Entity Linking at the Tail: Sparse Signals, Unknown Entities, and Phrase Models

Yuzhe Jin, Emre Kıcıman, Kuansan Wang, Ricky Loynd
Microsoft
One Microsoft Way, Redmond, WA 98052, USA
{yuzjin,emrek,kuansan.wang,riloynd}@microsoft.com

ABSTRACT

Web search is seeing a paradigm shift from keyword based search to an entity-centric organization of web data. To support web search with this deeper level of understanding, a web-scale entity linking system must have 3 key properties: First, its feature extraction must be robust to the diversity of web documents and their varied writing styles and content structures. Second, it must maintain high-precision linking for "tail" (unpopular) entities that is robust to the existence of confounding entities outside of the knowledge base and entity profiles with minimal information. Finally, the system must represent large-scale knowledge bases with a scalable and powerful feature representation. We have built and deployed a web-scale unsupervised entity linking system for a commercial search engine that addresses these requirements by combining new developments in sparse signal recovery to identify the most discriminative features from noisy, free-text web documents; explicit modeling of out-of-knowledge-base entities to improve precision at the tail; and the development of a new phrase-unigram language model to efficiently capture high-order dependencies in lexical features. Using a knowledge base of 100M unique people from a popular social networking site, we present experimental results in the challenging domain of people-linking at the tail, where most entities have limited web presence. Our experimental results show that this system substantially improves on the precision-recall tradeoff over baseline methods, achieving precision over 95% with recall over 60%.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

Algorithm, experimentation

Keywords

Web-scale system, tail entity linking, sparsity, unknown entity, phrase language model

WSDM'14, February 24–28, 2014, New York, New York, USA.

http://dx.doi.org/10.1145/2556195.2556230.

1. INTRODUCTION

Web search is undergoing a paradigm shift from keyword-based search to an entity-centric organization of web data. A core component of this transformation is our improved understanding of the contents of documents — replacing the shallow summarization of documents sketched out by keywords with a deeper association between documents and the entities mentioned in them. Together with other technologies to allow better understanding of searchers' intents, this enables powerful new user experiences, from search results that directly show key facts about people, places and things, to improved refinement interfaces that allow searchers to quickly locate web documents that mention only the specific people, places or other things they are looking for.

Entity linking is the process of matching the references to people, places and other entities in a document to specific instances in an entity knowledge base. Entity linking is a critical step in establishing entity-level understanding of web documents and subsequent entity-centric search experiences. An entity linking system must have three desired properties to be successful at web-scale. First, it must work with heterogeneous web documents that differ in content structure, writing style, vocabulary and editorial quality, enabling a unified approach to analyzing noisy personal blogs as well as professional news articles. The second desired property is robustly linking "tail" entities that, individually, are infrequently seen in web documents, but in aggregate make up a significant bulk of entity references on the web. Linking these "tail" or unpopular entities is challenging because the entity descriptions at the "tail" are more likely to contain minimal information, in contrast with the comprehensive articles written about more popular entities in Wikipedia. Furthermore, because of the sheer number of existent tail entities, the corresponding tail knowledge bases include an order of magnitude or more entities than Wikipedia and other head knowledge bases. As a result of its size, a tail knowledge base is more likely to include entities that match confounds, including unknown homonymous entities that are not contained in even the largest of today's knowledge bases. Finally, and crucially, the system must effectively represent the information in the knowledge base and web documents using scalable representations for web-scale deployment.

We present an unsupervised entity linking system that provides a complete solution to all of these desired properties. Our primary contribution is the presentation and evaluation of the three key components of our system. First, we achieve a unified approach to analyzing diverse web documents through sparse signal recovery. We use a sparse signal recovery technique, adapted with a novel probability-based distance metric, to efficiently determine the most discriminative features in a web document — and ignoring noisy and irrelevant features — regardless of the document's

content structure or style. Secondly, we incorporate into our algorithms an explicit model of the entities outside of our knowledge base. This model improves the accuracy of "tail" entity linking by acting as a competing hypothesis that, in effect, dynamically adapts our decision criteria based on the statistical distributions of population features. Third, we capture higher-order dependencies among the lexical features in our web documents and knowledge base by replacing a token-based unigram language model with a phrase language model. This phrase language model maintains the same parameter complexity as a unigram language model, enabling efficient storage and access. While there is much existing research on entity recognition and linking, to our knowledge, no previous work has met all of these requirements for practically operating entity-linking in a web-scale environment with diverse document styles and structures and large-scale, tail-heavy knowledge bases. We have built and deployed our system for a commercial search engine, linking entity references in web documents with a large entity knowledge base.

We believe that entity linking at the tail is the core challenge that must be met for entity-linking systems to meet the requirements of next-generation web search and entity-centric applications, and previous entity linking works cannot provide web-scale solutions with desired requirements. In this paper, we use a knowledge base of 100 million unique people from a popular social networking site. We present experimental results in the challenging domain of people-linking at the tail, where most entities have many confounding and often unknown homonyms. We characterize this complexity, and demonstrate that our system substantially improves on the precision-recall tradeoff over baseline methods, achieving 96% precision with 62% recall. While it is not the focus of this paper, our quality reviews of our system's production deployment show that we achieve precision over 95% and recall over 80% on entity linking against Wikipedia entities, which consist of mainly "head" (popular) entities. We also test the performance of our parallelizable system, demonstrating that, with an average per-node processing capacity of 750 documents per second, it can be easily scaled to process tens of billions of web documents using existing distributed computing infrastructures.

The rest of the paper is organized as follows. Section 2 briefly reviews the necessary background on sparse signal recovery. Section 3 introduces our entity linking system in detail. Performance evaluation and algorithm complexity are analyzed in Section 4. We review related works in entity linking in Section 5. Section 6 concludes the paper.

2. SPARSE SIGNAL RECOVERY

Since the entity linking system exploits the recent developments in sparse signal recovery, we briefly review the model and techniques for finding sparse signals in this section as a technical preparation. Readers familiar with this area are free to skip this section without loss of understanding.

2.1 Model

Sparse signal recovery, also known as compressed sensing, has received much research attention [6, 11]. It studies how to represent a signal of interest (e.g., speech, image, video, text) by using only a few building elements (usually out of a dictionary with many building elements) and still capture the gist of the signal for an application. Mathematically, the goal is to approximate a signal $\mathbf{y}$ using a signal $\mathbf{x}$ under the constraint that $\mathbf{x}$ is *sparse*, which has only a few nonzero coefficients. Let $\mathcal{T}(\mathbf{x})$ be some transformation of $\mathbf{x}$ as the approximated signal. As a popular scenario, when $\mathcal{T}(\mathbf{x}) = A\mathbf{x}$ for a known matrix $A \in \mathbb{R}^{n\times m}$, the underlying signal model reduces to the linear model $\mathbf{y} = A\mathbf{x} + \mathbf{z}$, where $\mathbf{z}$ represents the fitting error. Let $\mathcal{D}(\mathcal{T}(\mathbf{x}), \mathbf{y})$ be a metric of the distance between the target signal $\mathbf{y}$ and the approximated signal $\mathcal{T}(\mathbf{x})$. For the linear model, $\mathcal{D}$ can be the Euclidean distance, i.e., $\mathcal{D}(A\mathbf{x}, \mathbf{y}) = \|A\mathbf{x} - \mathbf{y}\|_2^2$. Let $\mathcal{S}(\mathbf{x}) \geq 0$ measure the sparsity of the signal $\mathbf{x}$, i.e., a smaller $\mathcal{S}(\mathbf{x})$ indicates a sparser $\mathbf{x}$. Sparse signal recovery aims to find $\mathbf{x}$ according to the following generic optimization

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \mathcal{D}(\mathcal{T}(\mathbf{x}), \mathbf{y}) \text{ s.t. } \mathcal{S}(\mathbf{x}) \leq \epsilon \quad (1)$$

where $\epsilon > 0$ is a pre-defined threshold on the sparsity level, and $\mathbf{x}^*$ is the estimate of the sparse signal.

