_c (\nu_c^{(\delta)} - 1) \log \theta_c\Big] \tag{10.36}$$

$$+ \sum_j E_{q(w_j)}\Big[\nu^{(\alpha)} \log \nu^{(\beta)} - \log \Gamma(\nu^{(\alpha)}) + (-\nu^{(\alpha)} - 1) \log w_j - (\frac{\nu^{(\beta)}}{w_j})\Big]$$

$$+ \sum_i E_{q(y_i)}[\log q(y_i)]$$

$$= - \log B(\nu^{(\delta)}) + \sum_c (\nu_c^{(\delta)} - 1) E_{q(\theta)}[\log \theta_c] \tag{10.37}$$

$$+ \sum_j \nu^{(\alpha)} \log \nu^{(\beta)} - \log \Gamma(\nu^{(\alpha)}) + (-\nu^{(\alpha)} - 1) E_{q(w_j)}[\log w_j] - E_{q(w_j)}[w_j^{-1}] \nu^{(\beta)}$$

$$+ \sum_i \sum_c \nu_{i,c}^{(y)} \log \nu_{i,c}^{(y)}$$

$$= - \log B(\nu^{(\delta)}) + \sum_c (\nu_c^{(\delta)} - 1) \big(\psi(\nu_c^{(\delta)}) - \psi(\sum_{c'} \nu_{c'}^{(\delta)}\big) \tag{10.38}$$

$$+ \sum_j \nu^{(\alpha)} \log \nu^{(\beta)} - \log \Gamma(\nu^{(\alpha)}) + (-\nu^{(\alpha)} - 1)(\log(\nu_j^{(\beta)}) - \psi(\nu_j^{(\alpha)})) - \frac{\nu^{(\alpha)}}{\nu^{(\beta)}} \nu^{(\beta)}$$

$$+ \sum_i \sum_c \nu_{i,c}^{(y)} \log \nu_{i,c}^{(y)}$$

$$= - \log B(\nu^{(\delta)}) + \sum_c (\nu_c^{(\delta)} - 1) \big(\psi(\nu_c^{(\delta)}) - \psi(\sum_{c'} \nu_{c'}^{(\delta)}\big) \tag{10.39}$$

$$- \Big(\sum_j \log \Gamma(\nu_j^{(\alpha)}) - \psi(\nu_j^{(\alpha)})(\nu^{(\alpha)} + 1) + \log \nu_j^{(\beta)} + \nu_j^{(\alpha)}\Big)$$

$$+ \sum_i \sum_c \nu_{i,c}^{(y)} \log \nu_{i,c}^{(y)}$$

10.3 Calculating Expected Values

Measurement Type	$\sigma_{jk}(x, y)$	$\sum_i E_{q(y_i)}[\sigma_{jk}(x, y)]$
Label	$\mathbb{1}(x = x_m, y = c)$	$q(y_m = c)$
Labeled Predicate	$\mathbb{1}(f(x) = 1, y = c)$	$\sum_{i \in f(X)} q(y_i = c)$
Label Preference	$\mathbb{1}(y = c)$	$\sum_i q(y_i = c)$

Table 10.1: Some common measurement functions and their expected values. Where not listed, squared measurement functions are identical to measurement functions (true for all functions with binary outputs).

Variable Definition	Term	Value
$v\|\nu \sim$ Dirichlet	$E_{q(v)}[\log v_c]$	$\psi(\nu_c) - \psi(\sum_c \nu_c)$
$v\|\alpha, \beta \sim$ Inverse Gamma	$E_{q(v)}[v]$	$\frac{\beta}{\alpha - 1}$ for $\alpha > 1$
$v\|\alpha, \beta \sim$ Inverse Gamma	$E_{q(v)}[\log v]$	$\log(\beta) - \psi(\alpha)$

Table 10.2: Expected Value identities

10.4 Properties of Expectations

Marginal Expectation

$$
\begin{aligned}
E_{p(x,y)}[f(x)] &= \int_x \int_y p(x, y) f(x) \\
&= \int_x f(x) \int_y p(x, y) \\
&= \int_x f(x) p(x) \\
&= E_{p(x)}[f(x)]
\end{aligned}
\tag{10.40}
$$

This property also holds for discrete variables by substituting sums for integrals.

