A Scalable Gibbs Sampler for Probabilistic Entity Linking

Neil Houlsby[1] *, and Massimiliano Ciaramita[2]

[1] University of Cambridge nmth2@cam.ac.uk
[2] Google Research, Zürich massi@google.com

Abstract. Entity linking involves labeling phrases in text with their referent entities, such as Wikipedia or Freebase entries. This task is challenging due to the large number of possible entities, in the millions, and heavy-tailed mention ambiguity. We formulate the problem in terms of probabilistic inference within a topic model, where each topic is associated with a Wikipedia article. To deal with the large number of topics we propose a novel efficient Gibbs sampling scheme which can also incorporate side information, such as the Wikipedia graph. This conceptually simple probabilistic approach achieves state-of-the-art performance in entity-linking on the Aida-CoNLL dataset.

1 Introduction

Much recent work has focused on the 'entity-linking' task which involves annotating phrases, also known as *mentions*, with unambiguous identifiers, referring to *topics*, *concepts* or *entities*, drawn from large repositories such as Wikipedia or Freebase. Mapping text to unambiguous references provides a first scalable handle on long-standing problems such as language *polysemy* and *synonymy*, and more generally on the task of *semantic grounding* for language understanding.

Most current approaches use heuristic scoring rules or machine-learned models to rank candidate entities. In contrast, we cast the entity-linking problem as inference in a probabilistic model. This probabilistic interpretation has a number of advantages: (i) The model provides a principled interpretation of the objective function used to rank candidate entities. (ii) One gets automatic confidence estimates in the predictions returned by the algorithm. (iii) Additional information can be incorporated into the algorithm in a principled manner by extending the underlying model rather than hand tuning the scoring rule. (iv) In practice, probabilistic inference is often found to be less sensitive to the auxiliary parameters of the algorithm. Finally, our method has the advantage of being conceptually simple compared to many state-of-the-art entity-linking systems, but still achieves comparable, or better, performance.

The model underlying the linking algorithm presented here is based upon Latent Dirichlet Allocation (LDA) [1]. In a traditional LDA model, the topics have no inherent interpretation; they are simply collections of related words. Here

* Work carried out during an internship at Google.

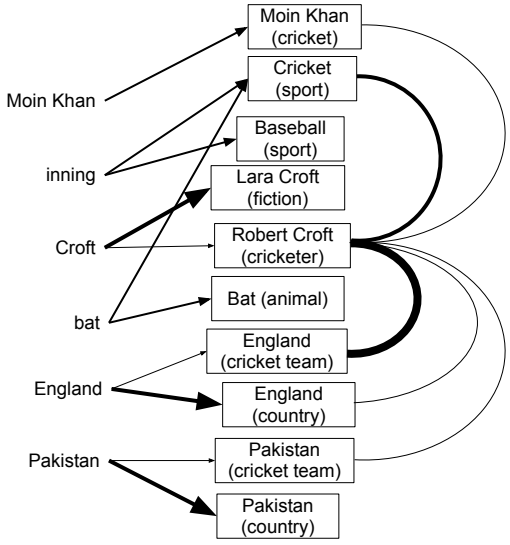

Fig. 1. Example of document-Wikipedia graph.

we construct an LDA model in which each topic is associated with a Wikipedia article. Using this 'Wikipedia-interpretable' LDA model we can use the topic-word assignments discovered during inference directly for entity linking. The topics are constructed using Wikipedia, and the corresponding parameters remain fixed. This model has one topic per Wikipedia article, resulting in over 4 million topics. Furthermore, the vocabulary size, including mention unigrams and phrases, is also in the order of millions.To ensure efficient inference we propose a novel Gibbs sampling scheme that exploits sparsity in the Wikipedia-LDA model. To better identify document-level consistent topic assignments, we introduce a 'sampler-memory' heuristic and propose a simple method to incorporate information from the Wikipedia in-link graph in the sampler. Our model achieves the best performance in entity-linking to date on the Aida-CoNLL dataset [2].

2 Background and Related Work

Much recent work has focused on associating textual mentions with Wikipedia topics [2–9]. The task is known as *topic annotation*, *entity linking* or *entity disambiguation*. Most of the proposed solutions exploit sources of information compiled from Wikipedia: the link graph, used to infer similarity measures between topics, anchor text, to estimate how likely a string is to refer to a given topic, and finally, to a lesser extent so far, local textual content.

