SAND20XX-XXXXR
LDRD PROJECT NUMBER: 183780
LDRD PROJECT TITLE: Graph Learning in Knowledge Bases
PROJECT TEAM MEMBERS: Sean Goldberg, Daisy Zhe Wang

ABSTRACT:

The amount of text data has been growing exponentially in recent years, giving rise to automatic information extraction methods that store text annotations in a database. The current state-of-the-art structured prediction methods, however, are likely to contain errors and it's important to be able to manage the overall uncertainty of the database. On the other hand, the advent of crowdsourcing has enabled humans to aid machine algorithms at scale. As part of this project we introduced pi-CASTLE , a system that optimizes and integrates human and machine computing as applied to a complex structured prediction problem involving conditional random fields (CRFs). We proposed strategies grounded in information theory to select a token subset, formulate questions for the crowd to label, and integrate these labelings back into the database using a method of constrained inference. On both a text segmentation task over academic citations and a named entity recognition task over tweets we showed an order of magnitude improvement in accuracy gain over baseline methods.

INTRODUCTION:

In recent years, there has been an explosion of unstructured text data from social networks like Twitter and Facebook, within enterprises via emails and digitized documents, and across the Web. Information extraction (IE) over large amounts of text is important for applications that depend on efficient search and analysis, such as question answering, trend analysis, and opinion mining. Various types of structured information that can be extracted include content annotations from bibliographic citations and entity relationships from news articles.

Automatic information extraction can be viewed as a structured classification problem using statistical machine learning techniques. Given an input sentence x, the output label y has a rich internal structure. An example is a probabilistic sequence of an-notations for each word in the sentence. This approach of sequence learning has been the focus of much research into automatic IE for tasks such as text segmentation (TS) or named entity recognition (NER). The most common and state-of-the-art sequence model for these tasks is the linear-chain conditional random field (CRF) [Lafferty et al. 2001].

Because of the inherent uncertainty and fallibility of many machine learning algorithms, recent work has turned to the incorporation of a human element for correcting errors or validating output of machine results. Crowdsourcing platforms like Amazon Mechanical Turk (AMT) have made it possible to utilize human computation efficiently and cheaply. Nevertheless, human annotations are still much more expensive and time-consuming compared to algorithmic labeling [Mozafari et al. 2014] and care must be taken to optimize the work administered to the crowd.

Previous work in utilizing a hybrid of traditional and crowd computation for structured classification include entity resolution [Mozafari et al. 2014], web table [Fan et al. 2014] and ontology alignment [Sarasua et al. 2012], and probabilistic query processing [Ciceri et

al. 2016]. The main research challenges are optimizing data selection and question construction. Data selection involves targeting the most significant human contributions given a fixed budget of questions. Question construction is concerned with extracting the maximum possible information out of each question and is intimately connected to the data selection problem.

There has been little work in combining human and machine computation for text classification due to the complexity of the structured prediction models. Data selection and question construction are more difficult because they have to reason with the internal structure of the probabilistic graphical models.

In this paper, we build on existing work to develop an end-to-end system that not only tackles both of these selection problems, but fully integrates the crowdsourced response back into the machine model. Our system, pi-CASTLE is a crowd-assisted statistical machine learning (SML)-based IE system that uses a probabilistic database to execute, optimize, and integrate human and machine computation for improving text extraction and processing. pi-CASTLE initially employs a linear-chain CRF to annotate all input text data. In contrast to other IE systems, however, pi-CASTLE uses a probabilistic data model to store IE results and manage data cleaning. It has the ability to automatically query humans through the deployment of Amazon Mechanical Turk Human Intelligence Tasks (HITs) to correct the most uncertain and influential tokens and integrate their responses back into the data model.

By allowing trained algorithms to do most of the work and focusing on humans only in the "last mile", pi-CASTLE achieves an optimal balance between cost, speed, and accuracy for IE problems. We address three challenges in the design and implementation of pi-CASTLE: the probabilistic data model, selection of uncertain entries, and integration of human corrections.

First, in order to manage uncertainty associated with classification results and do data cleaning from within the database, a probabilistic data model and system is needed. We use the model described in [Wang et al. 2010b], storing both uncertain relations and probabilistic models as first class objects. We also implement user-defined functions (UDFs) for statistical inference, question selection, and uncertain data integration over this probabilistic data model to connect the SML and crowd components in pi-CASTLE.

The data cleaning process entails automatically evaluating tokens in terms of their information value to the rest of the database and generating questions based on the highest scoring tokens to be pushed to AMT. Information value of each token is determined by a set of information functions that optimize different metrics over the database. pi-CASTLE uses concepts from information theory to select either the most uncertain tokens or the ones likely to have the most influence on other tokens. This is a technically challenging task as tokens, represented as nodes in a graphical model, are not independent, but adhere to the dependence properties modeled by the CRF. Optimal selection is N P ^{PP} -hard in general [Krause and Guestrin 2009] and we develop a set of approximate scoring functions. We choose to perform data cleaning at the to-ken level instead of the sentence or document level in order to exploit a phenomenon known as correction propagation [Culotta et al. 2006], wherein influence can spread throughout the graphical model from a single observation. This allows pi-CASTLE to maximize the efficiency and value of the data cleaning process.

pi-CASTLE is also able to construct questions in such a way that they maximize the impact on selected tokens. By exploiting redundancies in token usage on a global scale across documents, pi-CASTLE is able to map many tokens to a single question to achieve the greatest "bang for our buck". As an example, if the crowd is able to correctly resolve the token Obama as a reference to a PERSON, then all entries containing Obama in a particular context can be updated to reflect this new information. This directly improves on many existing systems [Kondreddi et al. 2014] that correct at best a single entry at a time.

The final design challenge is how to adequately handle evidence that has been collected from the crowd. Because every question posed costs financial and temporal re-sources, a central theme of pi-CASTLE is getting the most impact out of every question. Because in a relational learning task the output of one example may influence that of "nearby" examples, we develop a constrained inference framework that can use answers provided from the crowd to improve the results of other entries in the database. Understanding Obama is a person may improve the machine's decision making on other related tokens such as Barack.

One key application of this work is in knowledge transfer of models from one domain to another. Many pre-trained off-the-shelf models give poor results when applied to a new domain or data that follow a different distribution and are expensive or impossible to re-train. We demonstrate in our experiments pi-CASTLE's ability to optimize the cost of this knowledge transfer process through appropriate balancing of the human and machine components.