Expectation of a sum of independent random variables

Given a vector of independent random variables $\boldsymbol{x}$ such that $p(\boldsymbol{x}) = \prod_i p_i(x_i)$

$$\begin{aligned} E_{p(\boldsymbol{x})}[\sum_i^N f(x_i)] &= \sum_i^N E_{p(\boldsymbol{x})}[f(x_i)] \\ &= \sum_i^N E_{p_i(x_i)}[f(x_i)] \end{aligned} \tag{10.41}$$

The first line follows from the linearity of expectation, and the second from property 10.40.

Expectation of a product of independent random variables

Given a independent random variables x, y such that $p(x, y) = p(x)p(y)$

$$\begin{aligned} E_{p(x,y)}[f(x)g(y)] &= \int_x \int_y p(x,y) f(x) g(y) \\ &= \int_x \int_y p(x) p(y) f(x) g(y) \\ &= \int_x p(x) f(x) \int_y p(y) g(y) \\ &= E_{p(x)}[f(x)] E_{p(y)}[g(y)] \end{aligned} \tag{10.42}$$

Equality follows from the independence of variables and re-ordering multiplicative terms.

Expected value of a discrete indicator function

$$\begin{aligned} E_{p(x)}[\mathbb{1}[x = c]] &= \sum_x p(x) \mathbb{1}[x = c] \\ &= p(c) \end{aligned} \tag{10.43}$$

The second line follows from considering that there only exists one setting of x for which $\mathbb{1}[x = c]$ is non-zero, namely the setting where $x = c$. This is the discrete analog to the well-known integral of a time-shifted continuous dirac delta function $\int_x f(x)\delta(x - c) = f(c)$.

Chapter 11

Appendix: Deriving the majority vote procedure from an item response model under limiting assumptions

The majority vote procedure can be derived by starting with a multinomial item response model in Figure 4.1a and making some (strong!) limiting assumptions.

Note on notation: Dimensionality is indicated by indices (you'll see no bolds or bars except in the figure).

Limiting Assumptions

1. θ is fixed and known such that each class is equally likely. $\forall k\ \theta_k = \frac{1}{K}$
2. The γ_j matrices are fixed and known and populated with one of two values: V for the diagonals and a smaller v for the off-diagonals.
   $\forall j \forall k \forall k'\ \gamma_{jkk'} = V\mathbb{1}(k = k') + v(1 - \mathbb{1}(k = k'))$ where $V > v$

Majority Vote Derivation

First note that since θ and γ are fixed and known, y variables are independent from one another and thus $p(y|a, \theta, \gamma) = \prod_i p(y_i|a_i, \theta, \gamma)$. Therefore **from now on we will confine ourselves to reasoning about a single instance** i with its accompanying annotations. For visual simplicity, I'm not even going to include the i index into y or a anymore, although implicitly it is there. Don't let the omission of i confuse you.

$$p(y = k|a, \theta, \gamma) = \frac{p(y = k, a|\theta, \gamma)}{p(a|\theta, \gamma)} \tag{11.1}$$

$$= \frac{p(y = k|\theta) \prod_j p(a_j|y = k, \gamma)}{p(a|\theta, \gamma)} \tag{11.2}$$

$$= \frac{1}{K} \cdot \frac{1}{p(a|\theta, \gamma)} \cdot \prod_j \prod_{k'} \gamma_{jkk'}^{a_{jk'}} \tag{11.3}$$

$$= \frac{1}{Z} \cdot \prod_j \prod_{k'} \gamma_{jkk'}^{a_{jk'}} \tag{11.4}$$

where $Z = K \cdot p(a|\theta, \gamma)$ (note that $p(a|\theta, \gamma)$ is a constant.)

