A Probabilistic Model of Redundancy in Information Extraction

Doug Downey, Oren Etzioni, and Stephen Soderland
Department of Computer Science and Engineering
University of Washington
Seattle, WA 98195-2350
{ddowney,etzioni,soderlan}@cs.washington.edu

Abstract

Unsupervised Information Extraction (UIE) is the task of extracting knowledge from text without using hand-tagged training examples. A fundamental problem for both UIE and supervised IE is assessing the probability that extracted information is correct. In massive corpora such as the Web, the same extraction is found repeatedly in different documents. How does this redundancy impact the probability of correctness?

This paper introduces a combinatorial "balls-and-urns" model that computes the impact of sample size, redundancy, and corroboration from multiple distinct extraction rules on the probability that an extraction is correct. We describe methods for estimating the model's parameters in practice and demonstrate experimentally that for UIE the model's log likelihoods are 15 times better, on average, than those obtained by Pointwise Mutual Information (PMI) and the noisy-or model used in previous work. For supervised IE, the model's performance is comparable to that of Support Vector Machines, and Logistic Regression.

1 Introduction

Information Extraction (IE) is the task of automatically extracting knowledge from text. Unsupervised IE (UIE) is IE in the absence of hand-tagged training data. Because UIE systems do not require human intervention, they can recursively discover new relations, attributes, and instances in a rapid, scalable manner as in KNOWITALL [Etzioni *et al.*, 2004; 2005].

A fundamental problem for both supervised IE and UIE is assessing the probability that extracted information is correct. As explained in Section 5, previous IE work has used a variety of techniques to address this problem, but has yet to provide an adequate formal model of the impact of redundancy—repeatedly obtaining the same extraction from different documents—on the probability of correctness. Yet in massive corpora such as the Web, redundancy is one of the main sources of confidence in extractions.

An extraction that is obtained from multiple, distinct documents is more likely to be a bona fide extraction than one obtained only once. Because the documents that "support" the extraction are, by and large, independently authored, our confidence in an extraction increases dramatically with the number of supporting documents. But by how much? How do we precisely quantify our confidence in an extraction given the available textual evidence?

This paper introduces a combinatorial model that enables us to determine the probability that an observed extraction is correct. We validate the performance of the model empirically on the task of extracting information from the Web using KNOWITALL.

Our contributions are as follows:

1. A formal model that, unlike previous work, explicitly models the impact of sample size, redundancy, and different extraction rules on the probability that an extraction is correct. We analyze the conditions under which the model is applicable, and provide intuitions about its behavior in practice.
2. Methods for estimating the model's parameters in both the UIE and supervised IE tasks.
3. Experiments that demonstrate the model's improved performance over the techniques used to assess extraction probability in previous work. For UIE, our model is a factor of 15 closer to the correct log likelihood than the noisy-or model used in previous work; the model is 20 times closer than KNOWITALL's Pointwise Mutual Information (PMI) method [Etzioni *et al.*, 2004], which is based on Turney's PMI-IR algorithm [Turney, 2001]. For supervised IE, our model achieves a 19% improvement in average log likelihood over the noisy-or model, but is only marginally better than SVMs and logistic regression.

The remainder of the paper is organized as follows. Section 2 introduces our abstract probabilistic model, and Section 3 describes its implementation in practice. Section 4 reports on experimental results in four domains. Section 5 contrasts our model with previous work; the paper concludes with a discussion of future work.

2 The Urns Model

Our probabilistic model takes the form of a classic "balls-and-urns" model from combinatorics. We first consider the

single urn case, for simplicity, and then generalize to the full multiple *Urns Model* used in our experiments. We refer to the model simply as URNS.

We think of IE abstractly as a generative process that maps text to extractions. Extractions repeat because distinct documents may yield the same extraction. For example, the Web page containing "Scenic towns such as Yakima..." and the Web page containing "Washington towns such as Yakima..." both lead us to believe that Yakima is a correct extraction of the relation `City(x)`.

Each extraction is modeled as a labeled ball in an urn. A *label* represents either an instance of the target relation, or an error. The information extraction process is modeled as repeated draws from the urn, with replacement. Thus, in the above example, two balls are drawn from the urn, each with the label "Yakima". The labels are instances of the relation `City(x)`. Each label may appear on a different number of balls in the urn. Finally, there may be balls in the urn with *error labels* such as "California", representing cases where the IE process generated an extraction that is *not* a member of the target relation.

Formally, the parameters that characterize an urn are:

- C – the set of unique target labels; $|C|$ is the number of unique target labels in the urn.
- E – the set of unique error labels; $|E|$ is the number of unique error labels in the urn.
- $num(b)$ – the function giving the number of balls labeled by b where $b \in C \cup E$. $num(B)$ is the multi-set giving the number of balls for each label $b \in B$.