DETAILED DESCRIPTION OF EXPERIMENT/METHOD:

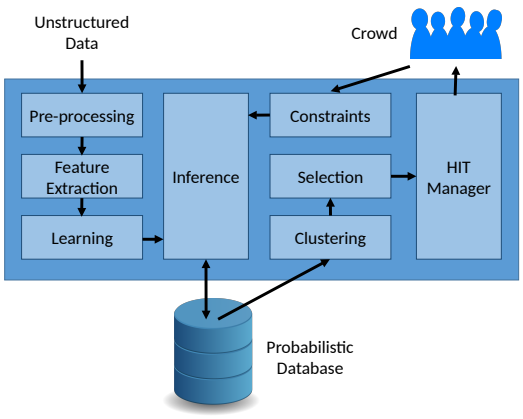


Figure 1: Architecture of the pi-CASTLE system.

Figure 1 outlines the basic architecture of the pi-CASTLE system, which can be conceptualized into four main components: (1) CRF Extraction & Inference, (2) Question Selection, (3) HIT Management, and (4) Human/Machine Integration. The arrows chart the flow of data within and between different components. Overall, data flows through the four components in order. First, the CRF model performs automatic text extraction and labeling. Both the extractions themselves and their associated uncertainties are stored in a probabilistic database. Next, a set of questions are generated as some function of the data uncertainty and given budget. The HIT manager formulates and pushes these questions to

the crowd and retrieves the answers. Finally, the Turker answers are integrated back into the database as a set of constraints on the inference that improve the initial results.

In this section we briefly outline each of the system's main components and how they are related, as well as how data is specifically stored in the data model through the use of the running example in Figure 2. While we use existing techniques for (1) CRF Extraction/Inference and (3) HIT Management, we develop novel techniques for (2) Question Selection and (4) Human/Machine Integration in Section 5 and Section 6.

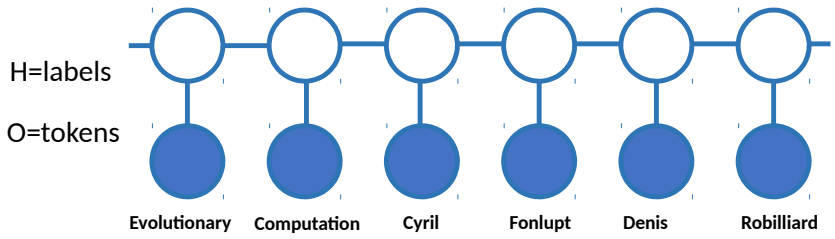
CRF Extraction & Inference

The initial machine approach to the structured prediction problem is handled by the components associated with the CRF, including pre-processing, feature extraction from the text, and model learning from a training set. The CRF model infers the hidden labels associated with each text token and stores the results in the database.

Unstructured text is treated as a set of documents or text string D. Each document $d \in D$ has a substructure comprising a set of tokens t_i^d, where $i \in \{1, \dots, N\}$ and N is the length of the string (document). For efficient and persistent data storage and retrieval, we store the tokens in a probabilistic database, adopting and expanding upon the data model outlined in [Wang et al. 2010a]. Each unique occurrence of a token, identified by a text-string ID (strID) and position (pos), is stored as a record in the relational table TOKENTBL. A TOKENTBL has the following schema:

TOKENTBL(strID, pos, token, label^p)

An example is shown in Figure 2.



strID	pos	token	label
…	…	…	
1	7	Evolutionary	
1	8	Computation	
1	9	Cyril	
1	10	Fonlupt	
1	11	Denis	
1	12	Robilliard	
…	…	…	

Fig. 2. An example CRF model and the associated storage format in the database. Observed nodes correspond to token values. Hidden nodes are probabilistic labels that contain a distribution over the label space.

Question Selection

The main contribution of pi-CASTLE is the ability to take an uncertain database as constructed using the CRF components and automatically improve it by pinpointing likely errors. In order to work within the constraints of a fixed budget, we need to select the most information-dense fields to correct. As we describe in more detail later in the paper, the relational structure of our model enables token inferences to affect other tokens through their dependencies. Additionally, highly redundant tokens have common contextual patterns across documents that allow fields to be mapped using the same human inferences.

Using these two notions, we develop a way to score each token in terms of its information density and select those most likely to have the strongest improvement on the database. Figure 4 shows a table view that includes the maximum likelihood label for each token and a score based on some scoring function. The goal of the Selection module is to evaluate tokens in such a way that their correction has a maximum influence on the database after integration.

In the discussion we examine two different optimization functions on the database and the in-formation functions that derive from them. One optimizes the reduction in uncertainty and suggests picking those tokens with the highest marginal entropy. The other maximizes the influence between the crowdsourced fields and the remaining fields and results in a mutual information calculation between tokens and their neighbors. In either case, we incorporate clustering into the scoring function so the most common errors have a higher weight in their selection and individual questions can be applied to many tokens simultaneously.

HIT Management

The HIT Management component has the responsibility of taking selected tokens, converting those tokens into questions in the form of Human Intelligence Tasks (HITs), and posting them onto Amazon Mechanical Turk. Part of the focus of pi-CASTLE is on reducing the problem of annotating an entire text string to annotating only specific to-kens at a time. The simplicity of this task avoids unneeded redundancy and translates into a simple question interface less prone to human-error.

An example interface is shown in Figure 3. The entire text document (in this case a citation) is shown with the query token bolded. Users select from the set of all la-bels the one they believe belongs to the bolded token. The brevity of each question allows bundling of multiple token annotations into a single HIT. For the time and cost of labeling a single unstructured text document from scratch, pi-CASTLE is able to acquire labels to the same number of super information-dense tokens which will have a much larger impact on improving the quality of the database.

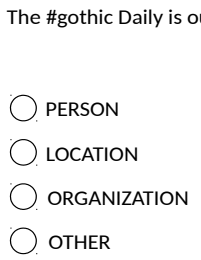


Fig. 3. Sample Mechanical Turk HIT Interface

Specific details of the AMT marketplace such as the price, length of posting, and number of Turkers assigned to each HIT are outside the focus of this paper. In practice, they would be set according to the constraints of the user.