Figure 1 illustrates the main intuitions behind most annotators' designs. The figure depicts a few words and names from a news article about cricket. Connections between strings and Wikipedia topics are represented by arrows whose line

weight represents the likelihood of that string mentioning the connected topic. In this example, a priori, it is more likely that "Croft" refers to the fictional character rather than the cricket player. However, a similarity graph induced from Wikipedia[3] would reveal that the cricket player topic is actually densely connected to several of the candidate topics on the page, those related to cricket (again line weight represents the connection strength). Virtually all topic annotators propose different ways of exploiting these ingredients.

Extensions to LDA for modeling both words and *observed* entities have been proposed [10, 11]. However, these methods treat entities as *strings*, not linked to a knowledge base. [5, 12, 13] propose LDA-inspired models for documents consisting of words and mentions being generated from distributions identified with Wikipedia articles. Only Kataria *et al.* investigate use of the Wikipedia category graph as well [12]. These works focus on both training the model and inference using Gibbs sampling, but do not exploit model sparsity in the sampler to achieve fast inference. Sen limits the topic space to $17k$ Wikipedia articles [13]. Kataria *et al.* propose a heuristic topic-pruning procedure for the sampler, but they still consider only a restricted space of $60k$ entities. Han and Sun propose a more complex hierarchical model and perform inference using incremental Gibbs sampling rather than with pre-constructed topics [5]. Porteous *et al.* speed up LDA Gibbs sampling by bounding on the normalizing constant of the sampling distribution [14]. They report up to 8 times speedup on a few thousand topics. Our approach exploits sparsity in the sampling distribution more directly and can handle millions of topics. Hansen *et al.* perform inference with, fewer, fixed topics [15]. We focus upon fast inference in this regime. Our algorithm exploits model sparsity without the need for pruning of topics. A preliminary investigation of a full distributed framework that includes re-estimation of the topics for the Wikipedia-LDA model is presented in [16].

3 Entity Linking with LDA

We follow the task formulation and evaluation framework of [2]. Given an input text where entity mentions have been identified by a pre-processor, e.g. a named entity tagger, the goal of a system is to disambiguate (link) the entity mentions with respect to a Wikipedia page. Thus, given a snippet of text such as "[Moin Khan] returns to lead [Pakistan]" where the NER tagger has identified entity mentions "Moin Kahn" and "Pakistan", the goal is to assign the cricketer id to the former, and the national cricket team id to the latter.

We are given a collection of D documents to be annotated, $\mathbf{w}_d$ for $d = 1, \ldots, D$. Each document is represented by a bag of L_d words, taken from a vocabulary of size V. The entity-linking task requires annotating only the mentions, and not the other words in the document (content words). Our model does not distinguish these, and will annotate both. As well as single words, mentions can be N-gram phrases as in the example "Moin Kahn" above. We assume the

[3] The similarity measure is typically symmetric.

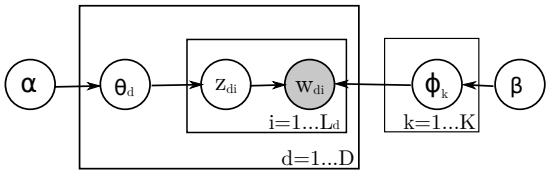

Fig. 2. Graphical model for LDA.

segmentation has already been performed using an NER tagger. Because the model treats mentions and content words equally, we use the term 'word' to refer to either type, and it includes phrases.

The underlying modeling framework is based upon LDA, a Bayesian model, commonly used for text collections [1]. We review the generative process of LDA below, the corresponding graphical model is given in Figure 2.

1. For each topic k, sample a distribution over the words $\phi_k \sim \text{Dir}(\beta)$.
2. For each document d sample a distribution over the topics $\theta_d \sim \text{Dir}(\alpha)$.
3. For each content word i in the document:
   (a) Sample a topic assignment: $z_i \sim \text{Multi}(\theta_d)$.
   (b) Sample the word from topic z_i: $w_j \sim \text{Multi}(\phi_{z_i})$.

The key modeling extension that allows LDA to be used for entity-linking is to associate each topic k directly with a single Wikipedia article. Thus the topic assignments z_i can be used directly to annotate entity mentions. Topic identifiability is achieved via the model construction; the model is built directly from Wikipedia such that each topic corresponds to an article (details in Section 5.1). After construction the parameters are not updated, only inference is performed.