Of course, IE systems do not have access to these parameters directly. The goal of an IE system is to discern which of the labels it extracts are in fact elements of C, based on repeated draws from the urn. Thus, the central question we are investigating is: *given that a particular label x was extracted k times in a set of n draws from the urn, what is the probability that $x \in C$?*

In deriving this probability formally below, we assume the IE system has access to multi-sets $num(C)$ and $num(E)$ giving the number of times the labels in C and E appear on balls in the urn. In our experiments, we provide methods that estimate these multi-sets in both unsupervised and supervised settings. We can express the probability that an element extracted k of n times is of the target relation as follows:

First, we have that

$$P(x \text{ appears } k \text{ times in } n \text{ draws} | x \in C) = \sum_r \binom{n}{k} \left(\frac{r}{s}\right)^k \left(1 - \frac{r}{s}\right)^{n-k} P(num(x) = r | x \in C)$$

where s is the total number of balls in the urn, and the sum is taken over possible repetition rates r.

Then we can express the desired quantity using Bayes Rule:

$$P(x \in C | x \text{ appears } k \text{ times in } n \text{ draws}) = \frac{P(x \text{ appears } k \text{ times in } n \text{ draws} | x \in C) P(x \in C)}{P(x \text{ appears } k \text{ times in } n \text{ draws})} \quad (1)$$

Note that these expressions include prior information about the label x – for example, $P(x \in C)$ is the prior probability that the string x is a target label, and $P(num(x) = r | x \in C)$ represents the probability that a target label x is repeated on r balls in the urn. In general, integrating this prior information could be valuable for IE systems; however, in the analysis and experiments that follow, we make the simplifying assumption of uniform priors, yielding the following simplified form:

Proposition 1

$$P(x \in C | x \text{ appears } k \text{ times in } n \text{ draws}) = \frac{\sum_{r \in num(C)} (\frac{r}{s})^k (1 - \frac{r}{s})^{n-k}}{\sum_{r' \in num(C \cup E)} (\frac{r'}{s})^k (1 - \frac{r'}{s})^{n-k}}$$

2.1 The Uniform Special Case

For illustration, consider the simple case in which all labels from C are repeated on the same number of balls. That is, $num(c_i) = R_C$ for all $c_i \in C$, and assume also that $num(e_i) = R_E$ for all $e_i \in E$. While these assumptions are unrealistic (in fact, we use a Zipf distribution for $num(b)$ in our experiments), they are a reasonable approximation for the majority of labels, which lie on the flat tail of the Zipf curve.

Define p to be the precision of the extraction process; that is, the probability that a given draw comes from the target relation. In the uniform case, we have:

$$p = \frac{|C| R_C}{|E| R_E + |C| R_C}$$

The probability that a *particular* element of C appears in a given draw is then $p_C = \frac{p}{|C|}$, and similarly $p_E = \frac{1-p}{|E|}$.

Using a Poisson model to approximate the binomial from Proposition 1, we have:

$$P(x \in C | x \text{ appears } k \text{ times in } n \text{ draws}) \approx \frac{1}{1 + \frac{|E|}{|C|} (\frac{p_E}{p_C})^k e^{n(p_C - p_E)}} \quad (2)$$

In practice, the extraction process is noisy but informative, so $p_C > p_E$. Notice that when this is true, Equation (2) shows that the odds that $x \in C$ increase exponentially with the number of times k that x is extracted, but also decrease exponentially with the sample size n.

A few numerical examples illustrate the behavior of this equation. The examples assume that the precision p is 0.9. Let $|C| = |E| = 2,000$. This means that $R_C = 9 \times R_E$—target balls are nine times as common in the urn as error balls. Now, for $k = 3$ and $n = 10,000$ we have $P(x \in C) = 93.0\%$. Thus, we see that a small number of repetitions can yield high confidence in an extraction. However, when the sample size increases so that $n = 20,000$, and the other parameters are unchanged, then $P(x \in C)$ drops to 19.6%. On the other hand, if C balls repeat much more frequently than E balls, say $R_C = 90 \times R_E$ (with $|E|$ set to 20,000, so that p remains unchanged), then $P(x \in C)$ rises to 99.9%.

The above examples enable us to illustrate the advantages of URNS over the noisy-or model used in previous work [Lin

et al., 2003; Agichtein and Gravano, 2000]. The noisy-or model assumes that each extraction is an independent assertion, correct a fraction p of the time, that the extracted label is "true." The noisy-or model assigns the following probability to extractions:

$$P_{noisy-or}(x \in C | x \text{ appears } k \text{ times}) = 1 - (1-p)^k$$

Therefore, the noisy-or model will assign the same probability— 99.9%—in *all three* of the above examples. Yet, as explained above, 99.9% is only correct in the case for which $n = 10,000$ and $R_C = 90 \times R_E$. As the other two examples show, for different sample sizes or repetition rates, the noisy-or model can be highly inaccurate. This is not surprising given that the noisy-or model ignores the sample size and the repetition rates. Section 4 quantifies the improvements obtained by URNS in practice.