Human/Machine Integration

The final component takes the human response to selected questions, aggregates their results, and integrates them into the final database. pi-CASTLE is agnostic to the method by which crowd results may be combined and any method which derives a single answer from a pool of possibilities may be employed. We found majority voting worked well enough to be useful due to high worker ability on text annotation tasks. For more difficult tasks with variable worker ability, pi-CASTLE is modular enough to utilize any number of quality control mechanisms as found in [Sheshadri and Lease 2013]

The key insight that distinguishes pi-CASTLE from other crowdsourced data cleaning systems is the treatment of new evidence as observed variables in the conditional random field inference process. CRF Inference finds a global best-fit path through the label space for each document. Fixing certain fields to the crowdsourced label con-strains the total label space and has a direct influence on tokens in the neighborhood of the observed one. The Constraints module stores the crowdsourced evidence and is used in subsequent inference passes over the data. The specific details of constrained CRF inference are detailed in Section 6.

Figure 4 shows a scenario in which identifying and correcting one node from a Title to an Author allows the machine to infer that the following token is also likely to be an Author. pi-CASTLE chooses selections wisely so as to maximize this degree of influence.

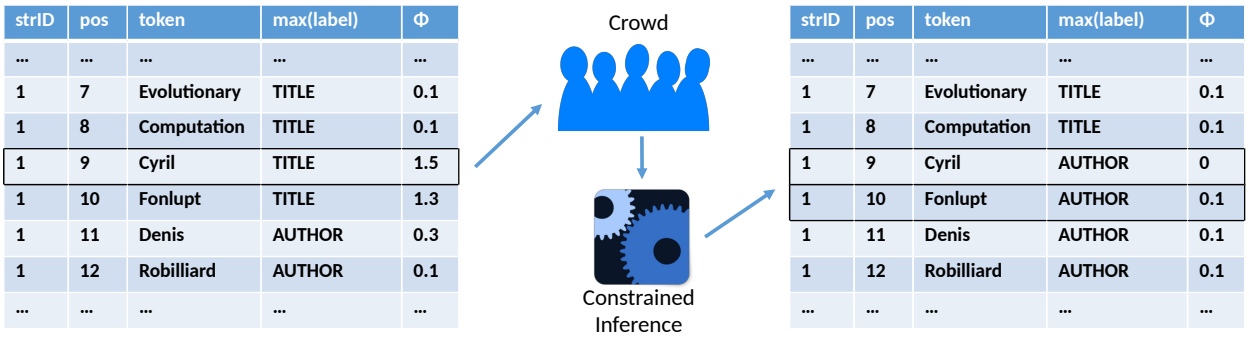


Fig. 4. Database view of the process of data selection and integration. An information function maps each token to a selection score. Here, because “Cyril” is an uncommon name, the machine confuses it as a Title. It has the highest scoring information function so it’s sent to the crowd for labeling where it’s confirmed to be an Author. Constrained inference propagates this information, changing “Fonlupt” to an Author as well

RESULTS:

In the following section we demonstrate the effectiveness of our approach applied to a text segmentation task and named entity recognition (NER) task. Given a semi-structured document like an academic citation string, text segmentation is a problem concerned with partitioning the string into different classes. This information can be used to build a database of records that allow for efficient search and analysis. We also test a NER task where the goal is to extract entities such as people, places, and organizations from tweets. NER can be distilled to an annotation task where every token is labeled as a specific entity type or else Other. NER is a common preprocessing step to higher-order semantic algorithms.

In order to fully show how our methods perform when the machine learning task is only partially accurate, we apply our training and testing sets to different datasets. This is a common practice in knowledge and learning transfer as well as in practical applications where vendors are required to use off-the-shelf models that are difficult or expensive to re-train. Our results do not apply only in this domain and could just as easily be applied to standard structured prediction tasks.

Our pipeline consists of two key phases for which we provide experimental evidence of their utility. The first phase involves selection of information-dense tokens that would be sent to the crowd to answer. We compare our entropy and mutual information approximation along with other selection baselines. In the integration phase the true label for selected tokens is applied and constrained inference is performed. By comparing directly to the unconstrained selection phase experiments, we demonstrate the performance gain coming directly from the influence of selected tokens.

In the following sections we describe the experimental setup and datasets in more detail before discussing the individual experiments.

Setup and Datasets

For training in the text segmentation task, we used the currently popular UMass citation dataset [Anzaroot and McCallum 2013]. State-of-the-art citation extraction has typically used the CORA dataset, which is much smaller and is being replaced by this current dataset. It contains 1800 bibliographic citations from physics, mathematics, computer science, and computational biology domains pulled from Arxiv.org and is fully-labeled with 32 fine-grained (FIRST NAME, LAST NAME, TITLE, etc.) fields.

The test set consists of 7K citations comprising 242K tokens extracted from the DBLP[2] database which contains primarily computer science papers. Each citation is represented as a row in the database with columns for the seven classes: TITLE, AUTHOR, CONFERENCE, SERIES, PROCEEDINGS, ISBN, and YEAR. We reproduced the original citation by concatenating all of the string text together as a single semi-structured document. From this we generated a training set and testing set based on a 50% split.

We used the IITB CRF[3] model for training and testing, which is heavily engineered to represent the state-of-the-art in the text segmentation task. The tokenizer it comes with includes commas and all numbers mapped to a single “DIGIT” token in the tokenization. We filtered these out for scoring.

Both UMass and DBLP sets contain citations from different domains and are structured in many different ways with differing pieces of information. While DBLP is more course than UMass, each of its labels maps onto either a direct label or set of labels in UMass. We computed this mapping in advance and translated the field sets prior to computing F1 scores.

For the named entity recognition task, we used the off-the-shelf Stanford NER[4] parser. This is the current state-of-the-art in NER and used widely throughout the literature. The model contains four classes (PERSON, LOCATION, ORGANIZATION, or OTHER) and is trained on the CoNLL 2003 shared task, which itself is a set of news wire articles from the Reuters corpus. This model performs best in its native news wire domain, but its commonly applied to other domains as well.

We wanted to see how the Stanford NER parser would apply to a Twitter domain, using the TwitterNLP[5] dataset, which consists of 2400 unstructured tweets comprising 34K total tokens. TwitterNLP contains many novel entities not found in the Stan-ford training set. In addition, the variability in punctuation and capitalization makes Twitter a very difficult domain for NER using purely machine learning. The original TwitterNLP included extraneous classes like Product or TV-Show, but we converted these to Other.