Counting annotations will be important, so we define the following shorthand: $n^{(k)} = \sum_j \sum_{k'} \mathbb{1}(a_{jk'} = k)$. That is, $n^{(k)}$ is the number of times that any annotator gave the annotation k for the instance i implicitly under consideration. Similarly, $n^{(\neg k)} = \sum_j \sum_{k'} \mathbb{1}(a_{jk'} \neq k)$ is the number of annotations not equal to k. Equation 11.4 can be simplified by applying assumption 2.

$$p(y = k|a, \theta, \gamma) = \frac{1}{Z} \cdot V^{n^{(k)}} \cdot v^{n^{(\neg k)}} \tag{11.5}$$

A useful property of the counts $n^{(k)}$ and $n^{(\neg k)}$ is that $n^{(k)} + n^{(\neg k)} = A$ where A is the total number of annotations for the instance. Therefore, for any pair of labels c and d, $n^{(c)} + n^{(\neg c)} = n^{(d)} + n^{(\neg d)} = A$, and using algebra

$$-(n^{(c)} - n^{(d)}) = n^{(\neg c)} - n^{(\neg d)} \tag{11.6}$$

Equations 11.5 and 11.6 can be used to define a simple decision rule δ between class c and d such that c is chosen iff $log\delta(c, d) > 0$.

$$log\delta(c, d) \tag{11.7}$$
$$= log\frac{p(y = c|a, \theta, \gamma)}{p(y = d|a, \theta, \gamma)} \tag{11.8}$$
$$= n^{(c)}logV + n^{(\neg c)}logv - logZ - n^{(d)}logV - n^{(\neg d)}logv + logZ \tag{11.9}$$
$$= (n^{(c)} - n^{(d)})logV + (n^{(\neg c)} - n^{(\neg d)})logv \tag{11.10}$$
$$= (n^{(c)} - n^{(d)})logV - (n^{(c)} - n^{(d)})logv \tag{11.11}$$
$$= (n^{(c)} - n^{(d)})(logV - logv) \tag{11.12}$$

Since $logV - logv$ is constant and positive (recall that $V > v$ by assumption #2), this decision rule results in simple majority vote.

References

[1] Arthur Asuncion, Max Welling, Padhraic Smyth, and Yee Whye Teh. On smoothing and inference for topic models. In *Proc. Conference on Uncertainty in Artificial Intelligence (UAI)*, 2009.

[2] AiTi Aw, Min Zhang, Juan Xiao, and Jian Su. A phrase-based statistical model for SMS text normalization. In *Proc. International Conference on Computational Linguistics (COLING)*, 2006.

[3] Elena Beisswanger, Vivian Lee, Jung-Jae Kim, Dietrich Rebholz-Schuhmann, Andrea Splendiani, Olivier Dameron, Stefan Schulz, and Udo Hahn. Gene regulation ontology (GRO): design principles and use cases. In *Proc. Medical Informatics Europe Conference (MIE)*, 2008.

[4] Michael W. Berry, Murray Browne, and Ben Signer. Topic annotated Enron email data set. *Linguistic Data Consortium*, 2001.

[5] Julian Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society*, 48(3):259–302, 1986.

[6] Steven Bird and Gary Simons. Seven dimensions of portability for language documentation and description. In *Proc. International Conference on Language Resources and Evaluation (LREC)*, 2002.

[7] David M. Blei. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84, 2012.

[8] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022, 2003.

[9] Jonathan Bragg, Mausam, and Daniel S. Weld. Crowdsourcing multi-label classification for taxonomy creation. In *Proc. Conference on Human Computation and Crowdsourcing (HCOMP)*, 2013.

[10] Eric Brill. A simple rule-based part of speech tagger. In *Proc. ACL Workshop on Speech and Natural Language*, 1992.

[11] Peter F Brown, Vincent J Della Pietra, Robert L Mercer, Stephen A Della Pietra, and Jennifer C Lai. An estimate of an upper bound for the entropy of English. *Computational Linguistics*, 18(1):31–40, 1992.

[12] Rainer E Burkard, Mauro Dell'Amico, and Silvano Martello. *Assignment Problems, Revised Reprint*. Siam, 2009.

[13] Jing Cao, S Lynne Stokes, and Song Zhang. A Bayesian approach to ranking and rater evaluation an application to grant reviews. *Journal of Educational and Behavioral Statistics*, 35(2):194–214, 2010.

[14] Ana Margarida de Jesus Cardoso-Cachopo. *Improving Methods for Single-label Text Categorization*. PhD thesis, Universidade Tecnica de Lisboa, 2007.

[15] Bob Carpenter. Multilevel Bayesian models of categorical data annotation. *Unpublished manuscript*, 2008.