2.2 Applicability of the Urns Model

Under what conditions does our redundancy model provide accurate probability estimates? First, labels from the target set C must be repeated on more balls in the urn than labels from the E set, as in Figure 1. The shaded region in Figure 1 represents the "confusion region" – some of the labels in this region will be classified incorrectly, even by the ideal classifier with infinite data, because for these labels there simply isn't enough information to decide whether they belong to C or E. Thus, our model is effective when the confusion region is relatively small. Secondly, even for small confusion regions, the sample size n must be large enough to approximate the two distributions shown in Figure 1; otherwise the probabilities output by the model will be inaccurate.

An attractive feature of URNS is that it enables us to estimate its expected recall and precision as a function of sample size. If the distributions in Figure 1 cross at the dotted line shown then, given a sufficiently large sample size n, expected recall will be the fraction of the area under the C curve lying to the right of the dotted line.

For a given sample size n, define τ_n to be the least number of appearances k at which an extraction is more likely to be from the C set than the E set (given the distributions in Figure 1, τ_n can be computed using Proposition 1). Then we have:

$$\mathbf{E}[TruePositives] = |C| - \sum_{r \in num(C)} \sum_{k=0}^{\tau_n - 1} \binom{n}{k} \left(\frac{r}{s}\right)^k \left(1 - \frac{r}{s}\right)^{n-k}$$

where we define "true positives" to be the number of extracted labels $c_i \in C$ for which the model assigns probability $P(c_i \in C) > 0.5$.

The expected number of false positives is similarly:

$$\mathbf{E}[FalsePositives] = |E| - \sum_{r \in num(E)} \sum_{k=0}^{\tau_n - 1} \binom{n}{k} \left(\frac{r}{s}\right)^k \left(1 - \frac{r}{s}\right)^{n-k}$$

The expected precision of the system can then be approximated as:

$$\mathbf{E}[Precision] \approx \frac{\mathbf{E}[TruePositives]}{\mathbf{E}[FalsePositives] + \mathbf{E}[TruePositives]}$$

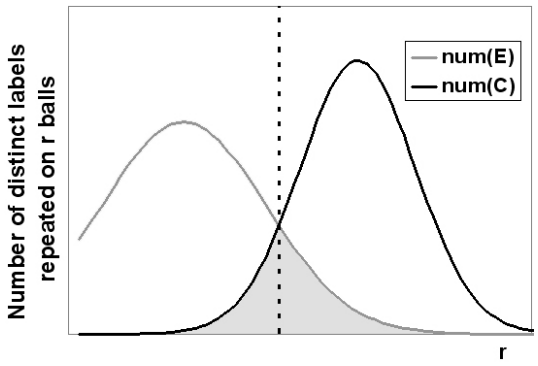

Figure 1: **Schematic illustration of the number of distinct labels in the C and E sets with repetition rate r. The "confusion region" is shaded.**

For example, consider the particular $num(C)$ and $num(E)$ learned (in the unsupervised setting) for the `Film` relation in our experiments. For the sample size $n = 134,912$ used in the experiments, expected number of true positives is 26,133 and expected precision is 70.2%, which is close to the actual observed true positives of 23,408 and precision of 67.7%. Were we to increase the sample size to 1,000,000, we would expect that true positives would increase to 47,609, and precision to 84.0%. Thus, URNS and the above equations enable an IE system to intelligently choose its sample size depending on precision and recall requirements and resource constraints, even in the absence of tagged training data.

2.3 Multiple Urns

We now generalize our model to encompass multiple urns. Information is often extracted using multiple, distinct mechanisms – for example, an IE system might employ several patterns for extracting city names, e.g. "cities including x" and "x and other towns." It is often the case that different patterns have different modes of failure, so extractions appearing across multiple patterns are generally more likely to be true than those appearing for a single pattern. We can model this situation by introducing multiple urns where each urn represents a different extraction mechanism.[1]

Thus, instead of n total extractions, we have a sample size n_m for each urn $m \in M$, with the extraction x appearing k_m times. Let $A(x, (k_1, \ldots, k_m), (n_1, \ldots, n_m))$ denote this event. Further, let $A_m(x, k, n)$ be the event that label x appears k times in n draws from urn m, and assuming that the draws from each urn are independent, we have:

Proposition 2

$$P(x \in C | A(x, (k_1, \ldots, k_m), (n_1, \ldots, n_m))) = \frac{\sum_{c_i \in C} \prod_{m \in M} P(A_m(c_i, k_m, n_m))}{\sum_{x \in C \cup E} \prod_{m \in M} P(A_m(x, k_m, n_m))}$$

With multiple urns, the distributions of labels among balls in the urns are represented by multi-sets $num_m(C)$ and

[1] We may lump several mechanisms into a single urn if they tend to behave similarly.

$num_m(E)$. Expressing the correlation between $num_m(x)$ and $num_{m'}(x)$ is an important modeling decision. Multiple urns are especially beneficial when the repetition rates for elements of C are more strongly correlated across different urns than they are for elements of E—that is, when $num_m(x)$ and $num_{m'}(x)$ tend to be closer to each other for $x \in C$ than for $x \in E$. Fortunately, this turns out to be the case in practice. Section 3 describes our method for modeling multi-urn correlation.