The goal of the selection problem is to select those tokens that are (1) the most likely to resolve errors by their own correction and (2) the most likely to improve incorrect neighbors through inference propagation. We evaluate by computing F1 scores on a token-by-token basis, where F1 is the harmonic mean of precision and recall. Precision is the ratio of correct tokens for a class to all predicted tokens for that class. Recall is the ratio of correct tokens to all true tokens for that class. Because both applications are instances of a multi-class classification problem, we use the micro-averaged F1. This uses the sums of all true positives, false positives, and true negatives to compute a final F1. Given a budget of K questions, we seek to maximize the F1 gain from an initial machine-learning only baseline we can achieve with each question. Experiments show F1 increases incrementally as questions are posed.

The experiments that follow contain both synthetic and real crowdsourced data. The synthetic data is designed to test the selection and integration mechanisms which are the focus of this paper independent of crowdsourcing uncertainty. We do this by substituting the true field in place of any selected tokens. It would also be untenable attempt as many permutations of the experiments as we require using only real data. The use of synthetic oracle data allows us to significantly scale up the experiments. We include a small set of end-to-end experiments where we crowdsourced the answers using Amazon Mechanical Turk to verify the ability of the crowd to perform this task.

Selection Experiments

For all experiments, we perform a filtering step prior to clustering or selection that reduces the pool of available tokens. For each citation in the DBLP dataset or tweet in the Twitter dataset, we select either the highest Mutual Information or Entropy token depending on the experiment being considered. For our random baseline, we randomly select a token from each document. This reduces the DBLP pool to 7000 tokens and the Twitter pool to 2400 tokens. Then we either select or cluster and select depending on the experiment.

Figures 5 and 6 show how the F1 scores are improved using different selection metrics. The plots are differentiated by permutations of dataset, clustering, and con-straining. Figure 10 shows experimental results for Umass to DBLP, while 11 shows results for newswire to Twitter. The initial F1 without any human correction is 33:8% for training on Umass and testing on DBLP, while for training on CoNLL 2003 and testing on TwitterNLP the initial F1 is 56:6%. While these scores may seem low, this is standard for a highly difficult knowledge transfer task and is the perfect application for pi-CASTLE to exploit machines performing one-half or one-third the task while humans fill in the gaps. The x-axis shows how the total system F1 increases for every question we ask. For unclustered experiments, this corresponds to only a single token correction in the entire dataset. For clustered experiments, each question maps on a cluster of tokens, all of whom are given the crowdsourced label.

The baseline for comparison without clustering is randomly selecting tokens (Rand) for improvement, bypassing the filtering and ranking steps. If we perform clustering, we choose a different baseline (Size that randomly selects a token from each citation (i.e. filtering) and then clusters them and ranks them according to size. This generates a much stronger baseline and directly compares a baseline data-centric algorithm with our information-theoretic algorithms. Ent selects tokens whose predicted marginal label distribution has the highest entropy in the filtering step and ranks them either by token entropy or total cluster entropy. Similarly, MI selects those tokens with the highest mutual information approximation as discussed in Section 5 and clusters and ranks in the same fashion.

The size of the DBLP test set is 242k tokens and without clustering, we're only at liberty to correct one token at a time per question asked. This results in mostly flat plots that don't at all differ by constraining. Clustering without constraining provides a marginal increase for both information-theoretic methods. Constraining the selected clusters, however, provides the largest increase across the board for all selection mechanisms. The performance of random is still quite good because when the F1 score is as poor as it is, even random corrections will result in some common incorrect to-kens being clustered and corrected. Entropy outperforms it by selecting tokens more likely to be incorrect and result in score increases upon correction. Mutual Information performs strongest here because it selects those tokens most likely to improve other tokens.

The performance on the Twitter test set is a bit closer for all plots compared to DBLP. There are two reasons for this. First, named entities appear more often as singleton mentions over a single token. This reduces the effect of constraining. Second, the in-creased variety among Twitter compared to semi-supervised citations leads to a reduction in the impact of clustering. Nevertheless, not performing any type of clustering or constraining is still the weakest across the board, while performing both clustering and constraining leads to the overall strongest performance. Selecting by entropy is best performer without constraining, but Mutual Information edges out slightly when constrained inference is performed. In all cases, the information-theoretic methods significantly outperform the random baselines.

There are a couple additional observations on both sets of plots. In the Twitter test-ing set, some of the graphs appear to dip despite getting a ground truth correction. There are two additional sources of error our methods introduce into the problem. One comes from errors in clustering, where incorrectly clustered tokens receive the same la-bel in error. Another comes from the constrained inference process, where neighboring tokens correctly labeled by chance are incorrectly labeled after applying corrections. Our experiments show both of these scenarios are rare and the benefits of clustering and constraining far outweigh the possible negative effects.

Of particular note is the small number of questions. For the DBLP set at 242k to-kens, 200 questions represents only 0:1% of the whole dataset and would cost only about $50 on Amazon Mechanical Turk. And yet using a Mutual Information frame-work with some question clustering we're able to improve the F1 score from 34% to 43%, a gain of nearly 33% of the original. This is much more efficient than selecting at the citation level for labeling, which requires more redundancy and is unable to take advantage of clustering. By allowing the machine to do most of the work, we can make the most necessary

improvements with a small number of questions. This shows that iterating on this framework with batches of K questions would lead to higher performance with lowered costs than a higher granularity labeling process.