Inference in LDA involves computing the topic assignments for each word in the document $\mathbf{z}_d = \{z_1, \ldots, z_{L_d}\}$. Each z_i indicates which topic (entity) is assigned to the word w_i. For example, if w_i = "Bush", then z_i could label this word with the topic "George Bush Sn.", "George Bush Jn.", or "bush (the shrub)" etc. The model must decide on the assignment based upon the context in which w_i is observed. LDA models are parametrized by their topic distributions. Each topic k is a multinomial distribution over words with parameter vector ϕ_k. This distribution puts high mass on words associated with the entity represented by topic k. In our model each topic corresponds to a Wikipedia entity, therefore the number of topic-word distributions, K, is large ($\approx$ 4M).

To characterize uncertainty in the choice of parameters most LDA models work with distributions over topics. Therefore, instead of storing topic multinomials ϕ_k (as in EDA [15]) we use Dirichlet distributions over the multinomial topics. That is, $\phi_k \sim \text{Dir}(\lambda_k)$, where λ_k are V-dimensional Dirichlet parameter vectors. The set of all vectors $\lambda_1, \ldots, \lambda_K$ represents the model. These Dirichlet distributions capture both the average behavior and the uncertainty in each topic. Intuitively, each element λ_{kv} governs the prevalence of vocabulary word v in topic k. For example, for the topic "Apple Inc." λ_{kv} will be large for words such as "Apple" and "Cupertino". The parameters need not sum to one, $||\lambda_k||_1 \neq 1$,

but the greater the values, the lower the variance of the distribution, that is, the more it concentrates around its mean topic.

Most topics will only have a small subset of words from the large vocabulary associated with them, that is, topic distributions are *sparse*. However, the model would not be robust if we were to rule out all possibility of assigning a particular topic to a new word – this would correspond to setting $\lambda_{kv} = 0$. Thus, each parameter takes at least a small minimum value β. Due to the sparsity, most λ_{kv} will take value β. To save memory we represent the model using 'centered' parameters, $\hat{\lambda}_{kv} = \lambda_{kv} - \beta$, most of which take value zero, and need not be stored explicitly. Formally, α, β are scalar hyper-parameters for the symmetric Dirichlet priors; they may be interpreted as topic and word 'pseudo-counts' respectively.

4 Efficient Inference with a Sparse Gibbs Sampler

The English Wikipedia contains around 4M articles (topics). The vocabulary size is around 11M. To cope with this vast parameter space we build a highly sparse model, where each topic only explicitly contains parameters for a small subset of words. Remember, during inference *any* topic could be associated with a word due to the residual probability mass from the hyper-parameter β.

The goal of probabilistic entity disambiguation is to infer the distribution over topics for each word, that is, compute $p(\mathbf{z}|\mathbf{w}, \hat{\lambda}_1, \ldots, \hat{\lambda}_K)$. This distribution is intractable, therefore, one must perform approximate Bayesian inference. Two popular approaches are to use Gibbs sampling [17] or variational Bayes [1]. We use Gibbs sampling, firstly, because it allows us to exploit model sparsity, and secondly, it provides a simple framework into which we may incorporate side information in a scalable manner. During inference, we wish to compute the topic assignments. To do this, Gibbs sampling involves sampling each assignment in turn conditioned on the other current assignments and the model, $z_i \sim p(z_i|\mathbf{z}^{\setminus i}, w_i, \hat{\lambda}_1, \ldots \hat{\lambda}_K) = \int p(z_i|w_i, \theta_d, \hat{\lambda}_1, \ldots \hat{\lambda}_K) p(\theta_d|z^{\setminus i}) d\theta_d$. Here, we integrate (collapse) out θ_d, rather than sample this variable in turn. Collapsed inference is found to yield faster mixing in practice for LDA [17, 18].

We adopt the sampling distribution that results from performing variational inference over all of the variables and parameters of the model. Although we only consider inference of the assignments with fixed topics here, this sampler can be incorporated into a scalable full variational Bayesian learning framework [16], a hybrid variational Bayes – Gibbs sampling approach originally proposed in [19]. Following [1, 16, 19], the sampling distribution for z_i is:

$$p(z_i = k|\mathbf{z}^{\setminus i}, w_i\lambda_1, \ldots, \lambda_K) \propto (\alpha + N_k^{\setminus i}) \exp\{\Psi(\beta + \hat{\lambda}_{kw_i}) - \Psi(V\beta + \sum_v \hat{\lambda}_{kv})\}, \quad (1)$$

where $N_k^{\setminus i} = \sum_{j \neq i} \mathbb{I}[z_j = k]$ counts the number of times topic k has been assigned in the document, not including the current word w_i. $\Psi()$ denotes the Digamma function. The sampling distribution is dependent upon both the current word w_i and the current topic counts $N_k^{\setminus i}$, therefore, naïvely one must