2.2 Algorithms

There are mainly two classes of algorithms for finding sparse signals: joint recovery algorithms and sequential selection algorithms. Joint recovery algorithms jointly estimate all nonzero entries in $\mathbf{x}$ by materializing (1) or its variants. In the case of a linear model, a well known example is the Lasso algorithm [36], which employs the Euclidean distance and $\mathcal{S}(\mathbf{x}) = \|\mathbf{x}\|_1$, and solves for

$$\hat{\mathbf{x}}_{\text{Lasso}} = \arg\min_{\mathbf{x}\in\mathbb{R}^m} \|A\mathbf{x} - \mathbf{y}\|_2^2 + \lambda\|\mathbf{x}\|_1 \quad (2)$$

where $\lambda \geq 0$ is a regularization parameter, which balances between fitting quality and sparsity of the solution vector.

Sequential selection algorithms do not materialize and solve (1) directly. Rather, they sequentially determine the nonzero entries of $\mathbf{x}$ one by one via a number of iterations. At each iteration, an algorithm of this class determines a nonzero entry whose addition into the signal $\mathbf{x}$ can most reduce the distance $\mathcal{D}$, and then updates the signal with the newly found nonzero element. The algorithm terminates using a stopping criterion, which guarantees the sparsity of $\mathbf{x}$ by limiting the number of iterations. A well known example for the linear signal model is the Matching Pursuit algorithm, which we briefly review as follows. As an initialization, let $\mathbf{y}^{(0)} = \mathbf{y}$ and $\mathcal{B}^{(0)} = \emptyset$. Consider the k-th iteration. There have been $(k-1)$ nonzero entries' locations recorded in $\mathcal{B}^{(k-1)}$ from the previous $(k-1)$ iterations. Now, Matching Pursuit finds among the remaining locations the column that best approximates the previous residual signal via the following optimization criterion

$$i^{(k)} = \arg\min_{i\in\{1,\ldots,m\}\setminus\mathcal{B}^{(k-1)}} \left[\min_{\alpha\in\mathbb{R}} \|\mathbf{y}^{(k-1)} - \alpha A_i\|_2^2\right] \quad (3)$$

where A_i denotes the i-th column vector of A. Note that the inner minimization finds the best fit to $\mathbf{y}^{(k-1)}$ using a given column A_i, whereas the outer minimization finds the best fit over all remaining columns. In effect, this step recovers a new nonzero entry that most reduces the matching error to the previous residual signal in Euclidean distance. Then, let $\mathcal{B}^{(k)} = \mathcal{B}^{(k-1)} \cup \{i^{(k)}\}$, and $\mathbf{y}^{(k)} = \mathbf{y}^{(k-1)} - \frac{A_{i^{(k)}} A_{i^{(k)}}^{\mathsf{T}}}{\|A_{i^{(k)}}\|_2^2} \cdot \mathbf{y}^{(k-1)}$. Matching Pursuit terminates as the number of iterations reaches a pre-defined threshold, or the approximation error is small enough. The final index set, denoted by $\mathcal{B}$, contains the indices corresponding to the locations of the nonzero entries. The values of the nonzero entries can be readily found by solving a least squares problem

$$\hat{\mathbf{x}}_{\mathcal{B}} = (A_{\mathcal{B}}^{\mathsf{T}} A_{\mathcal{B}})^{-1} A_{\mathcal{B}}^{\mathsf{T}} \mathbf{y}.$$

The full vector $\hat{\mathbf{x}}_{\text{MP}}$ can be determined by using the values of $\hat{\mathbf{x}}_{\mathcal{B}}$ at corresponding indices in $\mathcal{B}$ and setting zero to all other locations.

We will evaluate entity linking algorithm instances from both classes of algorithms in our experiments.

3. THE ENTITY LINKING SYSTEM

We present our web-scale entity linking system in this section. To assist the technical development, we present the following definitions and notations. Let $\mathcal{E}$ denote an entity knowledge base. For each entity $e \in \mathcal{E}$, the information is represented in the form of attribute-value pairs. Let a_e denote an attribute of e, and v_e denote the corresponding value of attribute a_e. We work with text-based (lexical) values, although, in general, the values can take on other forms such as image, audio, or video. Using this representation, the profile of an entity e in the knowledge base can be characterized by $\{(a_e^{(k)}, v_e^{(k)})\}_{k=1}^{r}$, where r is the number of attribute-value pairs available for entity e. For a given web document D, the goal of entity linking is to determine an entity $e \in \mathcal{E}$ so that D mentions e in its content. If no such entity exists, we claim that D mentions an entity that is outside of $\mathcal{E}$, or simply an *unknown* entity.

3.1 Importance of Sparsity in Entity Linking

For web-scale entity linking, the system is required to be able to process heterogeneous web documents that significantly differ in edit quality, writing style, content structure, among other characteristics. It is inevitable that a web document usually contains information that is irrelevant to the entity of interest. The key realization that motivates the design of our system is that the most discriminative features in a web document for entity linking are usually only a few, and thus highly sparse compared with all the information presented in the document. As an example, below is an excerpt from a person Ryan Smith's home page.

"*Ryan Smith is a versatile, classically trained pianist from South Carolina. Ryan has performed solo and collaborative recitals across the Southeast. He has recently appeared at the Columbia Museum of Art, Piccolo Spoleto Festival, [Edit: other venues he played at.]*

Ryan is a member of the Columbia Music Teachers Association. In 2007 Ryan was designated a Nationally Certified Teacher of Music (NCTM) by the National Music Teachers' Association. [Edit: continue to discuss his students.]

In addition to performing, [Edit: his other interests.]"

Among the many lexical terms presented above, the salient lexical features leading to a confident entity linking can be "pianist", "South Carolina", and "Columbia Music Teachers Association", together with the person's name. The rest of the lexical components include (i) language components which are merely related to the entity, e.g., "a member of", "in addition to performing"; (ii) features that are less critical (and more ambiguous) once the most salient features are identified, e.g., "versatile", "Southeast", and the places loosely related to him due to his appearance at, e.g., "Columbia Museum of Art".

Note that the excerpt above (from a person's home page) can be regarded as a document with average editorial quality. The web documents generated by social media, such as social network pages, blogs, forums, are usually of lower editorial quality and more arbitrary content organization. Conceivably, in such documents, the most discriminative information for entity linking is even sparser due to the usage of free style writing and the nature of discussion.