[16] Xavier Carreras and Lluís Màrquez. Introduction to the CoNLL-2005 shared task: semantic role labeling. In *Proc. Conference on Computational Natural Language Learning (CoNLL)*, 2005.

[17] James Carroll. *A Bayesian Decision Theoretical Approach to Supervised Learning, Selective Sampling, and Empirical Function Optimization*. PhD thesis, Brigham Young University, 2010.

[18] James Carroll, Robbie A. Haertel, Peter McClanahan, Eric K. Ringger, and Kevin Seppi. Modeling the annotation process for ancient corpus creation. In *Proc. International Conference of Electronic Corpora of Ancient Languages (ECAL)*, 2007.

[19] Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

[20] Ming-Wei Chang, Lev-Arie Ratinov, Nicholas Rizzolo, and Dan Roth. Learning and inference with constraints. In *Proc. Conference on Artificial Intelligence (AAAI)*, 2008.

[21] Fu-Dong Chiou, David Chiang, and Martha Palmer. Facilitating treebank annotation using a statistical parser. In *Proc. International Conference on Human Language Technology Research (HLT)*, 2001.

[22] Kenneth W. Church. A stochastic parts program and noun phrase parser for unrestricted text. In *Proc. Conference on Applied Natural Language Processing (ANLC)*, 1988.

[23] Christopher Cieri, Marian Reed, Denise DiPersio, and Mark Liberman. Twenty years of language resource development and distribution: A progress report on LDC activities. In *Proc. International Conference on Language Resources and Evaluation (LREC)*, 2012.

[24] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537, 2011.

[25] David Crystal. *Language Death*. Cambridge University Press, 2002.

[26] Aron Culotta and Andrew McCallum. Reducing labeling effort for structured prediction tasks. In *Proc. Conference on Artificial Intelligence (AAAI)*, 2005.

[27] Nilesh Dalvi, Anirban Dasgupta, Ravi Kumar, and Vibhor Rastogi. Aggregating crowdsourced binary ratings. In *Proc. International Conference on World Wide Web (WWW)*, 2013.

[28] Cristian Danescu-Niculescu-Mizil, Justin Cheng, Jon Kleinberg, and Lillian Lee. You had me at hello: How phrasing affects memorability. In *Proc. Annual Meeting of the Association for Computational Linguistics (ACL)*, 2012.

[29] Hal Daumé III. Frustratingly easy domain adaptation. In *Proc. Annual Meeting of the Association for Computational Linguistics (ACL)*, 2007.

[30] Alexander P. Dawid and Allan M. Skene. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Applied Statistics*, pages 20–28, 1979.

[31] Pinar Donmez and Jaime G. Carbonell. Proactive learning: Cost-sensitive active learning with multiple imperfect oracles. In *Proc. ACM Conference on Information and Knowledge Management*, 2008.

[32] Gregory Druck, Gideon Mann, and Andrew McCallum. Learning from labeled features using generalized expectation criteria. In *Proc. International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2008.

[33] Gregory Druck, Burr Settles, and Andrew McCallum. Active learning by labeling features. In *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2009.

[34] ELRA. European language resources association. `http://www.elra.info`, 2008.

[35] Paul Felt, Eric K. Ringger, Kevin Seppi, Kristian Heal, Robbie Haertel, and Deryle Lonsdale. First results in a study evaluating pre-labeling and correction propagation for machine-assisted Syriac morphological analysis. In *Proc. International Conference on Language Resources and Evaluation (LREC)*, 2012.

[36] Paul Felt, Robbie A. Haertel, Eric K. Ringger, and Kevin Seppi. MomResp: A Bayesian model for multi-annotator document labeling. In *Proc. International Conference on Language Resources and Evaluation (LREC)*, 2014.

[37] Paul Felt, Eric K. Ringger, Kevin Seppi, Kristian S. Heal, Robbie A. Haertel, and Deryle Lonsdale. Evaluating machine-assisted annotation in under-resourced settings. *Language Resources and Evaluation*, 48(4):561–599, 2014.

[38] Paul Felt, Eric K. Ringger, Kevin D. Seppi, and Kristian Heal. Using transfer learning to assist exploratory corpus annotation. In *Proc. International Conference on Language Resources and Evaluation (LREC)*, 2014.