re-compute its normalizing constant for every Gibbs sample. The distribution has K terms, and so this would be very expensive in this model. We therefore propose using the following rearrangement of Eqn. (1) that exploits the model and topic-count sparsity to avoid performing $\mathcal{O}(K)$ operations per sample:

$$p(z_i = k|\mathbf{z}^{\setminus i}, w_i, \hat{\lambda}_1, \ldots, \hat{\lambda}_K) \propto \underbrace{\frac{\alpha \exp\{\Psi(\beta)\}}{\kappa'_k}}_{\mu_k^{(d)}} + \underbrace{\frac{\alpha\kappa_{kw_i}}{\kappa'_k}}_{\mu_k^{(v)}} + \underbrace{\frac{N_k^{\setminus i} \exp\{\Psi(\beta)\}}{\kappa'_k}}_{\mu_k^{(c)}} + \underbrace{\frac{N_k^{\setminus i}\kappa_{kw_i}}{\kappa'_k}}_{\mu_k^{(c,v)}}, \tag{2}$$

where $\kappa_{kw} = \exp\{\Psi(\beta + \hat{\lambda}_{kw})\} - \exp\{\Psi(\beta)\}$ and $\kappa'_k = \exp\{\Psi(V\beta + \sum_v \hat{\lambda}_{kv})\}$ are transformed versions of the parameters. Clearly $\hat{\lambda}_{kv} = 0$ implies $\kappa_{kv} = 0$. κ'_k is dense. The distribution is now decomposed into four additive components: $\mu_k^{(d)}, \mu_k^{(v)}, \mu_k^{(c)}, \mu_k^{(c,v)}$, whose normalizing constants can be computed independently. $\mu_k^{(d)}$ is dense, but it can be pre-computed once before sampling. For each word we have a term $\mu_k^{(v)}$ which only has mass on the topics for which $\kappa_{kv} \neq 0$; this can be pre-computed for each unique word v in the document, again just once before sampling. $\mu_k^{(c)}$ only has mass on the topics currently observed in the document, i.e. those for which $N_k^{\setminus i} \neq 0$. This term must be updated at every sampling iteration, but this can be done incrementally. $\mu_k^{(c,v)}$ is non-zero only for topics which have non-zero parameters and counts. It is the only term that must be fully recomputed at every iteration. To compute the normalizing constant of the Eqn. (2), the normalizer of each component is computed when the component is constructed, and so all $\mathcal{O}(K)$ sums are performed in the initialization.

Algorithm 1 summarizes the sampling procedure. The algorithm is passed the document $\mathbf{w}_d$, initial topic assignment vector $\mathbf{z}_d^{(0)}$, and transformed parameters κ'_k, κ_{kv}. Firstly, the components of the sampling distribution in (2) that are independent of the topic counts $(\mu^{(d)}, \mu^{(v)})$ and their normalizing constants $(\mathcal{Z}^{(d)}, \mathcal{Z}^{(v)})$ are pre-computed (lines 2-3). This is the only stage at which the full dense K–dimensional vector $\mu^{(d)}$ needs to be computed. Note that one only computes $\mu_k^{(v)}$ for the words in the current document, not for the entire vocabulary. In lines 4-5, two counts are initialized from $\mathbf{z}^{(0)}$. N_{ki} contains the number of times topic k is assigned to word w_i, and N_k counts total number of occurrences of each topic in the current assignment. Both counts will be sparse as most topics are not sampled in a particular document. While sampling, the first operation is to subtract the current topic from N_k in line 8. Now that the topic count has changed, the two components of Eqn. (2) that are dependent on this count $(\mu_k^{(c)}, \mu_k^{(c,v)})$ are computed. $\mu_k^{(c)}$ can be updated incrementally, but $\mu_k^{(c,v)}$ must be re-computed as it is word-dependent. The four components and their normalizing constants are summed in lines 13-14, and a new topic assignment to w_i is sampled in line 15. N_{ki} is incremented in line 17 if burn-in is complete (due to the heuristic initialization we find $B = 0$ works well). If the topic has changed since the previous sweep then N_k is updated accordingly (line 20).

The key to efficient sampling from the multinomial in line 15 is to visit μ_k in order $\{k \in \mu_k^{(c,v)}, k \in \mu_k^{(c)}, k \in \mu_k^{(v)}, k \in \mu_k^{(d)}\}$. A random schedule would require on average $K/2$ evaluations of μ_k. However, if the distribution is skewed, with most of the mass on the topics contained in the sparse components, then much fewer evaluations are required if these topics are visited first. The degree of skewness is governed by the initialization of the parameters, and the priors α, β. In our experiments (see Section 6) we found that we visited on average 4-5 topics per iteration. Note that we perform no approximation or pruning, we still sample from the exact distribution Multi($\mu/\mathcal{Z}$). After completion of the Gibbs sweeps, the distribution of the topic assignments to each word is computed empirically from the sample counts in line 24.