In our system, we fully exploit the sparsity nature in entity linking by identifying a small amount of features with the most discriminative power. We designed a Bayesian sequential selection approach to select the most discriminative features one by one. The selection process is terminated as soon as a decision on entity linking can be confidently reached. This ensures that only a small set of features is selected, therefore guaranteeing sparsity. An accompanying novel aspect that distinguishes our system from virtually all other sparsity signal recovery systems is that we adopt the posterior probability as the distance metric, which is in sharp contrast to the dominant adoption of the Euclidean distance. The Bayesian framework naturally leads to the posterior probability as the distance metric. Our experiments show that much higher precision in entity linking can be achieved, which is a highly desirable property for novel entity search applications.

Now, let us start unveiling the system in mathematical details. Some details are intentionally left unexplained at present, and they will become clear as we progress into later sections.

Let $\mathcal{F}_e$ denote the set of lexical features for entity e. Let $\mathsf{P}(f|e)$ be the probability of seeing feature f in a document that mentions entity e. A typical approach for estimating $\mathsf{P}(f|e)$ is to apply the maximum likelihood estimate, which translates into the normalized frequency of the occurrence of f in $\mathcal{F}_e$. A smoothing method can be applied to improve the estimation [8]. Let $\mathsf{P}(e)$ denote the prior probability of the entity e.

For entity linking, it is often helpful to apply proper heuristics to reduce the search space to a set of entities that are most likely to contain the underlying entity. Let us work with a confined entity set, denoted by $\mathcal{E}_c$ with $\mathcal{E}_c \subseteq \mathcal{E}$, which is usually a much smaller subset of the entity knowledge base $\mathcal{E}$. For example, we might select an $\mathcal{E}_c$ such that each entity in the subset has a name or nickname that is at least a partial match with some name in the document.

To exploit the sparsity in entity linking, we develop the Posterior Probability Pursuit (PPP) algorithm, which is an instance of the sequential selection algorithms for sparse signal recovery. Given a document D and its lexical feature set $\mathcal{G}$, PPP iteratively selects the most discriminative features, and it terminates with a confident entity linking decision using as few iterations as possible, resulting in a sparse set of features being selected. The algorithm is described in Algorithm 1.

---

input : $\mathcal{E}_c, \mathcal{G}, \mathsf{P}(f|e), \mathsf{P}(e), t \in (0,1); K \in \mathbb{N}$.
output: e^*
begin
  Initialization. $\mathcal{F}_0 = \emptyset, k = 1$;
  while $\mathcal{G} \backslash \mathcal{F}_k \neq \emptyset$ **do**
    Feature selection. Let

$$f_k = \arg \min_{g \in \mathcal{G} \backslash \mathcal{F}_{k-1}} \left[\min_{e \in \mathcal{E}_c} -\log \mathsf{P}(e|\mathcal{F}_{k-1} \cup \{g\}) \right] \quad (\star)$$

    Set $\mathcal{F}_k = \mathcal{F}_{k-1} \cup \{f_k\}$;
    if $\max_{e \in \mathcal{E}_c} \mathsf{P}(e|\mathcal{F}_k) > t$ **or** $k = K$ **then**
      break;
    end
    $k \leftarrow k + 1$;
  end
  $e^* = \arg\min_e -\log \mathsf{P}(e|\mathcal{F}_k)$. $(\triangle)$
end

Algorithm 1: The Posterior Probability Pursuit algorithm.

The essence of PPP is to find the best features which can minimize the distance between the web document and *some* entity profile in the entity knowledge base. To this end, PPP adopts the sequential selection principle to augment the best feature set $\mathcal{F}$ over a number of iterations. At the beginning of each iteration, the most discriminative yet unselected lexical feature from $\mathcal{G}$ is found to maximally decrease the distance metric, which measures the agreement between an entity and the features selected from the web document thus far. Indeed, by letting $\mathcal{D}(D, e) = -\log \mathsf{P}(e|\mathcal{F})$ and $\mathcal{S}(\mathcal{F}) = |\mathcal{F}|$, this step of PPP resembles (3) in the Matching Pur-

suit algorithm. By enforcing an early termination, PPP encourages only a small amount of highly discriminative features to be utilized for resolving entities, which echoes the sparsity nature underlying the problem. At termination, the entity with the lowest distance to the web document is determined as the entity linking outcome. In particular, the distance metric, to be exact, is the negative log posterior probability, i.e., $-\log \mathsf{P}(e|\mathcal{F})$. This is a novel application of sparse signal recovery, which is different from the widely used Euclidean distance metric in this area but is shown later to be much more suitable for entity linking.

The PPP algorithm can be also interpreted as combining the Naive Bayes classifier with the sparse sequential selection principle. From a Bayesian viewpoint, PPP only uses a subset of features with the most discriminative power, which are determined *dynamically* for a given web document. This is different from the Naive Bayes classifier which bases its decision on all features.

3.2 Competing Hypothesis for Tail Entities

Web-scale entity knowledge bases significantly extend into the entities with limited web presence and popularity, which are colloquially termed as *tail* entities. As examples, many people entities on social networks such as LinkedIn and Facebook have limited web presence outside their profiles on the social network. Resolving tail entities are much more challenging due to the following reasons. First, the profile of a tail entity often contains much less information, i.e., smaller number of attribute-value pairs, and/or shorter value descriptions. This can be readily seen by comparing a random person's LinkedIn profile and, say, NBA player Michael Jordan's Wikipedia page. Another reason is the incomplete coverage of entity knowledge bases. The majority of tail entities are not been included in even the largest of today's entity knowledge bases. Ironically, however, while these knowledge bases are not large enough to contain the majority of tail entities, they are large enough to include some of the confounding, homonymous entities. The result is that, in practice, we discover that the out-of-knowledge-base entities indeed create a significant challenge for a robust entity linking system. If these unknown entities are not addressed in the model, a system will mistakenly assign an entity with the same name in the knowledge base to the web document, hurting the precision of the system and entity-centric experiences built upon it.

In our system, we adopt a simple and effective model for the out-of-knowledge-base entities, similar to the approach advocated in [18]. The essence of the model is to construct a profile of a special entity, which is designed to characterize all the entities outside the entity knowledge base. Thus, this artificially injected entity serves as a competing hypothesis that represents the tail entities outside the knowledge base and therefore cannot be linked yet.

To be concrete, let us formally define an *unknown entity*, denoted by e_u, to represent the out-of-knowledge-base entities. For e_u, we define

$$\mathsf{P}(f|e_u) = |\{e : f \in \mathcal{F}_e\}|/|\mathcal{E}|$$

for $f \in \cup_{e\in\mathcal{E}}\mathcal{F}_e$, which means that the probability of encountering feature f in a document describing an out-of-knowledge-base entity is approximated by the probability of seeing f in the feature set of a random entity from the knowledge base. Essentially, we assume that the knowledge base is representative of the greater, unknown population.