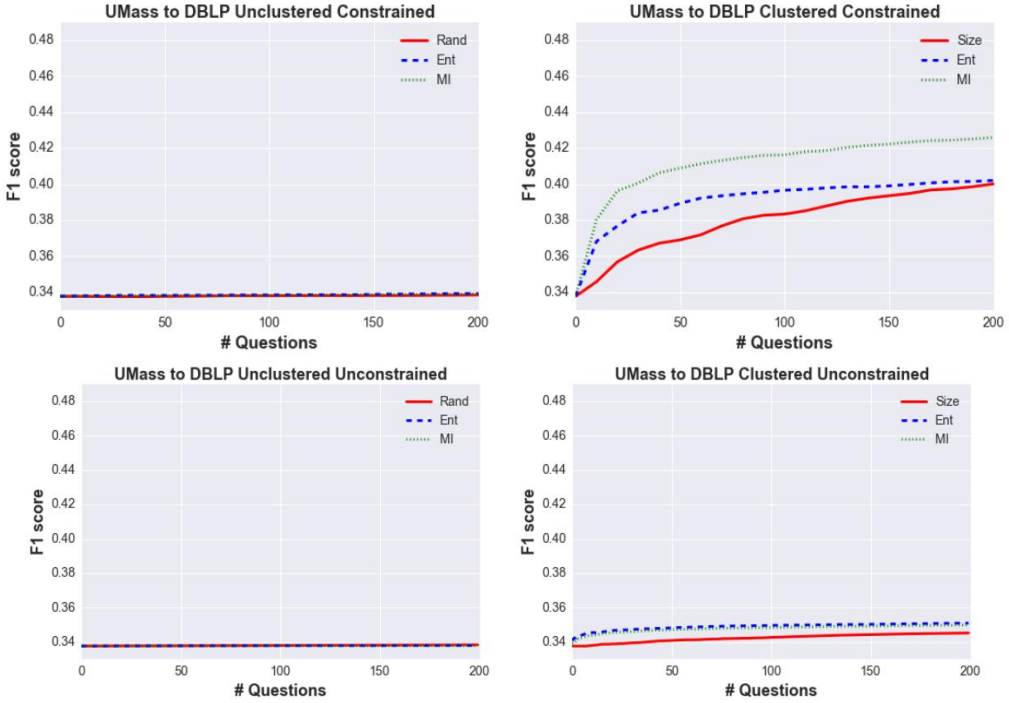


Fig. 5. F1 score increases for Umass training and DBLP testing.

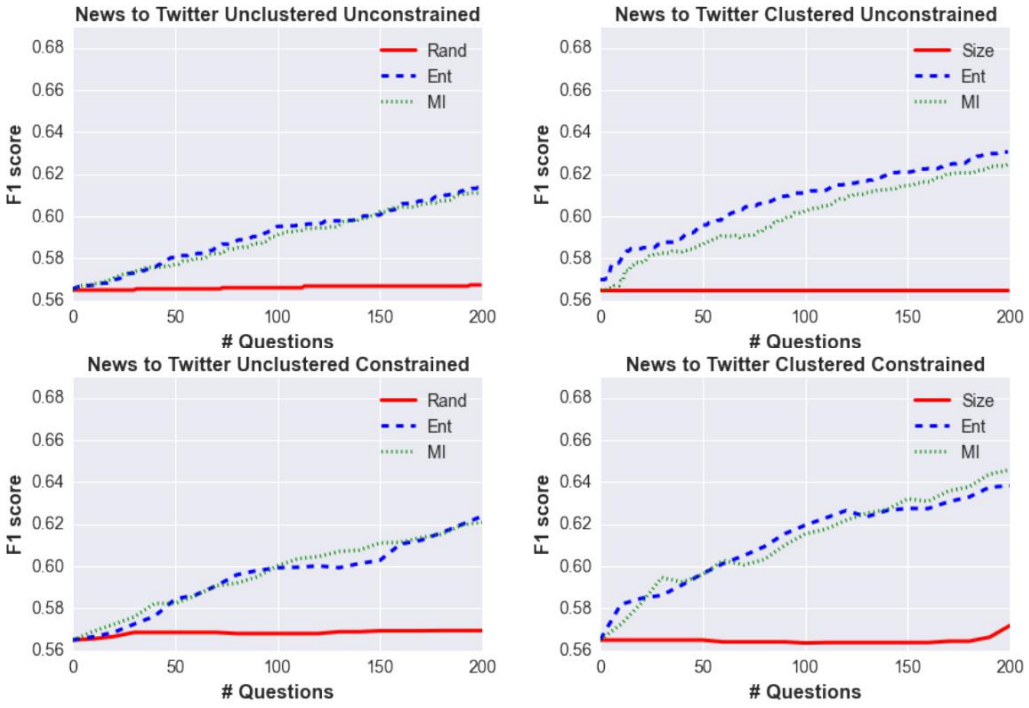


Fig. 6. F1 score increases for newswire training and Twitter testing.

DISCUSSION: (> 700 words with no upper limit)

In this section we formalize the question of optimally posing questions to the crowd by analyzing two key selection strategies in the context of which global optimization. functions they minimize. This leads us to strategies for either maximizing the uncertainty reduction of the system or the influence of the selected nodes. To maximize the information density of each question, we describe a contextual clustering technique that fold multiple uncertain examples into a single question.

A general probabilistic graph can be described simply as tuple $G = (V; E; P)$ that consists of nodes V, edges E, and a probability distribution P over those nodes and edges. In general, P may denote either a directed Bayesian network or an undirected Markov random field. Assume the V nodes are partitioned into a set of observed nodes O and hidden nodes H such that $O \cup H = V$. By virtue of having learned a model, there exists some P that can be used to make predictions on the distribution of values of H. Assume also that we are given a budget of K observations to make on the hidden nodes, after which we have a new partition between the observed hidden nodes Y, where $|Y| = K$, and the still unobserved hidden nodes X, such that $X \cup Y = H$. While this paper focuses on a specific probabilistic graph known as a CRF, the problem we're posing is a more general one applied to any probabilistic graph.

Question Selection Problem: Given a budget of $|Y| = K$ questions, how do we pose questions to the observer in such a way that the resulting observations Y minimize the prediction error on the remaining hidden nodes X?

The mapping of questions to observable nodes need not be one-to-one and the problem decomposes into using questions to trade off both the largest mapping from a question to many nodes and the individual information density of those nodes. The former is related to the content and redundancy of connected observations and the latter deeply related to the structure of the graph.

The Question Selection Problem is similar to that found in active learning where select examples are chosen from a pool of unlabeled data to be annotated based on some querying strategy. Traditionally these examples are independently and identically distributed (IID) and take no account of the causal influence of one example's label on another. While active learning has been applied to the structure prediction domain [Settles and Craven 2008; Cheng et al. 2008], the examples primarily consist of individual independent graphs and in each case the entire graph is labeled. This reduces the problem to labeling a set of IID examples. The Question Selection Problem is concerned with labeling individual hidden nodes in a larger graph structure. Given that most examples contain sparse labeling errors, we hope to achieve maximum efficiency in terms of financial and temporal cost while also being more amenable to AMT's micro-task framework.

Our approach to solving the Question Selection Problem is to phrase it as an optimization function $g(X; Y; O)$ over the graph. From the collection of hidden nodes H, we seek to find the set of new observations Y that maximizes $g(X; Y; O)$.

$$Y^* = arg \max_Y g(X, Y, O)$$

We discuss two possible optimization functions in the succeeding sections and approximate solutions to solving them. A key requirement in scaling to large graphs is the ability to

evaluate and rank nodes independently by some information function such that the optimal Y are just the Top-k scoring nodes $Y_{\{k\}}$. These information functions have been previously published as heuristics in [Goldberg et al. 2013] and in this paper we put them on a more sound theoretical footing.