Algorithm 1 Efficient Gibbs Sampling

1: **input**: $(\mathbf{w}_d, \mathbf{z}_d^{(0)}, \{\kappa_{kv}\}, \{\kappa'_k\})$
2: $\mu_k^{(d)} \leftarrow \alpha e^{\Psi(\beta)}/\kappa'_k$, $\mathcal{Z}^{(d)} \leftarrow \sum_k \mu_k^{(d)}$ ▷ Pre-compute dense component of Eqn. (2).
3: $\mu_k^{(v)} \leftarrow \alpha\kappa_{kv}/\kappa'_k$, $\mathcal{Z}^{(v)} \leftarrow \sum_k \mu_k^{(v)}$ $\forall v \in \mathbf{w}_d$
4: $N_{ki} \leftarrow \mathbb{I}_{z_i^{(0)}=k}$ ▷ Initial counts.
5: $N_k \leftarrow \sum_{i=1}^{L_d} N_{ki}$
6: **for** $s \in 1, \ldots, S$ **do** ▷ Perform S Gibbs sweeps.
7: **for** $i \in 1, \ldots, L_d$ **do** ▷ Loop over words in document.
8: $N_k^{\setminus i} \leftarrow N_k - \mathbb{I}_{z_i=k}$ ▷ Remove topic z_i from counts.
9: $\mu_k^{(c)} \leftarrow N_k^{\setminus i} e^{\Psi(\beta)}/\kappa'_k$ ▷ Compute sparse components of Eqn. (2).
10: $\mu_k^{(c,v)} \leftarrow N_k^{\setminus i}\kappa_{kw_i}/\kappa'_k$
11: $\mathcal{Z}^{(c)} \leftarrow \sum_k \mu_k^{(c)}$ ▷ Compute corresponding normalizing constants.
12: $\mathcal{Z}^{(c,v)} \leftarrow \sum_k \mu_k^{(c,v)}$
13: $\mu_k \leftarrow \mu_k^{(d)} + \mu_k^{(v)} + \mu_k^{(c)} + \mu_k^{(c,v)}$
14: $\mathcal{Z} \leftarrow \mathcal{Z}^{(d)} + \mathcal{Z}^{(v)} + \mathcal{Z}^{(c)} + \mathcal{Z}^{(c,v)}$
15: $z_i^{(s)} \sim \text{Multi}(\{\mu_k/\mathcal{Z}\}_{k=1}^K)$ ▷ Sample topic.
16: **if** $s > B$ **then** ▷ Discard burn in.
17: $N_{z_i^{(s)}i} \leftarrow N_{z_i^{(s)}i} + 1$ ▷ Update counts.
18: **end if**
19: **if** $z_i^{(s)} \neq z_i^{(s-1)}$ **then**
20: update N_k for $k \in \{z_i^{(s)}, z_i^{(s-1)}\}$ ▷ Update incrementally.
21: **end if**
22: **end for**
23: **end for**
24: $p(z_i = k|w_i) \leftarrow \frac{1}{S-B} N_{ki}$
25: **return**: $p(z_i = k|w_i)$ ▷ Return empirical distribution over topics.

4.1 Incorporating Memory and the Wikipedia Graph

When working with very large topic spaces, the sampler will take a long time to explore the full topic space and an impractical number of samples will be

required to achieve convergence. To address this issue we augment the sampler with a 'sampler memory' heuristic and information from the Wikipedia graph.

After a good initialization (see Section 5.2), to help the sampler stay on track we include the current sample in the topic counts when evaluating (2). Allowing the sampler to 'remember' the current assignment assists it in remaining in regions of good solutions. With memory the current effective topic-count is given by $N_k^{\setminus i} \leftarrow N_k \mathrm{coh}(z_k|w_i)$. An even better solution might be to include here an appropriate temporal decaying function, but we found this simple implementation yields strong empirical performance already.