To employ the unknown entity in the PPP algorithm, we simply treat it as a regular entity and factor it into the equations. Specifically, in the computation of ($\star$) in Algorithm 1, we use the Bayes rule as well as the conditional independence assumption to obtain

$$\mathsf{P}(e|\mathcal{F}) = \frac{\mathsf{P}(\mathcal{F}|e)\mathsf{P}(e)}{\sum_{e'\in\mathcal{E}_c\cup\{e_u\}}\mathsf{P}(\mathcal{F}|e')p(e')}$$

where

$$\mathsf{P}(\mathcal{F}|e) = \prod_{f\in\mathcal{F}} p(f|e),\ \ e \in \mathcal{E}_c \cup \{e_u\}$$

and the prior probabilities of the entities are computed as follows

$$\mathsf{P}(e) = \frac{1}{M + |\mathcal{E}_c|},\ \ e \in \mathcal{E}_c$$
$$\mathsf{P}(e_u) = \frac{M}{M + |\mathcal{E}_c|}.$$

Note that the free parameter $M \in \mathbb{N}$ controls the prior probability of the unknown entity as well as the known entities.

Meanwhile, the ($\triangle$) of the PPP algorithm should include the unknown entity as one possible outcome, as shown below.

Revised ($\triangle$) in PPP: $e^* = \arg\min_{e\in\mathcal{E}_c\cup\{e_u\}} -\log \mathsf{P}(e|\mathcal{F}_k)$.

Similarly, this unknown entity model may also be integrated with Naive Bayes and other probabilistic methods.

3.3 Phrase Language Model

The last component of our system is the extraction of features f from web documents and entities in our knowledge base for use in the PPP algorithm. Our system utilizes lexical features to build models for entities. All the (text-based) values (i.e., v_e's) in the knowledge base serve as the source of lexical features. Traditionally, lexical features are exploited using a bag of words model or a unigram language model [7]. These models have proven effective in many information retrieval tasks [10, 33, 21, 3]. A major drawback of the unigram LM is its inability to model high-order dependencies among words. Higher order language models have been pursued to address this issue. However, such LMs require a much larger parameter space and hence are not suitable for web-scale entity linking with potential storage and processing feasibility constraints, despite the modest performance gains [7].

In our system, we employ a novel phrase unigram language model $\mathsf{P}(f|e)$, in which a lexical feature f can be either a word or a phrase, where the latter can naturally preserve the dependencies among words [38, 26]. However, phrase boundaries are usually hidden, unlike those between words. Building a phrase unigram language model requires that phrases be systematically harvested from the text. Two phrase discovery approaches are explored.

The first approach relies on the observation that some attribute a_e usually takes a phrase as its value v_e. For example, in the LinkedIn knowledge base, attributes such as "geographic area", "professional association", and "job title" typically have values which are by themselves phrases, e.g., "South Carolina", "Columbia Music Teachers Association", and "classical pianist", respectively. Using this heuristic, the value v_e in its entirety can be extracted as a phrase.

The second approach algorithmically discovers phrases from free style texts, such as the main body of a Wikipedia article, a detailed review of a restaurant on Yelp, a summary of some person's professional experiences and qualifications on LinkedIn, and the biography of a movie director on IMDb. The phrases are extracted through application of a statistical language model, which models phrase boundaries as partially observable random variables. Punc-

tuation marks provide some phrase boundaries, while other phrase boundaries are unobserved. A phrase segmentation is a set of contiguous phrases spanning a particular range of text. The procedure applies successive EM iterations. The first step of an iteration is to define new phrases. The second step is to re-segment the training corpus. The third step is to rebuild the model. The iterations are terminated when over-fitting is observed on a held-out data set.

In our system, we use both approaches for harvesting phrases from text. The advantages of a phrase unigram language model is that it effectively captures useful dependencies among words in a model that can be efficiently stored and accessed.

3.4 Discussion on Algorithmic Detail

3.4.1 Parameter Selection

In PPP, the parameters t and K directly control the termination criteria. Obviously, t takes values between $[0, 1]$. When t is set closer to 1, the stopping criterion is more stringent. The effect is to require more lexical features for entity linking. Thus, it reduces the rate of falsely claiming a matching entity, at the expense of missing out entities that are actually mentioned in the web document. It is usually the requirement of the application that determines the operational performance tradeoff. The parameter K can be a pre-defined threshold indicating the maximum number of iterations. Our experiences show that $K = 20$ usually suffices. The parameter M determines the prior probability of all entities, and it can be interpreted as an estimate of the total amount of out-of-knowledge-base entities, which is itself difficult to estimate [15, 23].

3.4.2 Adaptive Feature Selection

Note that PPP dynamically determines the most salient features from a web document. This is different from the existing feature selection techniques [17, 32] in which a set of feature dimensions with fixed weights are learned using the training data and then used for classification in all test cases. Essentially, our system adaptively selects the salient features on a per-test-case basis. Beside the benefit that no labeled data is required to perform the proposed techniques, the mechanism of dynamic feature selection represents an adaptation to leverage the sparse intersection between lexical features of the web document and the entities caused by various reasons including frequent updates to the knowledge base as well as different writing-styles, which are circumstances that systems with fixed weighting schemes cannot handle robustly [5].

4. EVALUATION

In our regular quality reviews of our commercial deployment, we find that our system's performance on "head" (Wikipedia) entities achieves a precision over 95% with recall over 80%. In this paper, we focus our performance evaluation on the more challenging, and we believe as yet unaddressed, task of entity linking for "tail" entities.

For this purpose, we use an entity knowledge base of 100M people profiles, gathered from a social networking site; and evaluate the quality of our entity linking on a selection of diverse web pages mentioning people. Note that our experiments focus on the entity linking task, and we assume that entity recognition (finding entity mentions, for example) is handled separately.

4.1 Entity Knowledge Base

The knowledge base is built upon the information from a popular online social network. It contains the profiles of about 100 million unique people entities. Most of the people entities are tail entities, which means that their web presences are limited. Each profile employs the attribute-value pair representation to store various aspects of information about a person. For the purpose of illustration, Table 1 presents an example profile of a people entity.

Attribute	Value
Name	Ryan Smith
Location	Columbia, South Carolina
University	University of South Carolina
Degree	Doctor of Musical Arts
Work	South Carolina State University
Expertise	Taubman approach and physical aspects ...

Table 1: An example profile in the people knowledge base.

4.2 Documents with Entity Labels

Building an independent set of web documents with entity labels requires one to determine whether some or none of the people entities in the knowledge base are mentioned in a document. Comprehensively labeling entities is particularly important for measuring the recall for linking tail entities. Overall, we obtained labels for 555 web documents with entity labels. Among them, 185 web documents are found to have matching people entities in the entity knowledge base. For the rest of the 370 web documents, our exhaustive search found no matching entities within the knowledge base, and hence they are labeled as out-of-knowledge-base, unknown entities. Specifically, the large portion (about 67%) of the documents with an unknown entity label clearly signals the severity of out-of-knowledge-base tail entities, even for a large KB with over 100M entities.