3 Implementation of the Urns Model

This section describes how we implement URNS for both UIE and supervised IE, and identifies the assumptions made in each case.

In order to compute probabilities for extractions, we need a method for estimating $num(C)$ and $num(E)$. For the purpose of estimating these sets from tagged or untagged data, we assume that $num(C)$ and $num(E)$ are Zipf distributed, meaning that if c_i is the ith most frequently repeated label in C, then $num(c_i)$ is proportional to i^{-z_C}. We can then characterize the $num(C)$ and $num(E)$ sets with five parameters: the set sizes $|C|$ and $|E|$, the shape parameters z_C and z_E, and the extraction precision p.

To model multiple urns, we consider different extraction precisions p_m for each urn, but make the simplifying assumption that the size and shape parameters are the same for all urns. As mentioned in Section 2, we expect repetition rate correlation across urns to be higher for elements of the C set than for the E set. We model this correlation as follows: first, elements of the C set are assumed to come from the same location on the Zipf curve for all urns, that is, their relative frequencies are perfectly correlated. Some elements of the E set are similar, and have the same relative frequency across urns – these are the *systematic* errors. However, the rest of the E set is made up of *non-systematic* errors, meaning that they appear for only one kind of extraction mechanism (for example, "Eastman Kodak" is extracted as an instance of `Film` only in phrases involving the word "film", and not in those involving the word "movie."). Formally, non-systematic errors are labels that are present in some urns and not in others. Each type of non-systematic error makes up some fraction of the E set, and these fractions are the parameters of our correlation model. Assuming this simple correlation model and identical size and shape parameters across urns is too restrictive in general— differences between extraction mechanisms are often more complex. However, our assumptions allow us to compute probabilities efficiently (as described below) and do not appear to hurt performance significantly in practice.

With this correlation model, if a label x is an element of C or a systematic error, it will be present in all urns. In terms of Proposition 2, the probability that a label x appears k_m times in n_m draws from m is:

$$P(A_m(x, k_m, n_m)) = \binom{n_m}{k_m} (f_m(x))^{k_m} (1 - f_m(x))^{n_m - k_m} \quad (3)$$

where $f_m(x)$ is the frequency of extraction x. That is,

$$\begin{aligned} f_m(c_i) &= p_m Q_C i^{-z_C} \text{ for } c_i \in C \\ f_m(e_i) &= (1 - p_m) Q_E i^{-z_E} \text{ for } e_i \in E \end{aligned}$$

In these expressions, i is the frequency rank of the extraction, assumed to be the same across all urns, and Q_C and Q_E are normalizing constants such that

$$\sum_{c_i \in C} Q_C i^{-z_C} = \sum_{e_i \in E} Q_E i^{-z_E} = 1$$

For a non-systematic error x which is not present in urn m, $P(A_m(x, k_m, n_m))$ is 1 if $k_m = 0$ and 0 otherwise. Substituting these expressions for $P(A_m(x, k_m, n_m))$ into Proposition 2 gives the final form of our URNS model.

3.1 Efficient Computation

A feature of our implementation is that it allows for efficient computation of probabilities. In general, computing the sum in Proposition 2 over the potentially large C and E sets would require significant computation for each extraction. However, given a fixed number of urns, with $num(C)$ and $num(E)$ Zipf distributed, an integral approximation to the sum in Proposition 2 (using a Poisson in place of the binomial in Equation 3) can be solved in closed form in terms of incomplete Gamma functions. This closed form expression can be evaluated quickly, and thus probabilities for extractions can be obtained efficiently. This solution leverages our assumptions that size and shape parameters are identical across urns, and that relative frequencies are perfectly correlated. Finding efficient techniques for computing probabilities under less stringent assumptions is an item of future work.

3.2 Parameter Estimation

In the event that a large sample of hand-tagged training examples is available for each target relation of interest, we can directly estimate each of the parameters of URNS. We use a population-based stochastic optimization technique to identify parameter settings that maximize the conditional log likelihood of the training data.[2] Once the parameters are set, the model yields a probability for each extraction, given the number of times k_m it appears in each urn and the number of draws n_m from each urn.

As argued in [Etzioni *et al.*, 2005], IE systems cannot rely on hand-tagged training examples if they are to scale to extracting information on arbitrary relations that are not specified in advance. Implementing URNS for UIE requires a solution to the challenging problem of estimating $num(C)$ and $num(E)$ using untagged data. Let U be the multi-set consisting of the number of times each unique label was extracted; $|U|$ is the number of unique labels encountered, and the sample size $n = \sum_{u \in U} u$.

In order to learn $num(C)$ and $num(E)$ from untagged data, we make the following assumptions:

- Because the number of different possible errors is nearly unbounded, we assume that the error set is very large.[3]

[2] Specifically, we use the Differential Evolution routine built into Mathematica 5.0.