We also exploit the Wikipedia-interpretability of the topics to readily include the graph into our sampler to further improve performance. Intuitively, we would like to weight the probability of a topic by a measure of its consistency with the other topics in the document. This is in line with the Gibbs sampling approach where, by construction, all other topic assignments are known. For this purpose we use the following coherence score [4] for the word at location i:

$$\mathrm{coh}(z_k|i) = \frac{1}{|\{\mathbf{z}_d\}| - 1} \sum_{k' \in \{\mathbf{z}_d\}^{\setminus i}} \mathrm{sim}(z_k, z_{k'})\,. \tag{3}$$

where $\{\mathbf{z}_d\}$ is the set of topics in the assignment $\mathbf{z}_d$, and $\mathrm{sim}(z_k, z_{k'})$ is the 'Google similarity' [20] between two Wikipedia pages. We include the coherence score by augmenting $N_k^{\setminus i}$ in Eqn. 2 with this weighting function, i.e. line 8 in Algorithm 1 becomes $N_k^{\setminus i} \leftarrow (N_k - \mathbb{I}_{z_i=k})\mathrm{coh}(z_k|w_i)$.

Notice that the contributions of the graph-coherence and memory components are incorporated into the computation of the normalizing constant. Incorporating the graph and memory directly into the sampler provides cheap and scalable extensions which yield improved performance. However, it would be desirable to include such features more formally in the model, for example, by including the graph via hierarchical formulations, or appropriate document-specific priors α in stead of the memory. We leave this to future research.

5 Model and Algorithmic Details

5.1 Construction of the Model

We construct models from the English Wikipedia. An article is an admissible topic if it is not a disambiguation, redirect, category or list page.This step selects approximately 4M topics.Initial candidate word strings for a topic are generated from its title, the titles of all Wikipedia pages that redirect to it, and the anchor text of all its incoming links (within Wikipedia). All strings are lower-cased, single-character mentions are ignored. This amounts to roughly 11M words and 13M parameters. Remember, 'words' also includes mention phrases. This initialization is highly sparse - for most word-topic pairs, $\hat{\lambda}_{kv}$ is set to zero. The parameters $\hat{\lambda}_{kv}$ are initialized using the empirical distributions from Wikipedia counts, that is, we set $\hat{\lambda}_{kv} = P(k|v) - \beta = \frac{\mathrm{count}(v,k)}{\mathrm{count}(v)} - \beta$. Counts are collected

from titles (including redirects) and anchors. We found that initializing the parameters using $P(v|k)$, rather than $P(k|v)$ yields poor performance because the normalization by count(k) in this case penalizes popular entities too heavily.

5.2 Sampler initialization

A naive initialization of the Gibbs sampler could use the topic with the greatest parameter value for a word $z_i^{(0)} = \arg\max_k \lambda_{kv}$, or even random assignments. We find that these are not good solutions because the distribution of topics for a word is typically long-tailed. If the true topic is not the most likely one, its parameter value could be several orders of magnitude smaller than the primary topic. Topics have extremely fine granularity and even with sparse priors it is unlikely that the sampler will converge to the the right patterns of topic mixtures in reasonable time. We improve the initialization with a simpler, but fast, heuristic disambiguation algorithm, TagMe [4]. We re-implement TagMe and run it to initialize the sampler, thus providing a good set of initial assignments.

6 Experiments

We evaluate performance on the CoNLL-Aida dataset, a large public dataset for evaluation of entity linking systems [2]. The data is divided in three partitions: train (946 documents), test-a (216 documents, used for development) and test-b (231 documents, used for blind evaluation). We report *micro-accuracy*: the fraction of mentions whose predicted topic is the same as the gold-standard annotation. There are 4,788 mentions in test-a and 4,483 in test-b. We also report *macro-accuracy*, where document-level accuracy is averaged over the documents.

6.1 Algorithms

The baseline algorithm (Base) predicts for mention w the topic k maximizing $P(k|w)$, that is, it uses only empirical mention statistics collected from Wikipedia.This baseline is quite high due to the skewed distribution of topics – which makes the problem challenging. TagMe* is our implementation of TagMe, that we used to initialize the sampler. We also report the performance of two state-of-the-art systems: the best of the Aida systems on test-a and test-b, extensively benchmarked in [2] (Aida13)[4], and finally the system described in [9] (S&Y13) which reports the best micro precision on the CoNLL test-b set to date. The latter reference reports superior performance to a number of modern systems, including those in [21, 2, 3]. We also evaluate the contributions of the components to our algorithm. WLDA-base uses just the sparse sampler proposed in Section 4. WLDA-mem includes the sampler memory, and WLDA-full incorporates both the memory and the graph.

[4] We report figures for the latest best model ("r-prior sim-k r-coh") from the Aida web site, `http://www.mpi-inf.mpg.de/yago-naga/aida/`. We are grateful to Johannes Hoffart for providing us with the development set results of the Aida system.