The detailed procedure is described as follows. We randomly selected 50 names from our people knowledge base. For each selected name, we collect a set of web pages that may mention some people entity with this name as follows. By using each name as a query, we first obtain the top 20 results in the SERP via a commercial search engine. Then, the following types of web pages are manually removed: (i) Directory page. A web document of this kind usually mentions many different people entities with the same or similar names. Common domains for such web pages are, for example, `pipl.com` and `www.spokeo.com`. (ii) Password-protected page, which requires log-in to access the actual information. (iii) Web pages directly from the social network which are used to build our entity knowledge base. Using this procedure, we obtain 555 web documents with potential people entity mentions. These web documents encompass a large variety of writing styles and editorial qualities. Only about 10% of the documents are news articles and online knowledge base articles which have decent editorial quality. The rest of the documents include (i) home pages of people with different professions; (ii) home pages of different small businesses; (iii) articles/conversations/reviews from blogs (e.g., WordPress), technical forums, YouTube, academic research listings, social networks, etc. For each web document D, the people entities with the name which is used as the query to retrieve D comprise the confined entity set $\mathcal{E}_c$. The size of $\mathcal{E}_c$ ranges from only one to about $2,400$, depending on the number of namesakes in the people knowledge base.

To obtain the ground-truth entity labels, we manually examined each web document and applied exhaustive search to determine whether a matching entity exists in the confined entity set $\mathcal{E}_c$. The detailed procedure is described as follows. First, by reading through a web document, we determine whether it is actually about any person with the specific name. There are occasions that the first name

and the last name happen to co-occur in a web document without actually referring to any person with the specific name. Once we recognize such a situation, we determine that the web document refers to no entity in $\mathcal{E}_c$, or equivalently an unknown entity for our purpose. When the web document mentions some person with the specific name, we try to understand the information about the person by inspecting the content in the document. Then, we look for entities that matched the features on the web document from the people knowledge base. We declare a matching entity if we can find one from the inspection. Otherwise, we continue looking for the entity from the social networking site from which the knowledge base is built. Eventually, if a matching people entity still cannot be found, we declare that the web document mentions an unknown entity.

4.3 Baseline Methods

Our proposed approach for entity linking utilizes the lexical features in the documents as well as the knowledge base. Meanwhile, the people entity knowledge base in our experiment contains no direct relationships or interconnections among the people entities. Thus, it is impractical to compare with previous methods such as discussed in [9, 30, 29] due to their dependence on the structures and the interlinks in Wikipedia pages. Since lexical features are arguably the most important building block for text information processing tasks, we employ three baseline methods for entity linking that are all based on lexical features. Note that the confined entity set $\mathcal{E}_c$ is also employed by the baseline methods.

Cosine similarity with tf-idf weights (tf-idf). The vector space representation of the web document and the people entity profiles are formed using the tf-idf weights. The cosine similarity is calculated to determine the best matching people entity. To determine the unknown people entity, we preset a threshold on the cosine similarity. If the people entity with the highest cosine similarity does not exceed the threshold, an unknown entity will be claimed. This baseline method represents the lexical-features-based algorithmic component applied across a series of entity disambiguation techniques [2, 5, 9, 20].

Naive Bayes (NB). A Naive Bayes classifier with the additional unknown entity is constructed. The entity with the maximum posterior probability is linked to the people mention in the web document. The pre-defined parameter is the total number of unknown population M, which equivalently specifies the prior probabilities of the entities. This baseline method essentially reproduces the state-of-the-art technique developed in [18] without the name variation model, which has to be learned with information beyond lexical features.

Lasso based entity linking (Lasso). This method also exploits the sparsity in entity linking using sparse signal recovery. It is an instance of the joint recovery algorithms. It differs from PPP in three important aspects. First, it employs the Lasso algorithm to jointly select the discriminative features. Second, it adopts the Euclidean distance for measuring the closeness between the web document and entity profiles. Finally, it has to work with phrases by algorithmic nature: a phrase corresponds to a dictionary item, i.e., a columns of the matrix A. There are two parameters, one being λ in (2), another being a threshold on the Euclidean distance for determining unknown entity. Note that the Lasso algorithm has been adopted to build the state-of-the-art web information processing applications [22].

4.4 Definitions of Precision and Recall

We define the performance metrics. Since an entity search application can only utilize the web documents with *known* entities, the performance metrics should emphasize the importance of the correct discovery of known entities over the correct claim of unknown, out-of-knowledge-base entities. By slightly abusing the terminologies in traditional information retrieval, we define the *precision* and *recall* as follows.

First, we define the following auxiliary quantities:

n_1: the number of documents for which the algorithm correctly determines the matching people entity in the knowledge base.

n_2: the number of documents which the algorithm determines as mentioning some people entity in the knowledge base.

n_3: the number of documents which have a matching people entity in the knowledge base, i.e., $n_3 = 185$ in this experiment.

Then, we define the performance metrics as

$$\textit{precision} \triangleq \frac{n_1}{n_2}, \quad \textit{recall} \triangleq \frac{n_1}{n_3}.$$

To understand the metrics, consider the following examples. Suppose an additional web document mentions no entity in the knowledge base but the algorithm incorrectly determines a matching entity in the knowledge base. Then, it decreases the precision due to an increased n_2. As another example, suppose the algorithm correctly identifies an additional web document as mentioning an unknown entity. Then, n_1, n_2 stay unchanged so that both precision and recall are unaffected. This behavior reflects the downstream applications' need for web document with known entities, since their correct identifications can provide actual useful information to enrich the entity search experience.

Since different entity centric applications require different precision and recall in entity linking, it is incomplete and improper to use a single best performance metric (such as F_1) for evaluation. Instead, we compare the precision-recall tradeoffs offered by an algorithm. This methodology provides a comprehensive view of all the operational points of an algorithm.

4.5 Results

We first pivot the performance comparison on different perspectives to understand how the entity linking system requirements are met. The overall result is presented together in Fig.5 at the end of the section.

4.5.1 Usefulness of Sparsity

We examine the usefulness of exploiting the sparsity nature of entity linking. In this section, we employ the bag of phrases model for all algorithms. First, we compare two baseline methods, tf-idf and Lasso. Their distance metrics are both based on the cosine similarity defined in the sense of Euclidean distance. The difference is that while the tf-idf approach uses all the lexical features, the Lasso based approach only uses a sparse subset of lexical features. Fig.1 shows the results. We see that the performance tradeoff of Lasso is relatively insensitive to the parameter α (or, λ). Obviously, at the same recall, the Lasso based approach achieves much higher precision than the tf-idf approach. The entire range of precision score for the tf-idf approach stays rather low. By enforcing sparsity in the selection of lexical features, Lasso achieves much more favorable performance.

Next, we compare PPP to the NB classifier. They are both developed under the Bayesian framework and equipped with posterior probability as the distance metric. The difference is that PPP works with only a few features whereas NB utilizes all of them. The results are presented in Table 2.