[3] In our experiments, we set $|E| = 10^6$. A sensitivity analysis showed that changing $|E|$ by an order of magnitude, in either direction, resulted in only small changes to our results.

- We assume that both $num(C)$ and $num(E)$ are Zipf distributed where the z_E parameter is set to 1.
- In our experience with KNOWITALL, we found that while different extraction rules have differing precision, each rule's precision is stable across different relations [Etzioni *et al.*, 2005]. URNS takes this precision as an input. To demonstrate that URNS is not overly sensitive to this parameter, we chose a fixed value (0.9) and used it as the precision p_m for all urns in our experiments. [4]

We then use Expectation Maximization (EM) over U in order to arrive at appropriate values for $|C|$ and z_C (these two quantities uniquely determine $num(C)$ given our assumptions). Our EM algorithm proceeds as follows:

1. Initialize $|C|$ and z_C to starting values.
2. Repeat until convergence:
   (a) **E-step** Assign probabilities to each element of U using Proposition (1).
   (b) **M-step** Set $|C|$ and z_C from U using the probabilities assigned in the E-step (details below).

We obtain $|C|$ and z_C in the M-step by first estimating the rank-frequency distribution for labels from C in the untagged data. From the untagged data and the probabilities found in the E-step, we can obtain $E_C[k]$, the expected number of labels from C that were extracted k times. We then round these fractional expected counts into a discrete rank-frequency distribution with a number of elements equal to the expected total number of labels from C in the untagged data, $\sum_k E_C[k]$. We obtain z_C by fitting a Zipf curve to this rank-frequency distribution by linear regression on a log-log scale. Lastly, we set $|C| = \sum_k E_C[k] + unseen$, where we estimate the number of unseen labels of the C set using Good-Turing estimation ([Gale and Sampson, 1995]). Specifically, we choose $unseen$ such that the probability mass of unseen labels is equal to the expected fraction of the draws from C that extracted labels seen only once.

This unsupervised learning strategy proved effective for target relations of different sizes; for example, the number of elements of the `Country` relation with non-negligible extraction probability was about two orders of magnitude smaller than that of the `Film` and `City` relations.

Clearly, unsupervised learning relies on several strong assumptions, though our sensitivity analysis has shown that the model's performance is robust to some of them. In future work, we plan to perform a more comprehensive sensitivity analysis of the model and also investigate its performance in a semi-supervised setting.

4 Experimental Results

This section describes our experimental results under two settings: unsupervised and supervised. We begin by describing the two unsupervised methods used in previous work: the noisy-or model and PMI. We then compare URNS with these methods experimentally, and lastly compare URNS with several baseline methods in a supervised setting.

We evaluated our algorithms on extraction sets for the relations `City(x)`, `Film(x)`, `Country(x)`, and `MayorOf(x,y)`, taken from experiments performed in [Etzioni *et al.*, 2005]. The sample size n was 64,581 for `City`, 134,912 for `Film`, 51,313 for `Country` and 46,129 for `MayorOf`. The extraction patterns were partitioned into urns based on the name they employed for their target relation (e.g. "country" or "nation") and whether they were left-handed (e.g. "countries including x") or right-handed (e.g. "x and other countries"). Each combination of relation name and handedness was treated as a separate urn, resulting in four urns for each of `City(x)`, `Film(x)`, and `Country(x)`, and two urns for `MayorOf(x)`.[5] For each relation, we tagged a sample of 1000 extracted labels, using external knowledge bases (the Tipster Gazetteer for cities and the Internet Movie Database for films) and manually tagging those instances not found in a knowledge base. In the UIE experiments, we evaluate our algorithms on all 1000 examples, and in the supervised IE experiments we perform 10-fold cross validation.

4.1 UIE Experiments

We compare URNS against two other methods for unsupervised information extraction. First, in the *noisy-or* model used in previous work, an extraction appearing k times is assigned probability $1 - \prod_{m \in M}(1 - p_m)^k$, where p_m is the extraction precision for urn m. We describe the second method below.

Pointwise Mutual Information

Our previous work on KNOWITALL used Pointwise Mutual Information (PMI) to obtain probability estimates for extractions [Etzioni *et al.*, 2005]. Specifically, the PMI between an extraction and a set of automatically generated *discriminator phrases* (e.g., "movies such as x") is computed from Web search engine hit counts. These PMI scores are used as features in a Naive Bayes Classifier (NBC) to produce a probability estimate for the extraction. The NBC is trained using a set of automatically bootstrapped seed instances. The positive seed instances are taken to be those having the highest PMI with the discriminator phrases after the bootstrapping process; the negative seeds are taken from the positive seeds of other relations, as in other work (e.g., [Lin *et al.*, 2003]).