Our primary purpose is to establish and make progress on the Question Selection Problem as it pertains to text-based linear-chain CRFs which have wide applicability in natural language processing applications.

5.2. Uncertainty Reduction

Most classifiers give as output both a class prediction and confidence score. The confidence score measures how difficult the machine finds the problem to be and subsequently how confident it is in its output. It is reasonable to assume then that for properly trained classifiers there is some correlation between prediction accuracy and confidence. For structured prediction problems, the confidence is typically re-normalized as a probability distribution over all possible joint distributions of the hidden nodes. One strategy for improving prediction performance is to try to reduce the uncertainty associated with the output distribution.

Uncertainty can be modeled using entropy. The entropy of a distribution of possible outputs T is defined as

$$H(T) = -\sum_{t \in T} P(t) \log P(t)$$

When the distribution is peaked towards some particular output value t we say the system has low uncertainty and is confident in that t is the correct output. If the distribution is spread over a wider range of possible values, the system is characterized as having high uncertainty.

After the initial machine prediction phase, the system of hidden nodes $H = X \cup Y$ has some associated uncertainty $H(X; Y|O)$. After selecting Y nodes for observation, the remaining uncertainty is $H(X|Y; O)$. One way of minimizing final prediction er-ror is to observe those nodes that give the largest reduction between $H(X; Y|O)$ and $H(X|Y; O)$. Thus the optimization function is

$$g(X, Y, O) = H(X, Y|O) - H(X|Y, O)$$

Using the identity

$$H(X, Y|O) = H(Y|O) + H(X|Y, O)$$

this uncertainty reduction is equivalent to maximizing the marginal entropy of the selected nodes $H(Y|O)$. Solving this problem exactly requires calculating the joint distribution of all possible subsets of the budget size K. As a simplifying assumption, we relax the notion of connectivity between nodes.

For a set of independent random variables $Y_1, \ldots, Y_K$, the entropy can be written as the sum of the individual marginal entropies

$$H(Y_1, \ldots, Y_k|O) = H(Y_1|O) + \cdots + H(Y_k|O)$$

Under this independence assumption, maximization of $H(Y|O)$ is equivalent to selecting the individual $H(Y_{\{k\}}|O)$ that have the largest individual marginal entropies given the observed nodes. This results in an information function

$$\phi_{ENT}(Y_k) = H(Y_k|O)$$

We refer to this information function as the token entropy.

Token entropy has appeared elsewhere in the literature where it is referred to as uncertainty sampling [Lewis and Gale 1994]. Token entropy is ideal for rooting out specific individual tokens that the machine has difficulty classifying. The fundamental shortcoming is that for a structured prediction problem, it's unable to take the data's connectivity into account. Even without relaxing the dependency assumptions, maximization of $H(Y|O)$ in no way takes into account how the newly observed nodes Yare related to the still-unobserved nodes X. In the next section we introduce a novel way of incorporating token dependencies into the Question Selection Problem using the concept of mutual information.

Influence Maximization

Due to the dependence properties of certain nodes in the graph on other nodes, an ideal selection strategy would take into account the influence an observation has on its surrounding neighborhood. Mutual information (MI) is a pairwise metric between random variables that quantifies how much the uncertainty of one is reduced when the other is observed.

Specifically, for two sets of random variables X and Y, we can define the mutual information between them in terms of their entropies as

$$I(X;Y|O) = H(X|O) + H(Y|O) - H(X,Y|O)$$

It represents the difference between the joint entropy $H(X;\, Y|O)$ and the individual entropies $H(X|O)$ and $H(Y|O)$. Random variables that are highly uncorrelated will have a joint entropy equivalent to the sum of their entropies and thus zero information. On the other hand, highly correlated variables will have large degrees of mutual information.

In terms of the Question Selection Problem, we'd like to select those variables Y which, once observed, have the largest "effect" on the remaining variables X. The impact of observation on surrounding random variables was discussed in Section 4 in reference to the Viterbi algorithm since each label depends on the labels of its neighbors. Thus we are concerned with optimizing the difference in uncertainty between the variables X initially and those same variables conditioned on the selected variables Y, given the original observed variables O. The optimization function for this strategy becomes

$$g(X,Y,O) = H(X|O) - H(X|Y,O)$$

Using the identity for conditional entropy

$$H(X|Y,O) = H(X,Y|O) - H(Y|O)$$

as well as the definition of MI above, it's clear that this is precisely equivalent to optimizing the mutual information between the newly observed variables Y and remaining variables X.

This problem is in some sense "harder" than the uncertainty reduction problem of the previous section. As before, we have to calculate all possible partitions of variables into X and Y and calculate marginal entropies. Here, we also have mutual entropies to calculate and the problem cannot be reduced by relaxing dependency properties. In fact, if we do assume independence we lose the entire ability to reason using mutual information.

Instead we rely on a different approximation strategy, exploiting the structural properties of the graph. Given that all hidden nodes are composed as a linear-chain, the bulk of influence a node has is purely to its two neighbors. This is due to the Data Processing Inequality [Kinney and Atwal 2014], which says states that information along a Markov chain can only decrease. As a first-order approximation we consider only this influence, for each node calculating the mutual information between it and its neighbors:

$$g(X,Y,O) = \sum_{Y_k \in Y} I(Y_k; neighbors(Y_k)|O)$$
$$= \sum_{Y_k \in Y} I(Y_k, left(Y_k)|O) + (Y_k, right(Y_k)|O)$$

where $left(Yk)$ refers to the neighbor preceding Y_k in the chain and $right(Yk)$ refers to the neighbor succeeding it. This results in a simple information function define by mutual information:

$$\phi_{MI}(Y_k) = (Y_k, left(Y_k)|O) + (Y_k, right(Y_k)|O)$$

Mutual information can be useful in determining the impact a node's observation has on other nodes within an individual sequence, but tells us nothing about the distribution of tokens across all documents. If we want to optimize our selection strategy, especially for a batched selection process, we should additionally incorporate a token's frequency and its influence/uncertainty.

Clustering by Information Density

In addition to selecting tokens that exhibit either the highest uncertainty or largest influence, we'd like to construct questions in such a way that a single question can have maximum impact on the whole of the database. Equivalently, we'd like the questions posed to be varied enough that we're not wasting financial resources asking the same question twice. A data-driven solution is to utilize simple clustering to group tokens together and map individual questions onto entire clusters instead of single tokens. As we show in our experiments, in tasks such as text segmentation and named entity recognition there are many redundant tokens that produce clusters of large size.