Note that varying the parameters M and t in the PPP algorithm delivers a tradeoff between precision (P) and recall (R). For a fixed M, NB can be viewed as a special case of PPP by using *all* the

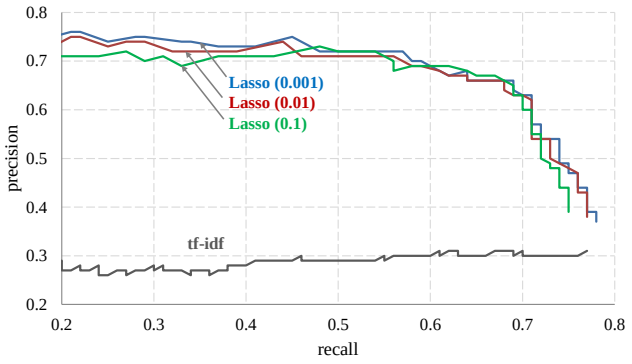

Figure 1: Sparsity helps: tf-idf (non-sparse) vs. Lasso (sparse). For tf-idf, different points on the curve are obtained using different thresholds (from 0 to 1) on the cosine similarity for determining unknown entity. For Lasso, the value in the parenthesis indicates the parameter α (as used in the regularization parameter $\lambda = \alpha\|A^{\intercal}\mathbf{y}\|_{\infty}$ [13]). Different points on a Lasso curve correspond to different thresholds (from 0 to 1) for determining unknown entity.

		PPP						NB
M	t_0	1	3	5	7	9	11	
10^8	P	0.62	0.76	0.82	**0.87**	**0.87**	**0.87**	0.83
	R	0.79	0.78	0.77	**0.76**	**0.73**	**0.73**	0.70
10^{10}	P	0.77	0.86	0.93	0.94	0.94	**0.96**	0.96
	R	0.66	0.65	0.65	0.63	0.62	**0.62**	0.58
10^{12}	P	0.87	0.95	0.97	0.97	**0.99**	**0.99**	0.99
	R	0.57	0.56	0.54	0.52	**0.52**	**0.52**	0.48
10^{14}	P	0.96	0.98	0.98	**1**	**1**	**1**	1
	R	0.47	0.44	0.44	**0.43**	**0.43**	**0.42**	0.38

Table 2: Sparsity helps: NB (non-sparse) vs. PPP (sparse). For PPP, $t = 1 - 10^{-t_0}$.

lexical features, which is algorithmically achieved by setting $t = 1$ (or, $t_0 = \infty$) and $K = \infty$ in PPP. From the P-R scores in boldface, PPP improves the recall at precisions no lower than its NB counterpart when t is close to, but surely less than, one. Clearly, properly exploiting the sparsity nature enables performance improvement.

4.5.2 *Effect of Distance Metric and Unknown Entity*

For PPP (and NB), the posterior probability distance metric naturally incorporates the probabilistic unknown entity model as an entity class. For Lasso (and tf-idf), the Euclidean distance metric directly works with a threshold on the distance metric to determine unknown entities. The distance metric and the unknown entity model require mutual compatibility and should not be paired with arbitrary substitutes. Therefore, we examine their joint effect by studying the performance comparison between PPP and Lasso using the phrase language model. Fig.2 shows the performance.

Based on Fig.2, the precision-recall tradeoff of PPP over different M is relatively insensitive to the parameter t. Obviously, the two sets of curves unveil the sharp difference between the performance tradeoffs of PPP and Lasso, respectively. PPP enjoys a much better precision-recall tradeoff, in which the precision can be greatly improved over the Lasso based approach at a given recall.

Next, we analyze the result from a different perspective. Note that, in effect, a mechanism for modeling unknown entity controls the degree of aggression in outputting unknown labels. Let us compare the tradeoff between the entity linking accuracy on the web documents mentioning a known people entity and the accuracy on the web documents mentioning an unknown people entity for PPP and Lasso, respectively, in Fig.3.

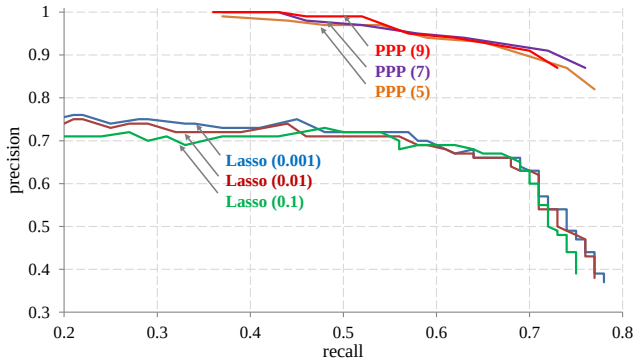

Figure 2: PPP outperforms Lasso. For PPP, each curve corresponds to $t = 1 - 10^{-t_0}$ where t_0 is provided in the parenthesis; different points on a curve represent different M (from 10^8 to 10^{15}), where higher precision corresponds to greater M.

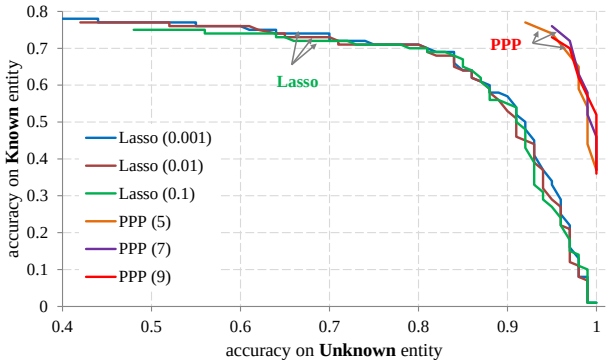

Figure 3: How well an algorithm does in linking *known* entities vs. determining *unknown* entities?

Note that within each set of curves (PPP and Lasso, respectively) in Fig.3, the performance tradeoffs are similar. At a given linking accuracy on the web documents mentioning known people entities, PPP offers much better linking accuracy on the web documents mentioning unknown people entities. From an algorithmic viewpoint, PPP employs a principled unknown entity model which is probabilistic in nature. In contrast, Lasso uses a simple threshold on the similarity measure, which is mainly an engineering solution to determining unknown entities. Since there is a nontrivial probability of encountering unknown entities (in this experiment, about 67%) as in web-scale applications with many tail entities, the unknown entity model introduced in Section 3.2 (adopted by PPP) delivers a much greater capability for determining whether a people entity is outside of the knowledge base.

Finally, we completely disable the unknown entity model by setting $M = 0$ in PPP. The threshold t on the posterior probability is used to determine unknown entity. We found that the precision stays at a very low level, $27\% \pm 1\%$, at a recall of $84\% \pm 1\%$, for $t \in [0.5, 1 - 10^{-12}]$. (Similar results are also observed for NB with $M = 0$.) Clearly, the unknown entity model is critical for achieving high precision in entity linking.

4.5.3 *Utility of Phrase Language Model*

In this section, we examine the utility of the phrase unigram language model for entity linking. For ease of illustration, only PPP is considered. We plot the best precision-recall tradeoffs by using phrases (labeled as "PPP (phrase)") and using the traditional word unigram (labeled as "PPP (word)"), respectively, in Fig.4.

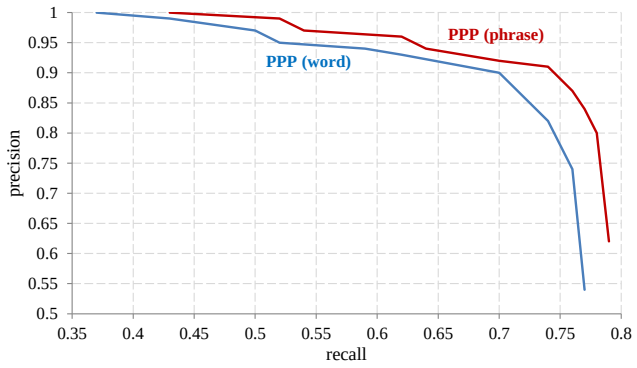

Figure 4: Phrase LM improves performance.