[39] Paul Felt, Eric K. Ringger, Jordan Boyd-Graber, and Kevin Seppi. Making the most of crowdsourced document annotations: Confused supervised LDA. In *Proc. Conference on Computational Natural Language Learning (CoNLL)*, 2015.

[40] Paul Felt, Eric K. Ringger, Kevin Seppi, and Robbie A. Haertel. Early gains matter: A case for preferring generative over discriminative crowdsourcing models. In *Proc. Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2015.

[41] Dan Flickinger, Yi Zhang, and Valia Kordoni. Deepbank: A dynamically annotated treebank of the Wall Street Journal. In *Proc. International Workshop on Treebanks and Linguistic Theories (TLT)*, 2012.

[42] W. Nelson Francis and Henry Kučera. Brown Corpus Manual. Technical report, Brown University, 1979.

[43] W. Nelson Francis, Henry Kučera, and Andrew W. Mackie. *Frequency Analysis of English Usage*. Houghton Mifflin Company, 1982.

[44] Alona Fyshe, Leila Wehbe, Partha P. Talukdar, Brian Murphy, and Tom M. Mitchell. A compositional and interpretable semantic space. In *Proc. Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2015.

[45] Kuzman Ganchev, Fernando Pereira, Mark Mandel, Steven Carroll, and Peter White. Semi-automated named entity annotation. In *Proc. ACL Linguistic Annotation Workshop (LAW)*, 2007.

[46] Kuzman Ganchev, João Graça, Jennifer Gillenwater, and Ben Taskar. Posterior regularization for structured latent variable models. *The Journal of Machine Learning Research*, 11: 2001–2049, 2010.

[47] Dee Gardner and Mark Davies. Pointing out frequent phrasal verbs: A corpus-based analysis. *Tesol Quarterly*, 41(2):339–359, 2007.

[48] Arpita Ghosh, Satyen Kale, and Preston McAfee. Who moderates the moderators?: crowdsourcing abuse detection in user-generated content. In *Proc. ACM Conference on Electronic Commerce (EC)*, 2011.

[49] Jost Gippert, Nikolaus P. Himmelmann, and Ulrike Mosel. *Essentials of Language Documentation*, volume 178. Walter de Gruyter, 2006.

[50] Lenore A. Grenoble and Lindsay J. Whaley. *Endangered Languages: Language Loss and Community Response*. Cambridge University Press, 1998.

[51] Robbie A. Haertel. *Practical Cost-Conscious Active Learning for Data Annotation in Annotator-Initiated Environments*. PhD thesis, Brigham Young University, 2013.

[52] Robbie A. Haertel, Paul Felt, Eric K. Ringger, and Kevin Seppi. Parallel active learning: Eliminating wait time with minimal staleness. In *Proc. HLT-NAACL 2010 Workshop on Active Learning for Natural Language Processing*, 2010.

[53] Jan Hajic. Building a syntactically annotated corpus: The Prague dependency treebank. *Issues of Valency and Meaning*, pages 106–132, 1998.

[54] Bo Han, Paul Cook, and Timothy Baldwin. Automatically constructing a normalisation dictionary for microblogs. In *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2012.

[55] Jiawei Han. Mining heterogeneous information networks by exploring the power of links. In *Proc. International Conference on Discovery Science (DS)*, 2009.

[56] Chien-Ju Ho, Aleksandrs Slivkins, Siddharth Suri, and Jennifer W. Vaughan. Incentivizing high quality crowdwork. In *Proc. International Conference on World Wide Web*, 2015.

[57] Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard H. Hovy. Learning whom to trust with MACE. In *Proc. Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2013.

[58] Eduard Hovy and Julia Lavid. Towards a 'science' of corpus annotation: a new methodological challenge for corpus linguistics. *International Journal of Translation*, 22(1):13–36, 2010.

[59] Rong Jin and Zoubin Ghahramani. Learning with multiple labels. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2002.

[60] T. Joachims. A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. In *Proc. International Conference on Machine Learning (ICML)*, 1997.

[61] David Jurgens. Embracing ambiguity: A comparison of annotation methodologies for crowdsourcing word sense labels. In *Proc. Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2013.