Our strategy for clustering is to collect tokens that should be labeled the same based on label criteria for our model. Since the CRF model contains features $F_J(H_{I+1}\,;\, H_I;\, O_I)$ that depend on

observations at each token and their labeling neighborhood, we utilize the same aspects as a means for clustering. The label trigram method operates after the initial machine prediction and clusters together tokens H_i that that have the same previous and succeeding prediction labels (H_{i-1} ; H_{i+1}) as well as the same observation token O_i. Though the features are sequential and don't consider succeeding tokens H_{i+1}), we include it in the clustering to differentiate tokens further and reduce possible errors. An example is shown in Figure 6 which clusters five documents into two clusters based on their appearing in either the TITLE or the SERIES as compared to a typical token trigram approach. Clustering algorithms are furthered compared with experimental results in [Goldberg et al. 2013] with the conclusion that label trigrams produce superior results.

When tokens are selected for crowdsourcing, the entire context of the token is dis-played to the user. Since each item in a cluster has its own independent context, we select a specific token at random to be the representative token for that cluster. When mapping clusters to questions, the representative token is the one specifically used to provide context and to formulate the question. When retrieving the answer to a question, the observed label is applied to all tokens in the representative token's cluster. The effect of clustering and constrained inference influences many tokens with only a single question.

The final concern is how to map the selection strategies introduced earlier in this section onto entire clusters. We looked at a number of strategies for aggregating $_{ENT}$ and $_{MI}$ including taking the max information function in the cluster (MAX), taking the average information function (AVG), and taking the sum of all information functions (SUM). Since clusters are unevenly distributed in size, we found a metric that takes into account cluster size to be preferable to one that does not. Therefore, taking the sum of all information functions in a cluster reflects both the high information redundancy (size of the cluster) and high uncertainty/influence (value of information function) in each question.

ALGORITHM 1: Selection algorithm.

input : Set of all tokens T
output: Ranked set C of maximum information clusters

```
1 Initialize selected token set S;
2 Initialize cluster set C;
3 foreach t 2 T do
      //Apply information function;
4     t:info(t);
      //Clustering;
5     Add t to cluster c(t; t:label; t:prev label; t:post label);
6     if size(c) == 1 then
          c:rep_token    t;
8     c:totalInfo   c:totalInfo + t:info;
9 SORT clusters c ∈ C by c.totalInfoGain;
```

Algorithm 1 reviews the basic selection strategy for applying information functions to clusters and ranking them to determine the Top-k questions. We first iterate through all tokens in an initial pass applying the requisite information function. Since both $_{ENT}$ and $_{MI}$ employ simple approximate entropy calculations, they can be computed in constant time with respect to the token space. During the same iteration pass, to-kens are hashed to a cluster according to the 4-tuple ($H_{i\ 1}$; H_i; H_{i+1}; O_i). These correspond to the token and its label as well as its neighbors' labels. The first token put into a cluster is mad made the representative token and all subsequent tokens contribute to a running sum of the information function associated with each cluster. Finally, the clusters are ordered by their total information functions and selected based on the budget to be mapped into questions.

Built on
LDRD
Laboratory Directed Research and Development

ANTICIPATED IMPACT:

We have begun developing and applying our methods to more general graphs in probabilistic knowledge base completion.

There has been a vast amount of research in generating structured knowledge bases from unstructured and semi-structure data on the web. This has produced several large-scale KBs able such as YAGO, NELL, and OpenIE able to store vast amounts of data for querying and analysis. Limitations of the extraction process are such that only information explicitly written down has the potential to be extracted and stored in the KB. There is a wealth of untapped implicit information and higher order knowledge that can be used to augment and extend existing KBs.

Figuring out this additional information is known as the Knowledge Expansion problem and the higher order information is typically in the form of learned inference rules. Application of these rules to an existing KB results in new information that can be added to the KB. This information can be noisy, owing to uncertainty in both the rule itself and the approximation scheme. Uncertainty in the rule comes from applying soft rules that may only hold or be broken in certain situations. Uncertainty in the approximation scheme depends on whether top-down or bottom-up inference is being applied. Top-down inference resolves all possible facts as random variables and solves a constrained optimization problem using MCMC, which produces only approximate solutions for large graphs. Bottom-up inference requires local application of the inference rules, adding new facts in iterative batches. Such a scheme is unable to take into account more global properties of the inference like functional constraints. The compounding of such probabilistic errors results in low precision on the final set of returned facts.

The availability of human feedback enables the correction of some of these errors at a large financial and temporal cost typical of human input. This cost can be controlled and minimized if human feedback is used efficiently to target the areas of the KBs most need of correcting while allowing machine algorithms to infer what remains. If the knowledge base is modeled as a graph with nodes representing facts and edges representing inference rules, the problem amounts to selecting an optimal set of nodes query nodes to submit for human feedback.

This human-machine hybridization is designed to raise inference precision and control the propagation of errors in the knowledge graph inference at near-optimal cost. Using the theory of Value of Information in probabilistic graphical models, we can select those facts whose knowledge gives the largest return with the respect to the remaining state of the knowledge graph. Seminal work has already been done in deriving optimal or near-optimal algorithms and performance guarantees for both chain and general graphical models. We foresee extending this theory into the domain of probabilistic knowledge graphs.

Following notation similar to Chen et al. (2014), a probabilistic knowledge base is defined as a 5-tuple $\Gamma = (E, C, R\}, \Pi, L)$ of entities, classes, relations, probabilistic facts, and rules. Entities E are real-world objects such as people, locations, and things. Each entity $e \in E$ can belong to one

.

or more classes or types T. It's natural to think of the set of classes C as a subset of the entities E. Entities have relationships $r \in R$ with other entities, which define pairwise relations $r(e_i, e_j)$ between entities. An entity-level graph has nodes representing entities E and edges representing relations R. A fact F is a tuple $r(e_i, e_j)$ where $r(e_i, e_j) \in R$, $e_i \in E$, and $e_j \in E$.