From Fig.4, we can clearly see a substantial improvement by using the phrase unigram language model in PPP. Especially, by including the phrases in the model, PPP can achieve much higher recall at the high precision region.

4.5.4 *Overall Result*

Finally, the best precision-recall tradeoff of each algorithm is shown in Fig.5. Overall, the sparsity-exploiting PPP algorithm with

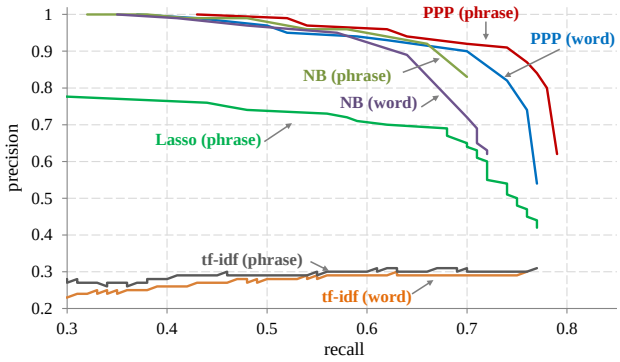

Figure 5: Overall performance comparison. Our system (PPP with Phrase LM) achieves the best performance tradeoff.

the phrase unigram language model, which is employed in our entity linking system, yields the best precision-recall tradeoff among all approaches in comparison.

4.6 Performance at Large-Scale

In this section, we analyze the time complexity of the system and demonstrate its feasibility for web-scale entity linking. We divide the system into three major stages and discuss their respective complexities.

Phrase Discovery. As a preprocessing step, this stage extracts phrases from free-style attribute values. Given an entity knowledge base, the phrase breaker routine finds a phrase segmentation which maximally reduces perplexity on a held-out portion of all free-style values in the knowledge base. For a knowledge base with 100 million entities, both of these procedures can be completed within one day using a single machine with 16 cores.

Building Entity Feature Model. In this stage, the system builds the phrase unigram language model, i.e., $\mathsf{P}(f|e)$, for every entity (including the unknown entity) in the knowledge base. We can build these models for 100 million entities in 20 minutes using a MapReduce cluster with thousands of nodes.

Entity linking. This stage invokes the PPP algorithm on web documents paired with the entity models. To obtain meaningful estimate of average complexity, we performed entity linking for more than 20 million documents and calculated the number of documents processed per node per second. On average, one node (with a single core) can process about 750 documents per second. In practice, we run thousands of nodes in parallel on a MapReduce cluster to process 10 billion web documents. In conclusion, the system is highly scalable for web-scale entity linking applications.

5. RELATED WORK

The major technique for disambiguating entity mentions in documents is to compare the contexts of both the mention in the document and an entity. Lexical features have proven very useful in previous entity linking and disambiguation works. In cross-document coreferencing, Bagga and Baldwin [2] compute the lexical similarity between the sentences extracted for each pair of documents based on a vector space model and threshold the similarity score to determine if two documents refer to the same entity. In addition to lexical features, during the past decade there has been much work that focused on the special structures available in an entity knowledge base. A representative example is Wikipedia, which not only contains texts describing the entities but also possesses rich structures such as redirect pages, disambiguation pages, categories, anchor texts and hyperlinks. These structures establish the relationships between entities, leading to systems built to exploit the correlations among entities [1, 5, 12, 25, 37, 30, 24, 19, 4]. Milne and Witten [30] consider the relatedness of the entities, which is measured in terms of the shared in-links to the entities' Wikipedia pages, in building a feature representation of the mentions to be revolved. Machine learning approaches are proposed to determine the weight among commonness, relatedness and context in the disambiguation module. The method reported in Medelyan et al. [28] computes the average semantic similarity between documents, which is also defined using the in-links structure in Wikipedia, and combine it with other features to resolve entity mentions. In Bunescu and Paşca's system [5], the context-article similarity, which is basically measured in the cosine similarity between the standard vector space models, and the taxonomy kernel, which essentially utilizes the Wikipedia categories to build feature vectors, are jointly exploited for designing a supervised learning approach for entity disambiguation using SVM. Hoffart et al. [20] models the popularity prior of entities, the context similarity between mentions and entities, and the coherence among entities. Then, a mention-entity graph is constructed with the edges representing the linear combination of the factors weighted by coefficients learned from a held-out data set. A dense subgraph is computed with each mention exactly linking with one entity, hence collectively disambiguate the entity mentions. These approaches build supervised learning systems to disambiguate entity mentions. However, unfortunately, it is infeasible to extend these supervised approaches for web-scale entity linking due to the size and the frequent updates of the knowledge bases, as well as the difficulty in obtaining high quality labeled data in the presence of vast ambiguity among entities.

Cucerzan [9] proposed an approach to solve the named entity disambiguation problem by optimizing the agreement between each of the mentions and the document in terms of contextual information as well as all pairwise agreements between mentions in terms of Wikipedia category information. Mihalcea and Csomai [29] considered linking mentions to the Wikipedia entities. In the link disambiguation process, they apply a knowledge-based approach which computes the contextual overlap between the entity definitions and the words to be disambiguated, and an approach that adopts features including part-of-speech, local contexts with specified locations, among others. A voting scheme is further devised based on these two orthogonal approaches to filter out incorrect predictions by seeking agreements between the two methods. Han et al. [19] consider a collective approach for disambiguate multiple entities in a web document using the Wikipedia knowledge base. Global interdependence between different entity disambiguations are modeled and a referent graph is constructed to jointly disambiguate the meanings of the entities in the web document by exploiting their semantic relatedness. Han and Sun [18] employs a generative entity-mention model for linking entity mentions to a knowledge base, which can incorporate the prior popularity of entities, name variations, and the context of the entity in the knowledge base. A Naive Bayes classifier is then employed based on the generative model to determine the best matching entity in the knowledge base. These techniques perform well on web documents with decent editorial quality, such as news or Wikipedia articles. However, for establishing entity mapping on the entire web, the heterogeneous edit qualities, various writing styles, and arbitrary content organizations present a new challenge beyond the capacity of existing systems.

The increasing popularity of social networks has attracted a series of research on entity recognition and disambiguation on social network data. In particular, entity linking in tweets against the Wikipedia entity knowledge base has become a fast growing research area [34, 27, 14, 16, 35]. In this setting, the tweets are viewed as very short and noisy "documents" unlike typical web documents which are generally longer and richer in useful context for disambiguating the entity mentions. To alleviate this problem, it is shown useful to consider a collection of tweets from one user and build a topical model that can draw the user's past tweeting preference to help understand new posts. Meanwhile, the relations among the entities, especially the categorical structures in Wikipedia, are commonly exploited to improve entity linking performance. Our work, in contrast, considers the entity linking at the tail. We focus on exploring the utility of lexical features only, since relations and connections among entities in a tail entity knowledge base may not exist or be poorly constructed.