[62] David R. Karger, Sewoong Oh, and Devavrat Shah. Efficient crowdsourcing for multi-class labeling. *ACM SIGMETRICS Performance Evaluation Review*, 41(1):81–92, 2013.

[63] Adam Kilgarriff. Gold standard datasets for evaluating word sense disambiguation programs. *Computer Speech & Language*, 12(4):453–472, 1998.

[64] Göran Kjellmer. *A Dictionary of English Collocations: Based on the Brown Corpus*. Oxford University Press, USA, 1994.

[65] Klaus Krippendorff. *Content Analysis: An Introduction to its Methodology*. Sage, 2012.

[66] Anthony S. Kroch. Reflexes of grammar in patterns of language change. *Language Variation and Change*, 1(03):199–244, 1989.

[67] Germán Kruszewski and Marco Baroni. So similar and yet incompatible: Toward the automated identification of semantically compatible words. In *Proc. Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2015.

[68] Henry Kučera and W. Nelson Francis. *Computational Analysis of Present-day American English*. Brown University Press, Providence, RI, 1967.

[69] Chuck P. Lam and David G. Stork. Toward optimal labeling strategy under multiple unreliable labelers. In *Proc. AAAI Spring Symposium: Knowledge Collection from Volunteer Contributors*, 2005.

[70] Quoc V. Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proc. International Conference on Machine Learning (ICML)*, 2014.

[71] Claudia Leacock, George A Miller, and Martin Chodorow. Using corpus statistics and WordNet relations for sense identification. *Computational Linguistics*, 24(1):147–165, 1998.

[72] Geoffrey Leech. Corpus annotation schemes. *Literary and Linguistic Computing*, 8(4): 275–281, 1993.

[73] Abby Levenberg, Stephen Pulman, Karo Moilanen, Edwin Simpson, and Stephen Roberts. Predicting economic indicators from web text using sentiment composition. *International Journal of Computer and Communication Engineering*, 3(2):109–115, 2014.

[74] Pery Liang, Michael I. Jordan, and Dan Klein. Learning from measurements in exponential families. In *Proc. International Conference on Machine Learning (ICML)*, 2009.

[75] C. Lin, Mausam, and D. Weld. Dynamically switching between synergistic workflows for crowdsourcing. In *Proc. Conference on Artificial Intelligence (AAAI)*, 2012.

[76] Christopher H. Lin, Mausam, and Daniel S. Weld. To re (label), or not to re (label). In *Proc. Conference on Human Computation and Crowdsourcing (HCOMP)*, 2014.

[77] Chu-Cheng Lin, Waleed Ammar, Chris Dyer, and Lori Levin. Unsupervised POS induction with word embeddings. In *Proc. Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2015.

[78] Jun S. Liu. The collapsed Gibbs sampler in Bayesian computations with applications to a gene regulation problem. *Journal of the American Statistical Association*, 89(427):958–966, 1994.

[79] Qiang Liu, Jian Peng, and Alex T. Ihler. Variational inference for crowdsourcing. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2012.

[80] Mohamed Maamouri, Ann Bies, Tim Buckwalter, and Wigdan Mekki. The Penn Arabic treebank: Building a large-scale annotated Arabic corpus. In *Proc. Conference on Arabic Language Resources and Tools (NEMLAR)*, 2004.

[81] Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT press, 1999.

[82] Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. *Treebank-2 LDC95T7*. Linguistic Data Consortium, 1995.

[83] Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993.

[84] Jon D. Mcauliffe and David Blei. Supervised topic models. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2007.

[85] Andrew McCallum. MALLET: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.

[86] Jean-Baptiste Michel, Yuan Kui Shen, Aviva Presser Aiden, Adrian Veres, Matthew K Gray, Joseph P Pickett, Dale Hoiberg, Dan Clancy, Peter Norvig, and Jon Orwant. Quantitative analysis of culture using millions of digitized books. *Science*, 331(6014):176–182, 2011.

[87] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *Proc. Workshops at International Conference on Learning Representations (ICLR)*, 2013.

[88] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems (NIPS)*, 2013.

[89] Thomas Minka. Estimating a Dirichlet distribution. Technical report, MIT, 2000.