Table 1. Accuracy on the CoNLL-Aida corpus. In each row, the best performing algorithm, and those whose performance is statistically indistinguishable from the best, are highlighted in bold. Error bars indicate ±1 standard deviation. An empty cell indicates that no results are reported.

				test-a			
	Base	TagMe*	Aida13	S&Y13	WLDA-base	WLDA-mem	WLDA-full
Micro	70.76	76.89	**79.29**	-	75.21 ± 0.57	78.99 ± 0.50	$\mathbf{79.65 \pm 0.52}$
Macro	69.58	74.57	**77.00**	-	74.51 ± 0.55	$\mathbf{76.10 \pm 0.72}$	$\mathbf{76.61 \pm 0.72}$
				test-b			
Micro	69.82	78.64	82.54	**84.22**	78.75 ± 0.54	$\mathbf{84.88 \pm 0.47}$	$\mathbf{84.89 \pm 0.43}$
Macro	72.74	78.21	81.66	-	79.18 ± 0.71	$\mathbf{83.47 \pm 0.61}$	$\mathbf{83.51 \pm 0.62}$

6.2 Hyper-parameters

We set hyper-parameters, α, β and S using a greedy search that optimizes the sum of the micro and macro scores on both the train and test-a partitions. Setting α, β is a trade-off between sparsity and exploration. Smaller values result in sparser sampling distributions but larger α allows the model to visit topics not currently sampled and larger β lets the model sample topics with parameter values $\hat{\lambda}_{kv}$ equal to zero. We found that comparable performance can be achieved using a wide range of values: $\alpha \in [10^{-5}, 10^{-1}]$, $\beta \in [10^{-7}, 10^{-3})$. Regarding the sweeps, performance starts to plateau at $S = 50$. The robustness of the model's performance to these wide ranges of hyper-parameter settings advocates the use of this type of probabilistic approach. As for TagMe's hyper-parameters, in our experiments ϵ and τ values around 0.25 and 0.01 respectively worked best.

6.3 Results and Discussion

Table 1 summarizes the evaluation results. Confidence intervals are estimated using bootstrap re-sampling, and statistical significance is assessed using a unpaired t-test at the 5% significance level. Overall, WLDA-full, produces state-of-the-art results on both development (test-a) and blind evaluation (test-b). Table 1 shows that Base and Tagme*, used for model construction and sampler initialization respectively, are significantly outperformed by the full system. TagMe includes the information contained in Base and performs better, particularly on test-a. The gap between TagMe and WLDA-full is greatest on test-b. This is probably because the parameters are tuned on test-a, and are kept fixed for test-b and the proposed probabilistic method is more robust to the parameter values. The inclusion of memory produces a large performance gains and inclusion of the graph adds some further improvements, particularly on test-a.

In all cases we perform as well as, or better than, the current best systems. This result is particularly remarkable due to the simplicity of our approach. The S&Y13 system addresses the broader task of entity linking and named entity recognition. They train a supervised model from Freebase, using extensively engineered feature vectors. The Aida systems incorporate a significant amount of

knowledge from the YAGO ontology, that is, they also know the *type* of the entity being disambiguated. Our algorithm is conceptually simple and requires no training or additional resources beyond Wikipedia, nor hand crafting of features or scoring rules. Our approach is based upon Bayesian inference with a model created from simple statistics taken from Wikipedia. It is therefore remarkable that we are performing favorably against the best systems to date an this provides strong motivation to extend this probabilistic approach further.

Inspection of errors on the development partitions reveals scenarios in which further improvements can be made. In some documents, a mention can appear multiple times with different gold annotations. E.g. in one article, 'Washington' appears multiple times, sometimes annotated as the city, and sometimes as USA (country); in another, 'Wigan' is annotated both as the UK town and its rugby club. Due to the 'bag-of-words' assumption, LDA is not able to discriminate such cases and naturally tends to commit to one assignment for all occurrences of a string in a document. Local context could help disambiguate these cases. Within our sampling framework it would be straightforward to incorporate contextual information e.g. via up-weighting of topics using a distance function.

7 Conclusion and Future Work

Topic models provide a principled, flexible framework for analyzing latent structure in text. These are desirable properties for a whole new area of work that is beginning to systematically explore semantic grounding with respect to web-scale knowledge bases such as Wikipedia and Freebase. We have proposed a Gibbs sampling scheme for inference in a static Wikipedia-identifiable LDA model to perform entity linking. This sampler exploits model sparsity to remain efficient when confronted with millions of topics. Further, the sampler is able to incorporate side information from the Wikipedia in-link graph in a straightforward manner. To achieve good performance it is important to construct a good model and initialize the sampler sensibly. We provide algorithms to address both of these issues and report state-of-the-art performance in entity-linking.