In a probabilistic knowledge base, facts are weighted by probabilities corresponding to their perceived veracity. A probabilistic fact $\pi = (F, p)$ contains a fact tuple and its associated probability $p \in [0,1]$. Higher order knowledge about the facts are encoded in a set of weighted inference rules L. For the purposes of this paper, we examine two types of inference rules R of length 2 and length 3:

$$F_i \rightarrow F_j \quad (1)$$
$$F_i \,\&\, F_j \rightarrow F_k \quad (2)$$

here each F stands for a fact or its negation. A soft or probabilistic inference rule $l = (R, w) \in L$ has an associated weight that determines how likely that rule is to hold. Inference over the knowledge base uses the probabilistic rule to generate additional probabilistic facts. A fact-level graph represents individual facts as nodes and the rules between them as factors in an undirected factor graph. Eq. 1is modeled using pairwise factors and Eq 2 uses 3-way factors.

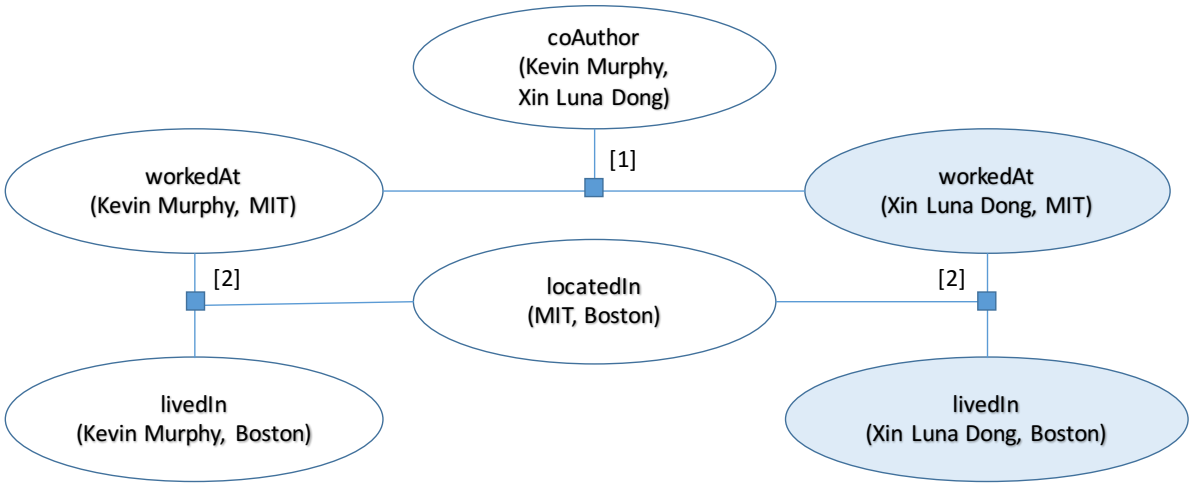
Inference rules may be applied in a top-down or bottom-up formulation. The top-down method enumerates all possible new triplets of entities and relations and samples from their distribution using techniques like MCMC to reduce the number of violated rules. Top-down solutions are limited by the convergence speed, which depends on the size of the graph, and can produce only approximate noisy solutions. The bottom-up or iterative method expands the graph by iterative application of the inference rules, also known as rule propagation. For each satisfied rule on the left hand side of Eqs, 1 or 2, the fact on the right hand side is added to the KB with some probability. The rules are thus used to propagate or grow the graph with new facts. State-of-the-art rule propagation methods use logistic regression to combine the input of multiple rules and output a posterior probability of the inferred fact. The iterative method scales better to larger data sets than the top-down method and will be the focus of this proposal.

Both ways of performing inference introduce errors into the knowledge graph. Because rules are probabilistic, they will not always apply and an inference error in one fact can propagate itself to produce noise in the rest of the graph.

An example of this error propagation is depicted in the fact-level graph in Figure 7. The graph reasons about basic facts over people Kevin Murphy and Xin Luna Dong and their institutions and locations. Edges connecting nodes are factors representing two different rules. The first says that two authors are likely to work at the same institution. The second that people are likely have lived where they attended school. These rules are soft and are only true with some probability. It is in fact true that despite being co-authors, Kevin Murphy and Xin Luna Dong did not attend

.

the same institution. In addition to providing this incorrect fact, the error propagates to incorrectly infer where Xin Luna Dong has lived as well.



[1] coAuthor(X,Y) ^ workedAt(Y,Z) -> workedAt(X,Z)

[2] WorkedAt(X,Y) ^ locatedIn(Y,Z) -> livedIn(X,Z)

Figure 7: Example Fact-level Knowledge Graph. Nodes represent facts and edges higher-order rules over those facts. Because rules are probabilistic, they may contain errors that propagate through the graph. In this example, the two shaded nodes represent incorrectly inferred facts.

In order to stem the propagation of errors, we'd like utilize correction via human feedback. An inference algorithm is constrained only by the knowledge in its own database, but humans can retrieve additional information to improve prediction accuracy. While it's an inefficient use of resources to have a human verify every possible true or false fact in a knowledge base, it is possible to guide labeling to those facts that are deemed most important by some metric. In general, given a knowledge graph $G = \{V, E\}$, we'd like to select a subset of those facts $A \in V$ that provide the greatest overall benefit, whether that be current prediction accuracy or future performance of the inference algorithm.

Each possible selection of A derives a certain reward score $R(A)$, where R is a set function that maps each subset A into a real number and represents the optimization function to maximize. Presumably each selection also incurs some cost of observation $c(A)$ that can be measured in terms of temporal or financial resources. For example, a set of facts containing many of the same entities will take less time to verify than a large set of disparate facts. There may also be different ways of asking questions that map onto node observations that incur different costs.

Using this notion of reward and cost, the overall goal is to find an optimal set A^* that solves the optimization problem

.

$$A^* = arg\max_{A \subset V} R(A)\ s.t\ c(A) \leq B.$$

where B is a budget that can be spent for selecting nodes.

This problem is NP-hard for most classes of reward and cost functions. We expect to utilize the methods learned in this report for improving upon this problem.

CONCLUSION:

In this report we introduced pi-CASTLE, a crowd-assisted SML-based IE system that can improve the accuracy of its automated results through a crowdsourced workforce. We developed two information functions and a clustering heuristic to formulate the most information-dense questions to the crowd given a fixed budget. Our experiments showed order-of-magnitude performance increases for a given set of questions compared to baselines.

While we focus on text extraction in the paper, we envision a more general Crowd-Assisted Machine Learning (CAMeL) system that uses a probabilistic database to efficiently connect and integrate crowdsourcing to improve the imperfect results from SML methods. Many of the core elements developed in pi-CASTLE such as uncertainty management, question selection, and human/machine integration are applicable to other SML-based tasks in the CAMeL framework.

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525.

.