Recently, web information processing has provided an emerging opportunity for the theories and algorithms of sparse signal recovery [6] to harness the large-scale data available on the web. In [22], Kasiviswanathan et al. apply the technique for sparse signal recovery to detect emerging topics in streaming user-generated contents. The novelty of a document can be characterized by the quality of a sparse representation. In [31], Min et al. utilize a low-rank and sparse matrix decomposition technique to separate the background topics from keywords on a set of documents. The topical background shared across multiple documents is modeled by a low-rank matrix, whereas the keywords specifically related to each individual document are captured by a sparse matrix. These works pioneered the exciting application of sparse signal recovery techniques to the web information processing. The techniques therein, however, are all based on the Euclidean distance as the metric for model fitting, which resides in the traditional sparse signal recovery regime. In contrast, our system employs a probabilistic distance metric, which is a highly innovative extension and a provably effective adaptation of sparse signal recovery to web information processing.

6. CONCLUSIONS

We built a web-scale entity linking system for tail entities on the web. At the core of the system is a posterior probability pursuit that exploits the sparse nature of entity linking, augmented by an unknown entity model characterizing the presence of out-of-knowledge-base entities, and the phrase unigram language model that effectively captures high-order dependencies among words. The system achieves the superior precision-recall tradeoff which is desired by many entity centric web applications.

For the past decade, the availability of Wikipedia, Freebase and other (head-focused) knowledge bases has enabled substantial advances in entity linking for popular, well-described entities. As that challenge comes closer to resolution, and new, large-scale, tail-focused knowledge bases — built from social networking and social media sites — become available, we believe focus will and must shift to linking tail entities in order to satisfy the breadth of requirements for next generation search and entity-centric applications. Our system is, to our knowledge, the first large-scale system to be designed for and demonstrate high performance entity linking at the tail.

7. REFERENCES

[1] S. F. Adafre and M. de Rijke. Discovering missing links in wikipedia. In *Proceedings of the 3rd International Workshop on Link Discovery*, 2005.

[2] A. Bagga and B. Baldwin. Entity-based cross-document co-referencing using the vector space model. In *Proceedings of ACL*, pages 79–85, 1998.

[3] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, (3):993–1022, 2003.

[4] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of SIGMOD*, 2008.

[5] R. Bunescu and M. Pasca. Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of EACL*, 2006.

[6] E. J. Candes. Compressive sampling. *Proc. Int. Congr. Mathematicians*, 2006.

[7] C.D.Manning, P. Raghavan, and H. Schutze. *Introduction to Information Retrieval*. Cambrige Univ. Press, 2008.

[8] S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. *Harvard University, TR-10-98*, 1998.

[9] S. Cucerzan. Large-scale named entity disambiguation based on wikipedia data. In *Proceedings of EMNLP*, pages 708–716, June 2007.

[10] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

[11] D. L. Donoho. Compressed sensing. *IEEE Trans. Inf. Theory*, 52(4):1289–1306, 2006.

[12] M. Dredze, P. McNamee, D. Rao, A. Gerber, and T. Finin. Entity disambiguation for knowledge base population. In *Proceedings of COLING*, 2010.

[13] M. Figueiredo, R. D. Nowak, and S. J. Wright. Gradient projection for sparse reconstruction: Application to

compressed sensing and other inverse problems. *IEEE JSTSP*, 1:586–597, 2007.

[14] A. Gattani, D. S. Lamba, N. Garera, M. Tiwari, X. Chai, S. Das, S. Subramaniam, A. Rajaraman, V. Harinarayan, and A. Doan. Entity extraction, linking, classification, and tagging for social media: a wikipedia-based approach. In *Proceedings of VLDB*, 2013.

[15] I. Good. The population frequency of species and the estimation of population parameters. *Biometrika*, 40(3):237–264, 1953.

[16] S. Guo, M.-W. Chang, and E. Kıcıman. To link or not to link? a study on end-to-end tweet entity linking. In *Proceedings of NAACL-HLT*, 2013.

[17] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of the Journal of Machine Learning Research*, 3:1157–1182, 2003.

[18] X. Han and L. Sun. A generative entity-mention model for linking entities with knowledge base. In *Proceedings of ACL-HLT*, pages 945–954, June 2011.

[19] X. Han, L. Sun, and J. Zhao. Collective entity linking in web text: A graph-based method. In *Proceedings of SIGIR*, pages 765–774, July 2011.

[20] J. Hoffart, M. A. Yosef, I. Bordino, H. Furstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. Robust disambiguation of named entities in text. In *Proceedings of EMNLP*, pages 782–792, 2011.

[21] T. Hofmann. Probabilistic latent semantic indexing. In *Proceedings of SIGIR*, 1999.

[22] S. Kasiviswanathan, P. Melville, A. Banerjee, and V. Sindhwani. Emerging topic detection using dictionary learning. In *Proceedings of CIKM*, 2011.

[23] S. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *Ieee Transactions On Acoustics, Speech, And Signal Processing*, 35(3):400–401, 1987.

[24] S. Kulkarni, A. Singh, G. Ramakrishnan, and S. Chakrabarti. Collective annotation of wikipedia entities in web text. In *Proceedings of KDD*, 2009.

[25] X. Li, P. Morie, and D. Roth. Identification and tracing of ambiguous names: discriminative and generative approaches. In *Proceedings of the 19th national conference on Artificial intelligence*, 2004.

[26] D. Lin and X. Wu. Phrase clustering for discriminative learning. In *Proceedings of ACL and AFNLP*, 2009.

[27] X. Liu, Y. Li, H. Wu, M. Zhou, F. Wei, and Y. Lu. Entity linking for tweets. In *Proceedings of ACL*, 2013.

[28] O. Medelyan, I. Witten, and D. Milne. Topic indexing with wikipedia. In *Proceedings of WIKIAI*, pages 19–24, 2008.

[29] R. Mihalcea and A. Csomal. Wikify! linking documents to encyclopedic knowledge. In *Proceedings of CIKM*, 2007.

[30] D. Milne and I. H. Witten. Learning to link with wikipedia. In *Proceedings of CIKM*, 2008.

[31] K. Min, Z. Zhang, J. Wright, and Y. Ma. Decomposing background topics from keywords by principal component pursuit. In *Proceedings of CIKM*, 2010.

[32] A. Ng. Feature selection, l1 vs. l2 regularization, and rotational invariance. In *Proceeding of ICML*, 2004.

[33] J. Ponte and W. Croft. A language modeling approach to information retrieval. In *Proceedings of SIGIR*, 1998.

[34] A. Ritter, S. Clark, Mausam, and O. Etzioni. Named entity recognition in tweets: An experimental study. In *Proceedings of EMNLP*, 2011.

[35] W. Shen, J. Wang, P. Luo, and M. Wang. Linking named entities in tweets with knowledge base via user interest modeling. In *Proceedings of KDD*, 2013.

[36] R. Tibshirani. Regression shrinkage and selection via the LASSO. *J. R. Statist. Soc. B*, 58(1):267–288, 1996.

[37] Y. Zhou, L. Nie, O. Rouhani-Kalleh, F. Vasile, and S. Gaffney. Resolving surface forms to wikipedia topics. In *Proceedings of COLING*, 2010.

[38] M. Zhu, S. Shi, N. Yu, and J. Wen. Can phrase indexing help to process non-phrase queries? In *Proceedings of CIKM*, 2008.