We are currently exploring two directions for future work. In the first, we seek to further refine the parameters of the model λ_{kv} from data. This requires training an LDA model on huge datasets, for which we must exploit parallel architectures [16]. In the second, we wish to simultaneously infer the segmentation of the document into words/mentions and the topic assignments through use of techniques such as blocked Gibbs sampling,

Acknowledgments

We would like to thank Michelangelo Diligenti, Yasemin Altun, Amr Ahmed, Marc'Aurelio Ranzato, Alex Smola, Johannes Hoffart, Thomas Hofmann and Kuzman Ganchev for valuable feedback and discussions.

References

1. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet Allocation. JMLR **3** (2003) 993–1022
2. Hoffart, J., Yosef, M.A., Bordino, I., Fürstenau, H., Pinkal, M., Spaniol, M., Taneva, B., Thater, S., Weikum, G.: Robust disambiguation of named entities in text. In: EMNLP, ACL (2011) 782–792
3. Kulkarni, S., Singh, A., Ramakrishnan, G., Chakrabarti, S.: Collective annotation of wikipedia entities in web text. In: SIGKDD, ACM (2009) 457–466
4. Ferragina, P., Scaiella, U.: TagMe: On-the-fly annotation of short text fragments (by wikipedia entities). In: CIKM, ACM (2010) 1625–1628
5. Han, X., Sun, L.: An entity-topic model for entity linking. In: EMNLP-CoNLL, ACL (2012) 105–115
6. Mihalcea, R., Csomai, A.: Wikify!: Linking documents to encyclopedic knowledge. In: CIKM, ACM (2007) 233–242
7. Milne, D., Witten, I.H.: Learning to link with Wikipedia. In: CIKM, ACM (2008) 509–518
8. Ratinov, L.A., Roth, D., Downey, D., Anderson, M.: Local and global algorithms for disambiguation to wikipedia. In: ACL. Volume 11. (2011) 1375–1384
9. Sil, A., Yates, A.: Re-ranking for joint named-entity recognition and linking. In: CIKM. (2013)
10. Newman, D., Chemudugunta, C., Smyth, P.: Statistical entity-topic models. In: SIGKDD, ACM (2006) 680–686
11. Kim, H., Sun, Y., Hockenmaier, J., Han, J.: Etm: Entity topic models for mining documents associated with entities. In: Data Mining (ICDM), 2012 IEEE 12th International Conference on, IEEE (2012) 349–358
12. Kataria, S.S., Kumar, K.S., Rastogi, R.R., Sen, P., Sengamedu, S.H.: Entity disambiguation with hierarchical topic models. In: SIGKDD, ACM (2011) 1037–1045
13. Sen, P.: Collective context-aware topic models for entity disambiguation. In: Proceedings of the 21st international conference on World Wide Web, ACM (2012) 729–738
14. Porteous, I., Newman, D., Ihler, A., Asuncion, A., Smyth, P., Welling, M.: Fast collapsed gibbs sampling for latent dirichlet allocation. In: SIGKDD, ACM (2008) 569–577
15. Hansen, J.A., Ringger, E.K., Seppi, K.D.: Probabilistic explicit topic modeling using wikipedia. In: Language Processing and Knowledge in the Web. Springer (2013) 69–82
16. Houlsby, N., Ciaramita, M.: Scalable probabilistic entity-topic modeling. arXiv preprint arXiv:1309.0337 (2013)
17. Griffiths, T.L., Steyvers, M.: Finding scientific topics. PNAS **101**(Suppl 1) (2004) 5228–5235
18. Teh, Y.W., Newman, D., Welling, M.: A collapsed variational bayesian inference algorithm for latent dirichlet allocation. NIPS **19** (2007) 1353
19. Mimno, D., Hoffman, M., Blei, D.: Sparse stochastic inference for latent dirichlet allocation. In Langford, J., Pineau, J., eds.: ICML, New York, NY, USA, Omnipress (July 2012) 1599–1606
20. Milne, D., Witten, I.: An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. In: AAAI Workshop on Wikipedia and Artificial Intelligence. (2008)
21. Cucerzan, S.: Large-scale named entity disambiguation based on wikipedia data. In: EMNLP-CoNLL. Volume 7. (2007) 708–716