[90] Jeff Mitchell and Mirella Lapata. Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1429, 2010.

[91] Ruslan Mitkov. *Anaphora Resolution*. Routledge, 2014.

[92] Barzan Mozafari, Purna Sarkar, Michael Franklin, Michael Jordan, and Samuel Madden. Scaling up crowd-sourcing to very large datasets: a case for active learning. In *Proc. Very Large Databases (VLDB) Conference*, 2014.

[93] Kevin Murphy. *Machine Learning: A Probabilistic Perspective*. MIT press, 2012.

[94] Arvind Neelakantan, Jeevan Shankar, Alexandre Passos, and Andrew McCallum. Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

[95] Nadja Nesselhauf. Learner corpora and their potential for language teaching. *How to Use Corpora in Language Teaching*, 12:125–152, 2004.

[96] Andrew Y. Ng and Michael I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and Naive Bayes. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2001.

[97] An T. Nguyen, Byron C. Wallace, and Matthew Lease. Combining crowd and expert labels using decision theoretic active learning. In *Proc. 3rd AAAI Conference on Human Computation (HCOMP)*, 2015.

[98] Thien Huu Nguyen and Ralph Grishman. Employing word representations and regularization for domain adaptation of relation extraction. In *Proc. Annual Meeting of the Association for Computational Linguistics (ACL)*, 2014.

[99] Viet-An Nguyen, Jordan L. Boyd-Graber, and Philip Resnik. Lexical and hierarchical topic regression. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2013.

[100] Kamal Nigam, Andrew McCallum, and Tom Mitchell. Semi-supervised text classification using EM. *Semi-Supervised Learning*, pages 33–56, 2006.

[101] Martha Palmer, Daniel Gildea, and Paul Kingsbury. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106, 2005.

[102] Sinno J. Pan and Qiang Yang. A survey on transfer learning. *Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

[103] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2002.

[104] Rebecca J. Passonneau and Bob Carpenter. The benefits of a model of annotation. *Transactions of the Association for Computational Linguistics*, 2:311–326, 2014.

[105] Jeff Pasternack and Dan Roth. Latent credibility analysis. In *Proc. International Conference on World Wide Web (WWW)*.

[106] Jeff Pasternack and Dan Roth. Knowing what to believe (when you already know something). In *Proc. International Conference on Computational Linguistics (COLING)*, 2010.

[107] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Proc. Empiricial Methods in Natural Language Processing (EMNLP 2014)*, 2014.

[108] John P. Pestian, Pawel Matykiewicz, Michelle Linn-Gust, Brett South, Ozlem Uzuner, Jan Wiebe, K. Bretonnel Cohen, John Hurdle, and Christopher Brew. Sentiment analysis of suicide notes: A shared task. *Biomedical Informatics Insights*, 5(1):3, 2012.

[109] Vikas C. Raykar and Shipeng Yu. Eliminating spammers and ranking annotators for crowdsourced labeling tasks. *The Journal of Machine Learning Research*, 13:491–518, 2012.

[110] Vikas C. Raykar, Shipeng Yu, Linda H. Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. Learning from crowds. *The Journal of Machine Learning Research*, 11:1297–1322, 2010.

[111] Radim Řehůřek and Petr Sojka. Software framework for topic modelling with large corpora. In *Proc. LREC Workshop on New Challenges for NLP Frameworks*, 2010.

[112] Jeffrey C. Reynar and Adwait Ratnaparkhi. A maximum entropy approach to identifying sentence boundaries. In *Proc. Conference on Applied Natural Language Processing (ANLC)*, 1997.

[113] Dan Roth and Wen-tau Yih. *A Linear Programming Formulation for Global Inference in Natural Language Tasks*. Defense Technical Information Center, 2004.

[114] Geoffrey Sampson. *English for the Computer: The SUSANNE Corpus and Analytic Scheme*. Oxford, 1995.

[115] Geoffrey Sampson. The SUSANNE analytic scheme. `http://www.grsampson.net/RSue.html`, 2008. Accessed: 10/3/2012.

[116] Beatrice Santorini. Part-of-speech tagging guidelines for the Penn Treebank Project (3rd revision). Technical report, University of Pennsylvania, 1990.