Although PMI was shown in [Etzioni *et al.*, 2005] to order extractions fairly well, it has two significant shortcomings. First, obtaining the hit counts needed to compute the PMI scores is expensive, as it requires a large number of queries to

[4] A sensitivity analysis showed that choosing a substantially higher (0.95) or lower (0.80) value for p_m still resulted in URNS outperforming the noisy-or model by at least a factor of 8 and PMI by at least a factor of 10 in the experiments described in Section 4.1.

[5] Draws from URNS are intended to represent independent extractions. Because the same sentence can be duplicated across multiple different Web documents, in these experiments we consider only each *unique* sentence containing an extraction to be a draw from URNS. In experiments with other possibilities, including counting the number of unique documents producing each extraction, or simply counting every occurrence of each extraction, we found that performance differences between the various approaches were negligible for our task.

web search engines. Second, the seeds produced by the bootstrapping process tend not to be representative of the overall distribution of extractions. This combined with the probability polarization introduced by the NBC tends to give inaccurate probability estimates.

Discussion of UIE Results

The results of our unsupervised experiments are shown in Figure 2. We plot deviation from the *ideal* log likelihood—defined as the maximum achievable log likelihood given our feature set.

Our experimental results demonstrate that URNS overcomes the weaknesses of PMI. First, URNS's probabilities are far more accurate than PMI's, achieving a log likelihood that is a factor of 20 closer to the ideal, on average (Figure 2). Second, URNS is substantially more efficient as shown in Table 1.

This efficiency gain requires some explanation. KNOWITALL relies on queries to Web search engines to identify Web pages containing potential extractions. The number of queries KNOWITALL can issue daily is limited, and querying over the Web is, by far, KNOWITALL's most expensive operation. Thus, number of search engine queries is our efficiency metric. Let d be the number of discriminator phrases used by the PMI method as explained in Section 4.1. The PMI method requires $O(d)$ search engine queries to compute the PMI of each extraction from search engine hit counts. In contrast, URNS computes probabilities *directly* from the set of extractions—requiring *no* additional queries, which cuts KNOWITALL's queries by factors ranging from 1.9 to 17.

As explained in Section 2, the noisy-or model ignores target set size and sample size, which leads it to assign probabilities that are far too high for the `Country` and `MayorOf` relations, where the average number of times each label is extracted is high (see bottom row of Table 1). This is further illustrated for the `Country` relation in Figure 3. The noisy-or model assigns appropriate probabilities for low sample sizes, because in this case the overall precision of extracted labels is in fact fairly high, as predicted by the noisy-or model. However, as sample size increases relative to the number of true countries, the overall precision of the extracted labels decreases—and the noisy-or estimate worsens. On the other hand, URNS avoids this problem by accounting for the interaction between target set size and sample size, adjusting its probability estimates as sample size increases. Given sufficient sample size, URNS performs close to the ideal log likelihood, improving slightly with more samples as the estimates obtained by the EM process become more accurate. Overall, URNS assigns far more accurate probabilities than the noisy-or model, and its log likelihood is a factor of 15 closer to the ideal, on average. The very large differences between URNS and both the noisy-or model and PMI suggest that, even if the performance of URNS degrades in other domains, it is quite likely to still outperform both PMI and the noisy-or model.

Our computation of log-likelihood contains a numerical detail that could potentially influence our results. To avoid the possibility of a likelihood of zero, we restrict the probabilities generated by URNS and the other methods to lie within the

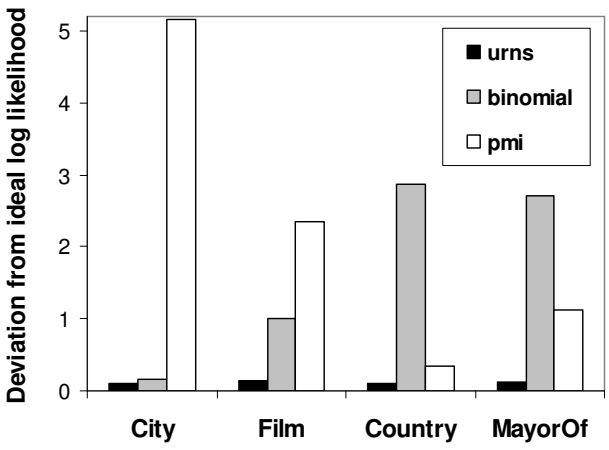

Figure 2: **Deviation of average log likelihood from the ideal for four relations (lower is better). On average,** URNS **outperforms noisy-or by a factor of 15, and PMI by a factor of 20.**

	City	*Film*	*MayorOf*	*Country*
Speedup	17.3x	9.5x	1.9x	3.1x
Average k	3.7	4.0	20.7	23.3

Table 1: **Improved Efficiency Due to** URNS**. The top row reports the number of search engine queries made by** KNOWITALL **using PMI divided by the number of queries for** KNOWITALL **using** URNS**. The bottom row shows that PMI's queries increase with k—the average number of distinct labels for each relation. Thus, speedup tends to vary inversely with the average number of times each label is drawn.**

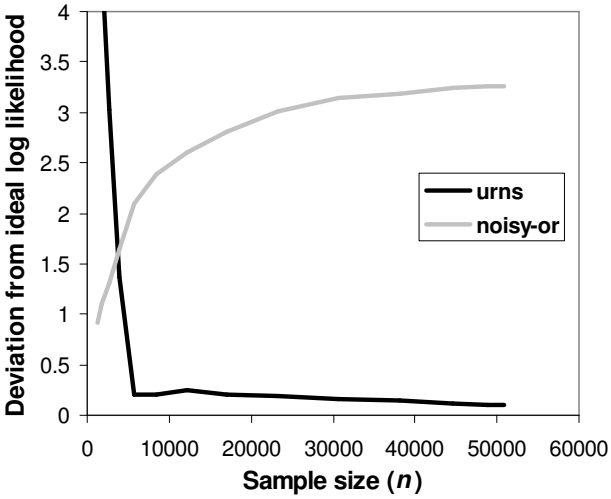

Figure 3: **Deviation of average log likelihood from the ideal as sample size varies for the `Country` relation (lower is better).** URNS **performs close to the ideal given sufficient sample size, whereas noisy-or becomes less accurate as sample size increases.**

range (0.00001, 0.99999). Widening this range tended to improve URNS's performance relative to the other methods, as this increases the penalty for erroneously assigning extreme probabilities—a problem more prevalent for PMI and noisy-or than for URNS. Even if we narrow the range by two digits of precision, to (0.001, 0.999), URNS still outperforms PMI by a factor of 15, and noisy-or by a factor of 13. Thus, we are comfortable that the differences observed are not an artifact of this design decision.

4.2 Supervised IE Experiments

We compare URNS with three supervised methods. All methods utilize the same feature set as URNS, namely the extraction counts k_m.

- **noisy-or** – Has one parameter per urn, making a set of M parameters $(h_1, \ldots, h_M)$, and assigns probability equal to
$$1 - \prod_{m \in M} (1 - h_m)^{k_m}.$$
- **logistic regression** – Has $M + 1$ parameters $(a, b_1, b_2, \ldots, b_M)$, and assigns probability equal to
$$\frac{1}{1 + e^{a + \sum_{m \in M} k_m b_m}}.$$
- **SVM** – Consists of an SVM classifier with a Gaussian kernel. To transform the output of the classifier into a probability, we use the probability estimation built-in to LIBSVM [Chang and Lin, 2001], which is based on logistic regression of the SVM decision values.

Parameters maximizing the conditional likelihood of the training data were found for the noisy-or and logistic regression models using Differential Evolution. In the SVM case, we performed grid search to find the kernel parameters giving the best likelihood performance for each training set – this grid search was required to get acceptable performance from the SVM on our task.

The results of our supervised learning experiments are shown in Table 2. URNS, because it is more expressive, is able to outperform the noisy-or and logistic regression models. In terms of deviation from the ideal log likelihood, we find that on average URNS outperforms the noisy-or model by 19%, logistic regression by 10%, but SVM by only 0.4%.

	City	*Film*	*Mayor*	*Country*	*Average*
noisy-or	0.0439	0.1256	0.0857	0.0795	0.0837
logistic regression	0.0466	0.0893	**0.0655**	0.1020	0.0759
SVM	0.0444	0.0865	0.0659	**0.0769**	0.0684
URNS	**0.0418**	**0.0764**	0.0721	0.0823	**0.0681**

Table 2: **Supervised IE experiments. Deviation from the ideal log likelihood for each method and each relation (lower is better). The overall performance differences are small, with URNS 19% closer to the ideal than noisy-or, on average, and 10% closer than logistic regression. The overall performance of SVM is close to that of URNS.**

5 Related Work

In contrast to the bulk of previous IE work, our focus is on unsupervised IE (UIE) where URNS substantially outperforms previous methods (Figure 2).

In addition to the noisy-or models we compare against in our experiments, the IE literature contains a variety of heuristics using repetition as an indication of the veracity of extracted information. For example, Riloff and Jones [Riloff and Jones, 1999] rank extractions by the number of distinct patterns generating them, plus a factor for the reliability of the patterns. Our work is intended to formalize these heuristic techniques, and unlike the noisy-or models, we explicitly model the distribution of the target and error sets (our $num(C)$ and $num(E)$), which is shown to be important for good performance in Section 4.1. The accuracy of the probability estimates produced by the heuristic and noisy-or methods is rarely evaluated explicitly in the IE literature, although most systems make implicit use of such estimates. For example, bootstrap-learning systems start with a set of seed instances of a given relation, which are used to identify extraction patterns for the relation; these patterns are in turn used to extract further instances (e.g. [Riloff and Jones, 1999; Lin *et al.*, 2003; Agichtein and Gravano, 2000]). As this process iterates, random extraction errors result in overly general extraction patterns, leading the system to extract further erroneous instances. The more accurate estimates of extraction probabilities produced by URNS would help prevent this "concept drift."