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

[118] Victor S. Sheng, Foster Provost, and Panagiotis G. Ipeirotis. Get another label? Improving data quality and data mining using multiple, noisy labelers. In *Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2008.

[119] Aashish Sheshadri and Matthew Lease. Square: A benchmark for research on computing crowd consensus. In *Proc. Conference on Human Computation and Crowdsourcing (HCOMP)*, 2013.

[120] Edwin Simpson and Stephen Roberts. Bayesian methods for intelligent task assignment in crowdsourcing systems. In *Decision Making: Uncertainty, Imperfection, Deliberation and Scalability*, pages 1–32. Springer, 2015.

[121] Edwin Simpson, Stephen Roberts, Ioannis Psorakis, and Arfon Smith. Dynamic Bayesian combination of multiple imperfect classifiers. In *Decision Making and Imperfection*, pages 1–35. Springer, 2013.

[122] Padhraic Smyth, Usama Fayyad, Michael Burl, Pietro Perona, and Pierre Baldi. Inferring ground truth from subjective labelling of Venus images. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, 1995.

[123] 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 *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2008.

[124] Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. Semantic compositionality through recursive matrix-vector spaces. In *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2012.

[125] Vivek Srikumar and Christopher D. Manning. Learning distributed representations for structured output prediction. In *Advances in Neural Information Processing Systems 27*, pages 3266–3274. 2014.

[126] Sanja Stajner. Towards a better exploitation of the Brown 'family' corpora in diachronic studies of British and American English language varieties. In *Proc. Recent Advances in NLP (RANLP) Student Research Workshop*, 2011.

[127] Matthew Stephens. Dealing with label switching in mixture models. *Journal of the Royal Statistical Society*, 62(4):795–809, 2000.

[128] James Surowiecki. *The Wisdom of Crowds*. Random House LLC, 2005.

[129] György Szarvas, Veronika Vincze, Richárd Farkas, and János Csirik. The Bioscope corpus: annotation for negation, uncertainty and their scope in biomedical texts. In *Proc. ACL Workshop on Current Trends in Biomedical Natural Language Processing*, 2008.

[130] E. Tjong Kim Sang and F. De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proc. Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2003.

[131] Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proc. Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2003.

[132] Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proc. Annual Meeting of the Association for Computational Linguistics (ACL)*, 2010.

[133] Daniel D. Walker. *Bayesian Text Analytics for Document Collections*. PhD thesis, Brigham Young University, 2012.

[134] Daniel S. Weld and Mausam Peng Dai. Human intelligence needs artificial intelligence. In *Proc. Workshops at the Conference on Artificial Intelligence (AAAI)*, 2011.

[135] Peter Welinder, Steve Branson, Pietro Perona, and Serge J. Belongie. The multidimensional wisdom of crowds. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2010.

[136] Jacob Whitehill, Ting-fan Wu, Jacob Bergsma, Javier R Movellan, and Paul L. Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2009.

[137] David H. Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–259, 1992.

[138] Wei Xu, Alan Ritter, Chris Callison-Burch, William B. Dolan, and Yangfeng Ji. Extracting lexically divergent paraphrases from Twitter. *Transactions of the Association for Computational Linguistics*, 2:435–448, 2014.

[139] Yan Yan, Glenn M. Fung, Rómer Rosales, and Jennifer G. Dy. Active learning from crowds. In *Proc. International Conference on Machine Learning (ICML)*, 2011.

[140] Yan Yan, Rómer Rosales, Glenn Fung, Ramanathan Subramanian, and Jennifer Dy. Learning from multiple annotators with varying expertise. *Machine Learning*, 95(3):291–327, 2014.

[141] Yuchen Zhang, Xi Chen, Dengyong Zhou, and Michael I. Jordan. Spectral methods meet EM: A provably optimal algorithm for crowdsourcing. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2014.

[142] Dengyong Zhou, Sumit Basu, Yi Mao, and John C. Platt. Learning from the wisdom of crowds by minimax entropy. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2012.

[143] Dengyong Zhou, Qiang Liu, John Platt, and Christopher Meek. Aggregating ordinal labels from crowds by minimax conditional entropy. In *Proc. International Conference on Machine Learning (ICML)*, 2014.