Skounakis and Craven [Skounakis and Craven, 2003] develop a probabilistic model for combining evidence from multiple extractions in a supervised setting. Their problem formulation differs from ours, as they classify each occurrence of an extraction, and then use a binomial model along with the false positive and true positive rates of the classifier to obtain the probability that at least one occurrence is a true positive. Similar to the above approaches, they do not explicitly account for sample size n, nor do they model the distribution of target and error extractions.

Culotta and McCallum [Culotta and McCallum, 2004] provide a model for assessing the confidence of extracted information using conditional random fields (CRFs). Their work focuses on assigning accurate confidence values to individual occurrences of an extracted field based on textual features. This is complementary to our focus on *combining* confidence estimates from multiple occurrences of the same extraction. In fact, each possible feature vector processed by the CRF in [Culotta and McCallum, 2004] can be thought of as a virtual urn m in our URNS. The confidence output of Culotta and McCallum's model could then be used to provide the precision p_m for the urn.

Our work is similar in spirit to BLOG, a language for specifying probability distributions over sets with unknown objects [Milch *et al.*, 2004]. As in our work, BLOG models treat observations as draws from a set of balls in an urn. Whereas BLOG is intended to be a general modeling framework for probabilistic first-order logic, our work is directed at modeling redundancy in IE. In contrast to [Milch *et al.*, 2004], we provide supervised and unsupervised learning methods

for our model and experiments demonstrating their efficacy in practice.

6 Conclusions and Future Work

This paper introduced a combinatorial URNS model to the problem of assessing the probability that an extraction is correct. The paper described supervised and unsupervised methods for estimating the parameters of the model from data, and reported on experiments showing that URNS massively outperforms previous methods in the unsupervised case, and is slightly better than baseline methods in the supervised case. Of course, additional experiments and a more comprehensive sensitivity analysis of URNS are necessary.

URNS is applicable to tasks other than IE. For example, PMI computed over search engine hit counts has been used to determine synonymy [Turney, 2001], and for question answering [Magnini *et al.*, 2002]. In the synonymy case, for example, the PMI between two terms is used as a measure of their synonymy; applying URNS to the same co-occurrence statistics should result in a more accurate probabilistic assessment of whether two terms are synonyms. Comparing URNS with PMI on these tasks is a topic for future work.

Acknowledgments

This research was supported in part by NSF grant IIS-0312988, DARPA contract NBCHD030010, ONR grant N00014-02-1-0324, and a gift from Google. Google generously allowed us to issue a large number of queries to their XML API to facilitate our experiments. We thank Anna Karlin, Marina Meila, and Dan Weld for helpful discussions, and Jeff Bigham for comments on previous drafts. Also, thanks to Alex Yates for suggesting we consider this problem.

References

[Agichtein and Gravano, 2000] E. Agichtein and L. Gravano. Snowball: Extracting relations from large plain-text collections. In *Proc. of the 5th ACM Intl. Conf. on Digital Libraries*, 2000.

[Chang and Lin, 2001] C. Chang and C. Lin. *LIBSVM: a library for support vector machines*, 2001.

[Culotta and McCallum, 2004] A. Culotta and A. McCallum. Confidence estimation for information extraction. In *HLT-NAACL*, 2004.

[Etzioni *et al.*, 2004] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A. Popescu, T. Shaked, S. Soderland, D. Weld, and A. Yates. Web-scale information extraction in system x: (preliminary results). In *WWW*, 2004.

[Etzioni *et al.*, 2005] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A. Popescu, T. Shaked, S. Soderland, D. Weld, and A. Yates. Unsupervised named-entity extraction from the web: An experimental study. In *To appear in AIJ*, 2005.

[Gale and Sampson, 1995] W. A. Gale and G. Sampson. Good-turing frequency estimation without tears. *Journal of Quantitative Linguistics*, 2(3):217–237, 1995.

[Lin *et al.*, 2003] W. Lin, R. Yangarber, and R. Grishman. Bootstrapped learning of semantic classes. In *ICML Workshop on The Continuum from Labeled to Unlabeled Data*, 2003.

[Magnini *et al.*, 2002] B. Magnini, M. Negri, R. Prevete, and H. Tanev. Is it the right answer? exploiting web redundancy for answer validation. In *ACL*, 2002.

[Milch *et al.*, 2004] B. Milch, B. Marthi, and S. Russell. BLOG: Relational modeling with unknown objects. In *ICML Workshop on Statistical Relational Learning and Its Connections to Other Fields*, 2004.

[Riloff and Jones, 1999] E. Riloff and R. Jones. Learning dictionaries for information extraction by multi-level bootstrapping. In *AAAI/IAAI*, 1999.

[Skounakis and Craven, 2003] M. Skounakis and M. Craven. Evidence combination in biomedical natural-language processing. In *BIOKDD*, 2003.

[Turney, 2001] P. D. Turney. Mining the Web for synonyms: PMI–IR versus LSA on TOEFL. *Lecture Notes in Computer Science*, 2167:491–502, 2001.